Министерство образования Республики Башкортостан
ГАОУ СПО Стерлитамакский колледж строительства, экономики и права

Учебно-методический комплекс

по разделу

«Объектно-ориентированное программирование»

МДК 01.02. «Прикладное программирование»

ПМ 01 Разработка программных модулей программного обеспечения для компьютерных систем

для групп специальности 230115
«Программирование в компьютерных системах»

Разработала преподаватель: ***В.Ф. Аришина***

2013

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Оглавление

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Современный этап развития среднего профессионального образования характеризуется переходом на практикоориентированную подготовку специалиста, которая обеспечивает прикладной характер знаний и умений выпускника, создает реальные возможности приступить к выполнению профессиональных обязанностей в соответствии с квалификацией по окончании учебного заведения. При этом теоретические знания обучающихся в СПО остаются на таком уровне что бы выпускник колледжа смог поступить в высшее учебное заведение и повысить свою квалификацию. Это требует особенного подхода к разработке содержания междисциплинарных курсов, чтобы сохранить баланс и выдержать процент самостоятельности в обучении, не потеряв в практическом опыте и уровне базовых теоретических знаний.

МДК 01.02 Р1 «Объектно-ориентированное программирование» включен в федеральный компонент государственного образовательного стандарта среднего профессионального образования по специальности 230115 «Программирование в компьютерных системах» от 23 июня 2010 г.. базовый уровень.

Данный МДК 02.01. Р1 «ООП» посвящен изучению отдельных разделов языка Pascal и основ визуального и объектно-ориентированного программирования. При разработке учебно-методического комплекса учитывается государственный стандарт по специальности 230115 «Программирование в компьютерных системах» и требования рынка труда в регионе. УМК для МДК 01.02 Р1 «ООП» состоит из программы курса, календарно-тематического плана, лекционного материала, примерных заданий на лабораторные занятия, программы, календарно-тематического плана и типового задания для учебной практики УП 02 «Разработка программного модуля в среде программирования» и методических рекомендаций для самостоятельной работы студентов.

Целью данного курса является обучение студентов основам визуального и объектно-ориентированного программирования, приобретение практического опыта по созданию программ на уровне модуля и написание к ней сопроводительной технической документации.

Задачи курса:

-изучение нового материала в рамках программы;

-приобретение практического опыта составления программного кода в изучаемой среде программирования;

-формирование умений составлять техническую документацию.

Применение учебно-методического комплекса позволит:

сформировать умения:

- осуществлять разработку кода программного модуля на современных языках программирования;
- создавать программу по разработанному алгоритму как отдельный модуль;
- выполнять отладку и тестирование программы на уровне модуля;
- оформлять документацию на программные средства;

освоить знания:

- основных принципов технологии структурного, объектно-ориентированного и визуального программирования;
- основные принципы отладки и тестирования программных продуктов,

а также приобрести практический опыт:

- разработки кода программного продукта на основе готовой спецификации на уровне модуля;
- использования инструментальных средств на этапе отладки программного продукта;
- проведения тестирования программно модуля по определенному сценарию.

Учебно-методический комплекс курса МДК 01.02 Р1 «Объектно-ориентированное программирование» представляет собой совокупность учебных, учебно-методических, контрольно-измерительных материалов, обеспечивающих организованную и содержательную целостность системы обучения, способствующих эффективному усвоению обучающимися учебной программы.

Представленный учебно-методический комплекс предназначен для преподавателей профессиональных модулей по образовательным программам группы специальностей 230000 «Информатика и вычислительная техника».

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

РАЗДЕЛ 1 РАБОЧАЯ ПРОГРАММА ПРОФЕССИОНАЛЬНОГО МОДУЛЯ 01

РАБОЧАЯ ПРОГРАММА ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

Разработка программных модулей программного обеспечения для компьютерных систем

Организация-разработчик:

ГАОУ СПО Стерлитамакский колледж строительства, экономики и права

Разработчики:

Хасанова А.Х., преподаватель высшей категории
Симакова Е.В., преподаватель первой категории
Аришина В.Ф., преподаватель первой категории
Маркова Ю.О., преподаватель высшей категории

2011 г.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1. ПАСПОРТ РАБОЧЕЙ ПРОГРАММЫ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ
Разработка программных модулей программного обеспечения для компьютерных систем

1.1. Область применения программы

Рабочая программа профессионального модуля – является частью рабочей основной профессиональной образовательной программы в соответствии с ФГОС по специальности СПО **230115 Программирование в компьютерных системах** в части освоения основного вида профессиональной деятельности (ВПД):

Разработка программных модулей программного обеспечения для компьютерных систем и соответствующих профессиональных компетенций (ПК):

1. Выполнять разработку спецификаций отдельных компонент.

2. Осуществлять разработку кода программного продукта на основе готовых спецификаций на уровне модуля.

3. Выполнять отладку программных модулей с использованием специализированных программных средств.

4. Выполнять тестирование программных модулей.

5. Осуществлять оптимизацию программного кода модуля.

6. Разрабатывать компоненты проектной и технической документации с использованием графических языков спецификаций.

Рабочая программа профессионального модуля может быть использована в профессиональной подготовке специальностей СПО **230100 Информатика и вычислительная техника** и в дополнительном профессиональном образовании повышения квалификации и переподготовки кадров в области разработки программного обеспечения.

1.2. Цели и задачи модуля – требования к результатам освоения модуля

С целью овладения указанным видом профессиональной деятельности и соответствующими профессиональными компетенциями обучающийся в ходе освоения профессионального модуля должен:

иметь практический опыт:

- разработки алгоритма поставленной задачи и реализации его средствами автоматизированного проектирования;
- разработки кода программного продукта на основе готовой спецификации на уровне модуля;

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

– использования инструментальных средств на этапе отладки программного продукта;
– проведения тестирования программно модуля по определенному сценарию;

уметь:

– осуществлять разработку кода программного модуля на современных языках программирования;
– создавать программу по разработанному алгоритму как отдельный модуль;
– выполнять отладку и тестирование программы на уровне модуля;
– оформлять документацию на программные средства;
– использовать инструментальные средства для автоматизации оформления документации;
– *применять языки гипертекстовой разметки для разработки ПО в Internet*;*
– *организовывать и обеспечивать технологический процесс информационного наполнения Web-сайтов*.*

знать:

– основные этапы разработки программного обеспечения;
– основные принципы технологии структурного и объектно-ориентированного программирования;
– основные принципы отладки и тестирования программных продуктов;
– методы и средства разработки технической документации;
– *специализированное ПО по проектированию и обработке информационного содержания*;*
– *принципы построения ПО в сети Internet*;*
– *принципы работы языков сценариев*.*

1.3. Рекомендуемое количество часов на освоение программы профессионального модуля:

всего – **651** часов, в том числе:

максимальной учебной нагрузки обучающегося – **507** часов, включая:

обязательную аудиторную учебную нагрузку обучающегося – **338** часов;

самостоятельную работу обучающегося – **169** часов;

учебную и производственную практики – **144** часа.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

2. РЕЗУЛЬТАТЫ ОСВОЕНИЯ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

Разработка программных модулей программного обеспечения для компьютерных систем

Результатом освоения программы профессионального модуля является овладение обучающимися видом профессиональной деятельности Разработка программных модулей программного обеспечения для компьютерных систем, в том числе профессиональными (ПК) и общими (ОК) компетенциями:

Код	Наименование результата обучения
ПК 1.1.	Выполнять разработку спецификаций отдельных компонент.
ПК 1.2.	Осуществлять разработку кода программного продукта на основе готовых спецификаций на уровне модуля.
ПК 1.3.	Выполнять отладку программных модулей с использованием специализированных программных средств.
ПК 1.4.	Выполнять тестирование программных модулей.
ПК 1.5.	Осуществлять оптимизацию программного кода модуля.
ПК 1.6.	Разрабатывать компоненты проектной и технической документации с использованием графических языков спецификаций.
ОК 1.	Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес.
ОК 2.	Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.
ОК 3.	Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.
ОК 4.	Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.
ОК 5.	Использовать информационно-коммуникационные технологии в профессиональной деятельности.
ОК 6.	Работать в коллективе и в команде, эффективно общаться с коллегами, руководством, потребителями.
ОК 7.	Брать на себя ответственность за работу членов команды (подчиненных), за результат выполнения заданий.
ОК 8.	Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.
ОК 9.	Ориентироваться в условиях частой смены технологий в профессиональной деятельности.
ОК 10.	Исполнять воинскую обязанность, в том числе с применением полученных профессиональных знаний (для юношей).

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

3. СТРУКТУРА И ПРИМЕРНОЕ СОДЕРЖАНИЕ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

3.1. Тематический план профессионального модуля

Разработка программных модулей программного обеспечения для компьютерных систем

Коды профессиональных компетенций	Наименования разделов профессионального модуля*	Всего часов	Объем времени, отведенный на освоение междисциплинарного курса (курсов)					Практика	
			Обязательная аудиторная учебная нагрузка обучающегося			Самостоятельная работа обучающегося		Учебная, часов	Производственная (по профилю специальности), часов *(если предусмотрена рассредоточенная практика)*
			Всего, часов	в т.ч. лабораторные работы и практические занятия, часов	в т.ч., курсовая работа (проект), часов	Всего, часов	в т.ч., курсовая работа (проект), часов		
1	2	3	4	5	6	7	8	9	10
ПК 1.1, ПК 1.6	**МДК 01.01. Системное программирование**	**207**	**138**	62		**69** *(7*)*			
	Раздел 1. Разработка программных модулей	**105**	**70**	30		**35**			
	Раздел 2. Разработка системного программного обеспечения	**102** *(21*)*	**68** *(14*)*	32 *(8*)*		**34** *(7*)*		**36**	
ПК 1.2., ПК 1.3, ПК 1.4, ПК 1.5.	**МДК 01.02. Прикладное программирование**	**300** *(108*)*	**200** *(72*)*	108 *(32*)*		**100** *(36*)*			
	Раздел 1. Объектно – ориентированное программирование	**192**	**128**	74	30	**64**	15	**36**	
	Раздел 2. *Разработка Web-приложений**	***108****	***72****	*32**		***36****			
	Производственная практика (по профилю специальности)	**72**							**72**
	Всего:	**651** *(129*)*	**338** *(86*)*	170 *(40*)*	30	**169** *(43*)*	15	**72**	**72**

* *в том числе из часов вариативной части*

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

3.2. Содержание обучения по профессиональному модулю (ПМ)

1	2		3	4
МДК 01.01 Системное программирование			**138**	
Раздел 1 Разработка программных модулей			**70**	
Тема 1.1. Программные продукты и их основные характеристики	**Содержание**		6	
	1	Основные понятия программного обеспечения		1
	2	Понятие жизненного цикла программы		2
	3	Основные этапы разработки программного обеспечения		2
Тема 1.2. Стадии разработки программ и программной документации	**Содержание**		4	
	1	Методы и средства разработки технической документации. Понятие о ЕСПД (ГОСТ 19.XXX-XX, ГОСТ 34.XXX-XX).		1
	2	Графические языки спецификаций		2
	Практические занятия		8	
	1	Построение структурной, информационной и алгоритмической модели программного компонента ОК 2.1, 2.2		
	2	Проектирование программного компонента ОК 2.1, 2.2		
	3	Оформление программной документации. Стадии «Техническое задание», «Эскизный проект» ОК 2.1, 2.2, 2.3.		
	4	Оформление программной документации. Стадии «Технический проект», «Реализация» ОК 2.1, 2.2, 2.3.		
Тема 1.3. Методы проектирования программных продуктов	**Содержание**		4	
	1	Принцип системного проектирования.		2
	2	Объектно-ориентированное проектирование программных продуктов.		2
Тема 1.4. Проектирование интерфейса поль-	**Содержание**		4	
	1	Интерфейс пользователя программного продукта ОК 6.1, 6.2, 6.3		1
	2	Требования, предъявляемые к стандартному графическому интерфейсу		2

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2		3	4
зователя		пользователя		
	Лабораторные занятия		2	
	1	Создание интерфейса пользователя в соответствии с требованиями к стандартному графическому интерфейсу ОК 2.1, 2.2, 2.3.		
Тема 1.5. Методы разработки программных модулей	**Содержание**		4	
	1	Сущность модульного программирования		2
	2	Основные принципы технологии структурного программирования ОК6.1		2
	Практические занятия		4	
	1	Проведение анализа программ с точки зрения стиля. Составление удобочитаемой программы. ОК 2.1, 2.2, 2.3.		
	2	Создание структурного алгоритма ОК 2.1, 2.2		
	Лабораторные занятия		2	
	1	Создание программы с применением методов структурирования ОК 2.2		
Тема 1.6. Объектно - ориентированное программирование	**Содержание**		4	
	1	Основные принципы технологии объектно-ориентированного программирования.		2
	2	Этапы объектно- ориентированного проектирования. ОК 6.1, 6.2		2
	Лабораторные занятия		2	
	1	Создание программы с применением методов объектно-ориентированного проектирования		
Тема 1.7. Эффективность и оптимизация программ	**Содержание**		4	
	1	Основные критерии эффективности программного продукта		2
	2	Принципы и приемы оптимизации. ОК 6.1, 6.2, 6.3		2
	Лабораторные занятия		2	
	1	Применение оптимизирующих компиляторов ОК 2.1, 2.2		
Тема 1.8. Отладка программ	**Содержание**		4	
	1	Понятие об ошибке программного обеспечения.		2
	2	Основные принципы отладки программных продуктов.		2
	Лабораторные занятия		2	
	1	Использование средств отладки программ ОК 2.1, 2.2, 2.3		
Тема 1.9. Методы тестирования программ	**Содержание**		4	
	1	Основные принципы тестирования программных продуктов		2
	2	Методы структурного и функционального тестирования программного обеспечения ОК 6.1, 6.2.		2

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2		3	4
	Лабораторные занятия		4	
	1	Тестирование программ методами «белого ящика»: покрытия операторов, покрытия решений, покрытия условий		
	2	Тестирование программ методами «белого ящика»: покрытия решений/условий, комбинаторного покрытия условий		
Тема 1.10. Сопровождение программ	**Содержание**		2	
	1	Виды программных документов. Методы разработки программной документации. Технология разработки документации DocBook		2
	Практические занятия		4	
	1	Использование инструментальных средств для автоматизации оформления документации.		
	2	Составление сопроводительной документации программного обеспечения с использованием технологии DocBook.		
Самостоятельная работа при изучении раздела 1 ПМ 01. 1. Реферирование темы «Вспомогательные процессы жизненного цикла программного продукта» 2. Аналитический обзор литературы по теме «Каскадная модель жизненного цикла разработки программного продукта» 3. Подготовка доклада по теме «Модель прототипирования жизненного цикла разработки программного продукта» 4. Подготовка доклада по теме «RAD-модель жизненного цикла разработки программного продукта» 5. Составление опорного конспекта по теме «V- образная модель жизненного цикла разработки программного продукта» 6. Реферирование темы «Многопроходная модель жизненного цикла разработки программного продукта» 7. Реферирование темы «Спиральная модель жизненного цикла разработки программного продукта» 8. Построение макетов программной документации по предложенным образцам 9. Составление кроссворда по теме «Методы проектирования программных продуктов» 10. Подготовка доклада по теме «Надежность программных продуктов» 11. Разработка мультимедийной презентации по теме «Методы разработки программных продуктов» 12. Аналитический обзор литературы по теме «Количественные характеристики надежности программ» 13. Подготовка доклада «Возможности встроенного отладчика интегрированной среды Turbo			**35**	

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2		3	4
Pascal» 14. Выполнение упражнения на применение простых приемов отладки программ в интегрированной среде Turbo Pascal 15. Аналитический обзор литературы по теме «ГОСТ 19.101—77» 16. Составление словаря терминов по разделу «Разработка программного модуля»				
Раздел 2 Разработка системного программного обеспечения			**68**	
Тема 2.1. Системное программирование	**Содержание**		8	
	1	Понятие программы, программного обеспечения . Требования к ПО		1
	2	Классификация программ по функциональному признаку		1
	3	Принципы модульной структуры		1
	4	Этапы подготовки программы		1
	Практические работы		2	
	1	Составление программы на языке автокод.		
	Лабораторные работы		2	
	1	Использование языка Ассемблер при составлении программы.		
Тема 2.2. Ассемблеры	**Содержание**		22	
	1	Система команд Ассемблера.		1
	2	Форматы данных Ассемблера.		1
	3	Режимы адресации		1
	4	Сегментация оперативной памяти. Понятие стека.		1
	5	Команды Ассемблера. Команды пересылки данных..		1
	6	Арифметические команды. Примеры программ		2
	7	Логические операции. Команды сдвига.		1
	8	Команды передачи управления. Примеры программ.		2
	9	Вектор прерываний. Команда обращения к подпрограмме CALL и команда обращения к системным средствам INT.		1
	10	Команды управления процессором. Программирование структуры вектора прерываний		2
	11	Отладка программ. Оптимизация программы на ассемблере		
	Практические работы:		10	

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2		3	4
	1	Выполнение задания с форматами команд Ассемблера.		
	2	Выполнение заданий на режимы адресации, сегментацию оперативной памяти. Отработка понятия стека и последующей модели по заданному алгоритму. (ОК 2.2.2)		
	3	Программирование с командами пересылки данных, с арифметические командами.(ОК 3.1.1)		
	4	Использование логических операций, команд сдвига, команд передачи управления в процессе программировании		
	5	Выбор метода отладки программ и метода тестирования		
	Лабораторные работы		10	
	1	Сложение и вычитание целых чисел со знаком в дополнительном коде.		
	2	Мнемоническое кодирование на языке ассемблера линейных, разветвляющихся и циклических алгоритмов.		
	3	Реализация алгоритмов работы со структурами данных: стеки, списки.		
	4	Выполнение основных операций и приемов программирования на Ассемблере.		
	5	Отладка программы, созданной на языке машинных кодов		
Тема 2.3 Макропрограммирование			***14****	
Самостоятельная работа при изучении раздела 2 ПМ 01 1. Реферирование темы «Архитектура RISC-процессора.» (ОК4.1.1) 2. Реферирование темы «TRS-программа.» 3. Реферирование темы «Программирование видеоадаптеров.» 4. Реферирование темы «Работа с окнами диалога Windows на Ассемблере» 5. Программирование функций работы с манипулятором «мышь». 6. Составление элементов программ на Ассемблере по индивидуальному заданию 7. Аналитический обзор литературы по теме. «Макроассемблер» 8. Составление элементов программ с командами пересылки данных на языке машинных кодов 9. Составление элементов программ с использованием математических операций и логических функций на языке машинных кодов 10. Составление элементов программ с использованием разветвляющих и циклических алгоритмов 11. Составление кроссворда по теме «Макроассемблеры»			**34** ***(7*)***	

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2		3	4
12. Разработка мультимедийной презентации по теме «конструкции языка машинных кодов» 13. Выполнение упражнения на применение простых приемов отладки программ на языке машинных кодов				
Учебная практика «Ассемблирование программного обеспечения компьютерных систем» **Виды работ** Инсталлирование программного обеспечения Выполнение сегментации оперативной памяти Управление процессором на языке машинных кодов Тестирование модулей программного обеспечения на языке машинных кодов Отладка модулей программного обеспечения на языке машинных кодов Использование функций настройки ПО на уровне машинных кодов			**36**	
МДК 01.02 Прикладное программирование			**184**	
Раздел 1 Объектно-ориентированное программирование			**128**	
Тема 1.1. Разработка программы сложной структуры	**Содержание**		6	
	1	Изучение способов разработки ПП. Структурирование составленной программы		2
	2	Оптимизация программного кода.		2
	3	Отладка и тестирование готовой программы.		2
	Практические занятия		4	
	1	Составление программы решения поставленной задачи.		
	2	Проведение анализа готовой программы		
	Лабораторные работы		12	
	1	Написание программы с использованием подпрограмм.		
	2	Написание подпрограммы для решения поставленной задачи.		
	3	Объединение подпрограмм в единый программный код.		
	4	Разработка программного кода сложной структуры		
	5	Выполнение оптимизации готовой программы		
	6	Выполнение отладки и тестирования готовой программы (ОК 6.2.1)		

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2		3	4
Тема 1.2. Файлы.	**Содержание**		4	
	1	Этапы разработки программы с обращением к файлам. Организация работы с файлами различных типов.		2
	2	Интеграция подпрограмм и файлов при решении поставленной задачи.		3
	Практические занятия		4	
	1	Составление структуры программы с обращением к файлу.		
	2	Составление теста для поиска ошибок в программе.		
	Лабораторные работы		12	
	1	Программирование с обращением к типизированному файлу.		
	2	Программирование с обращением к текстовому файлу.		
	3	Разработка программы с использованием подпрограмм.		
	4	Разработка программы с хранением данных в файле.		
	5	Выполнение отладки и тестирования программы.		
	6	Выполнение тестирования программы.		
Тема 1.3. Модули	**Содержание**		4	
	1	Применение модулей. Структура модуля. Компиляция модуля.		2
	2	Написание программного кода модуля. Использование готовых модулей при решении практических задач.		2
	Практические занятия		2	
	1	Написание структуры модуля для решения поставленной задачи.		
	Лабораторные работы		12	
	1	Составление программного кода модуля.		
	2	Создание основной программы .		
	3	Написание программы как отдельного модуля		
	4	Написание программы с использованием готовых модулей.		
	5	Выполнение отладки программного модуля		
	6	Выполнение тестирования программного модуля.		
Тема 1.4. Среда визуального программирования	**Содержание**		10	
	1.	Основные принципы технологии ООП. Интерфейс среды программирования. Основные приемы и команды работы в среде программирования.		2
	2.	Типы данных и операции работы с ними. Реализация основных алгоритмических конструкций.		2
	3	Работа с массивами в среде визуального программирования.		2
	4	Работа с подпрограммами в среде визуального программирования.		2

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2		3	4
	5	Работа с файлами в среде визуального программирования(ОК 2.2.1)		2
	Практические занятия		12	
	1.	Реализация алгоритма поставленной задачи в среде визуального проектирования.		
	2.	Решение задач на реализацию различных структур алгоритмов		
	3.	Решение задач на сортировку массивов.		
	4.	Написание программного модуля с использованием массивов.		
	5.	Написание программного модуля с использованием процедур.		
	6.	Составление программного модуля с использованием файлов (ОК 6.1.1)		
	Лабораторные работы		16	
	1.	Написание простейшего программного кода модуля в среде визуального программирования.		
	2.	Написание программы с использованием двумерных массивов.		
	3.	Составление программы по работе с записями и файлами.		
	4.	Написание программы с использованием готовых модулей.		
	5.	Формирование программного модуля для решения прикладной задачи. (ОК 3.1.1)		
	6.	Использование графических возможностей среды программирования.		
	7.	Выполнение отладки программного модуля.		
	8.	Выполнение тестирования программного модуля. (ОК 8.1.2)		
Раздел 2. Разработка Web-приложений				
Тема 2.1. Серверы приложений	**Содержание**		4	
	1	*Серверы приложений: типы, назначения, функции.*		*1*
	2	*Web сервер Apache его функции и предъявляемые к нему требования. Алгоритм установки сервера*		
Тема 2.2. Web-программирование и Web-дизайн	**Содержание**		6	
	1	*Основные понятия и термины Web - программирования.*		*1*
	2	*Web – дизайн. Требования, предъявляемые к внешнему виду страницы и эргономике.*		
	3	*Основные правила макетирования Web-страницы*		
Тема 2.3. Язык гипертек-	**Содержание**		18	
	1	*Гипертекстовая разметка текста. Языки гипертекстовой разметки*		*1*

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2		3	4
стовой размет-ки HTML		*(Принципы гипертекстовой разметки. Структура гипертекстовых документов. Идентификаторы UDI. Коды языков. Этапы создания HTML-документа.)*		
	2	*Основные теги языка HTML и их свойства (Определение тегов и их свойств. Служебные теги, теги форматирования текста и таблиц, физические стили текста)*		*1*
	3	*Представление данных в виде списков (Нумерованные, маркированные списки, списки с графическими маркерами). Теги включения ссылок (Текстовые, графические ссылки. Организация перехода в пределах одной страницы с помощью якорей и организация перехода между страницами.)*		*2*
	5	*Каскадные таблицы стилей.(Способы определения стилей. Элементы стилей. Синтаксис стилей.)*		*2*
	6	*Теги включения изображений, мультимедийных объектов.(Форматы графических файлов, используемых на Web-страницах. Оптимизация изображения для Web. Бегущая строка. Технология ImageMap. Примеры ее использования)*		*2*
	7	*Организация данных табличного вида на Web-странице(Фиксированные и динамические таблицы и их элементы.)*		*2*
	8	*Фреймы.(Определение фрейма. Основные правила создания HTML-документа с фреймовой структурой. Организация вывода данных в указанный фрейм: статические и динамические фреймы.)*		*2*
	9	*Формы и ее элементы*		*2*
	Лабораторные работы		18	
	1	*Форматирование текста на Web-странице, применение различного цветового решения к фону страницы*		
	2	*Создание текстового и графического меню*		
	3	*Использование технологии ImageMap и CSS.*		
	4	*Создание стилей и их применение к оформлению страниц*		
	5	*Создание страниц с статическими и динамическими фреймами*		
	6	*Создание страницы с фреймовой структурой*		
	7	*Организация взаимодействия между страницами различными способами*		
	8	*Использование обработчиков событий*		
	9	*Создание макета и разработка Web-сайта по заданной тематике*		
Тема 2.4.	**Содержание**		12	

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2		3	4
Клиентская часть Web-приложения	*1*	*Скрипты, выполняющиеся на стороне клиента. Способы интеграции скриптов с Web-страницей*		*1*
	2	*Объекты и классы JavaScript.*		
	3	*Типы переменных и операции над ними в JavaScript.*		
	4	*Основные алгоритмические конструкции JavaScript. Массивы в JavaScript*		
	5	*Обработка элементов форм средствами JavaScript*		
	6	*Анимационные эффекты в JavaScript*		
	Лабораторные работы		14	
	1	*Создание скрипта диалога*		
	2	*Создание скрипта с разветвляющейся алгоритмической конструкцией*		
	3	*Создание скрипта с циклической алгоритмической конструкцией **«for»***		
	4	*Создание скрипта с циклической алгоритмической конструкцией **«пока»***		
	5	*Создание скрипта – обработчика элементов форм Web-страницы*		
	6	*Назначение пароля на доступ к странице, перехват «секретных данных»*		
	7	*Написание скрипта для создания Web-страницы с анимационными эффектами*		
Самостоятельная работа при изучении разделов 1-2 ПМ 01. Составление опорного конспекта по темам: «Основные принципы отладки и тестирования», «Виды тестирования программ». (ОК 4.2.2) Проведение анализа готовой программы на определение основных структурных блоков программного кода. Разработка тестирующего задания для предложенной готовой программы. Составление алгоритма решения задачи и написания программного кода полученного алгоритма на языке визуального программирования. Выполнение упражнений на закрепление материала. *Аналитический обзор литературы по HTML** *Реферирование темы «Современные Web-редакторы (ОК 5)»** *Составление опорного конспекта «Основные приемы работы в Web-редакторе»** *Создание Web-страниц с помощью Web-редакторов** *Создание макета сайта по индивидуальной тематике** *Реализация макета сайта с помощью языка гипертекстовой разметки HTML** *Внедрение основных web-технологий в проект;* *Написание сценариев:** *- для организации диалога с посетителем страницы;**			**92** ***(36*)***	

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2	3	4
*- для решения математической задачи на вычисление значений функций, на обработку массивов;** *- для обработку введенных данных посетителя страницы;** *- для создания организацию динамических страниц;** *- для организации защиты данных web-страницы**			
	Примерная тематика курсовых проектов Написание программного кода модуля с ассемблерными вставками по индивидуальному заданию 1. Написание фрагмента программного кода игры «Крестики нолики» 2. Написание фрагмента программного кода игры «Тетрис» 3. Написание программного кода перевода чисел из одной системы счисления в другую (системы указываются пользователем) 4. Создание интерфейса пользователя с помощью графических возможностей системы программирования 5. Написание программного модуля для работы с комплексными числами 6. Составление программного модуля для приближенного вычисления определенного интеграла (выбор метода предоставляется студенту) 7. Создание модуля, реализующего графическое меню 8. Программная реализация симплекс- метода решения задач линейного программирования на уровне модуля 9. Написание модуля вычисления обратной матрицы второго порядка 10. Написание модуля для решения системы с 2 неизвестными методом Крамера 11. Разработка модуля, предназначенного для демонстрации работы алгоритма сортировки Флойда 12. Разработать модуль для демонстрации физического опыта «математический маятник». 13. Разработать модуль, реализующий функцию – генератор простых чисел, воспользовавшись первой формулой Вилланса 14. Разработать функцию – генератор простых чисел, воспользовавшись формулой Вормелла 15. Разработать модуль для демонстрации видов графиков основных алгебраических функций	**30**	
	Самостоятельная работа при выполнении курсового проекта Систематизация теоретического материала по техническому заданию курсового проекта Разработка фрагментов основного кода программного модуля Разработка ассемблерных вставок Оформление разделов курсового проекта Получение рецензии на проект от руководителя Составление текста выступления на защиту курсового проекта	**15**	

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1	2	3	4
Представление курсового проекта на защите			
Учебная практика «Разработка программного модуля в среде визуального программирования» **Виды работ** Составление алгоритма решения практической задачи на уровне модуля Написание программного модуля в среде визуального программирования по индивидуальному заданию. Проведение тестирования программного модуля. Корректировка кода программного модуля по результатам тестирования Демонстрация работы готового программного модуля		**36**	
Производственная практика «Разработка функционального программного модуля» **Виды работ** Определение проблемной области профессиональной задачи Согласование ожидаемых результатов функциональных возможностей модуля с руководителем практики от предприятия Составление алгоритмической модели поставленной задачи на уровне модуля Проведение анализа алгоритмической структуры модуля Разработка кода программы по составленному алгоритму; Разработка компонентов проектной и технической документации с использованием графических языков спецификаций Тестирование программы на выявление программных ошибок Отладка программного продукта с использованием инструментальных средств Внедрение программного продукта в производственный процесс Проведение анализа внедрения программного модуля в существующее программное обеспечение на производстве Представление рекомендаций для улучшения работы программы по результатам проведенного анализа		**72**	
	Всего	**651**	

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

4. УСЛОВИЯ РЕАЛИЗАЦИИ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ

4.1 Требования к минимальному материально-техническому обеспечению

Реализация программы модуля предполагает:

- наличие кабинета стандартизации и сертификации;
- наличие лабораторий: системного и прикладного программирования, управления проектной деятельностью;
- наличие полигонов: вычислительной техники, учебных баз практики.

Оборудование учебного кабинета и рабочих мест кабинета:

- рабочие места с персональным компьютером по количеству обучающихся;
- рабочее место преподавателя;
- комплект плакатов по МДК;
- комплект учебно-методической документации;
- макеты и наглядные пособия по МДК

Технические средства обучения:

- лицензионное программное обеспечение;
- выход в глобальную сеть Internet на каждом ПК;
- точки электропитания;
- сетевое оборудование, обеспечивающее работу локальной сети;
- мультимедийное оборудование;
- источники бесперебойного питания;
- интерактивная доска;

Реализация программы модуля предполагает учебную и производственную практики. Производственная практика проводится в организациях, направление деятельности которых соответствует профилю подготовки обучающихся.

Оборудование и технологическое оснащение рабочих мест: компьютеры, объединенные в локальную сеть с необходимым для решения профессиональной задачи программным обеспечением, оргтехника, выход в Internet.

4.2. Информационное обеспечение обучения

Перечень рекомендуемых учебных изданий, Интернет-ресурсов, дополнительной литературы

Основные источники:

1. Рудаков А. В. Технология разработки программных продуктов.- М.: Издательский центр «Академия», 2006. - 208 с
2. http://progbook.ru/assembler/756-abel-assembler-yazyk-i-programmirovanie-dlya-ibm.html.
3. Абель, П. Ассемблер IBM PC для программирования. 1988 г. – 736 с.
4. Канцедал С.А. Алгоритмизация и программирование: учебное пособие. – М.: ИД «Форум»: ИНФРА-М, 2008 – 352 с.
5. Программирование на языке Паскаль: задачник/ под редакцией Усковой О.Ф. – СПб.: Питер, 2005. – 336 с.
6. Установочный диск с развернутой системой помощи языка программирования Visual Basic, 2010
7. Культин, Н.Б. Turbo Pascal в задачах и примерах: учебное пособие. – БХВ., 2007 – 256 с.
8. Павловская, Т.А. Паскаль. Программирование на языке высокого уровня. - СПб: Питер, 2007 – 393 с.
9. Антонова Г.М., Байков А.Ю. Современные средства ЭВМ и телекоммуникаций: учеб. пособие: Допущено НМС по информатике, 2011. - 144 с., обл.
10. Пескова С.А., Кузин А.В., В о л к о в А.Н. Сети и телекоммуникации: учеб. пособие: Допущено УМО. - 4-е изд., стер., 2011. - 352 с.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Дополнительные источники:

1. Рудольф Марек Ассемблер на примерах.Базовый курс – СПб: Наука и техника, 2005. – 240с.
2. В.В. Фаронов Турбо Паскаль. Начальный курс. Учебное пособие-М.: «Нолидж», 1996. – 613 с.
3. С.А. Абрамов и др. Задачи по программированию. — М.: Наука, 1988. – 210 с.

Интернет-ресурсы

1. Технология разработки программных продуктов: http://chemisk.narod.ru/html/trpp01.html
2. Введение в технологию разработки программных продуктов: http://www.intuit.ru/department/se/introprogteach/1/
3. Изучение HTML 3.2 на примерах: http: // wmhelp.net / books / category / htm.html
4. Введение в javascript: http://citforum.ru/internet/koch/tutorial.htm

4.3. Общие требования к организации образовательного процесса

Перед изучением модуля обучающиеся изучают следующие дисциплины «Теория алгоритмов», «Операционные системы», «Основы программирования», «Элементы высшей математики».

До начала производственной практики изучаются дисциплины «Правовое обеспечение профессиональной деятельности» и «Безопасность жизнедеятельности».

4.4. Кадровое обеспечение образовательного процесса

Требования к квалификации педагогических кадров, обеспечивающих обучение по междисциплинарному курсу, руководство практикой:

- наличие высшего образования, соответствующего профилю модуля.

Инженерно–педагогический состав:

- преподаватели с высшим образованием, соответствующим профилю модуля, имеющие опыт деятельности в профильных организациях;
- обязательное прохождение стажировки в профильных организациях не реже 1 раза в 3 года

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

5. КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ (ВИДА ПРОФЕССИОНАЛЬНОЙ ДЕЯТЕЛЬНОСТИ)

Результаты (освоенные профессиональные компетенции)	**Основные показатели результатов подготовки**	**Формы и методы контроля и оценки результатов обучения**
ПК 1.1. Разработка спецификаций отдельных компонент.	Разработка спецификаций отдельных компонент выполнена в соответствии с требованиями: 1) составлена на основе ЕСПД (ГОСТ 19.XXX-XX, ГОСТ 34.XXX-XX) 2) отражает заданную функциональность и качество компонента	Оценка продукта учебной деятельности (спецификация) по критериям на квалификационном экзамене по модулю
ПК 1.2 Разработка кода программного продукта на основе готовых спецификаций	Метод разработки программного продукта выбран в соответствии с требованиями спецификации Разработка программного кода программного продукта ведется в соответствии с выбранным методом	Оценка продукта учебной деятельности (код программного продукта) по критериям (соответствие выданным спецификациям) на квалификационном экзамене по модулю
ПК 1.3. Выполнять отладку программных модулей с использованием специализированных программных средств	Тип выявленной ошибки программного модуля определен в соответствии с имеющимся перечнем ошибок. Выбранный метод отладки соответствует типу ошибки Отладка программного модуля, созданного на языке машинных кодов, выполняется с использованием специализированных средств в соответствии с условиями задания и выбранным методом Отладка программного модуля, созданного на языке программирования, выполняется с использованием специализированных средств в соответствии с условиями задания выбранным методом	Оценка продукта учебной деятельности (исправленный код программного продукта) по критериям (соответствие условиям задания) на квалификационном экзамене по модулю
ПК 1.4. Выполнять тестирование программных модулей	Метод тестирования выбран в соответствии с условиями поставленной задачи и обоснован	Оценка продукта учебной деятельности (выбранного метода) по критериям (соответствие условиям задания, приведены обоснования) на квалификационном экзамене по модулю
	Тестирование модуля, созданного на языке машинных кодов, осуществляется в соответствии с выбранным методом тестирования Тестирование модуля, созданного на языке программирования, осуществляется в соответствии с выбранным методом тестирова-	Оценка процесса учебной деятельности (выполнение тестирования) по критериям (соответствие методу тестирования) на учебной практике

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

	ния	
	Вывод о соответствии программного модуля условиям задачи сделан на основе результатов тестирования	Оценка продукта учебной деятельности (вывод) по критериям (соответствие результатам тестирования) на производственной практике
ПК 1.5. Осуществлять оптимизацию программного кода модуля	Анализ по оптимизации программного кода модуля проведен в соответствии с условиями задания. Участки программного кода модуля, требующие оптимизации, выявлены в соответствии с результатами проведенного анализа. Оптимизация программного кода модуля, выполнена в соответствии с предложенной методикой оптимизации	Оценка продукта учебной деятельности (оптимизированный код программного модуля) по критериям на квалификационном экзамене по модулю
ПК 1.6. Разрабатывать компоненты проектной и технической документации с использованием графических языков спецификаций	Разработка компонента проектной и технической документации выполнена в соответствии с требованиями: 1) использованы графические язык спецификаций 2) составлена на основе ЕСПД (ГОСТ 19.XXX-XX, ГОСТ 34.XXX-XX) 3) отражает заданную функциональность и качество	Оценка продукта учебной деятельности (компоненты проектной и технической документации) по критериям на производственной практике

Формы и методы контроля и оценки результатов обучения должны позволять проверять у обучающихся не только сформированность профессиональных компетенций, но и развитие общих компетенций и обеспечивающих их умений.

Результаты обучения (освоенные общие компетенции)		**Основные показатели оценки результата**	**Формы и методы контроля и оценки результатов обучения**
Код	**Формулировка**		
ОК 1	Понимать сущность и социальную значимость своей профессии, проявлять к ней устойчивый интерес	Приведены произвольные примеры социальной значимости своей профессии Сформулированы сущностные характеристики профессии «программист-техник»	Оценка результатов стандартизованного тестирования на итоговом испытании (экзамене/диф.зачете по УД или МДК, сертификационном экзамене по модулю)
ОК 2	Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.	Поставленная цель деятельности соответствует условиям профессиональной задачи Набор ресурсов определен в соответствии с поставленной целью Разметка времени выполнения профессиональной задачи проведена в соответствии с поставленной целью и имеющимися ресурсами	Оценка продукта деятельности обучающегося (решение практико-ориентированного задания) по критериям (соответствие условиям задачи, типовому алгоритму решения, методики оценивания) на итоговом испытании (экзамене/диф.зачете по УД или МДК, сертифи-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

		Выбранный метод и способ решения профессиональной задачи соответствует типовому (известному) алгоритму решения и проведенной разметки времени Оценка эффективности и качества метода и способа решения задачи соответствует заданной методике оценивания	кационном экзамене по модулю)
ОК 3	Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.	Анализ стандартной/нестандартной ситуации проведен в соответствии с инструкцией проведения анализа Проблемная область заданной ситуации установлена в соответствии с результатами проведенного анализа Принятое решение соответствует установленной проблемной области заданной ситуации Несет ответственность за принятое решение в соответствии с заданными полномочиями	Оценка результатов формализованного наблюдения за деятельностью обучающегося по критериям (решение принято в соответствии с условиями ситуации, соответствие проблемной области и заданным полномочиям) на учебной/производственной практике
ОК 4	Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.	Обращается к информационным системам, базам и банкам компьютерных данных по интересующему вопросу в ходе решения поставленной профессиональной задачи Общается со специалистами по интересующему вопросу для личностного развития	Оценка результатов формализованного наблюдения за учебной / профессиональной деятельностью обучающегося на учебной/производственной практике
ОК 5	Использовать информационно - коммуникационные технологии в профессиональной деятельности.	Использование информационно - коммуникационных технологий в профессиональной деятельности соответствует условиям поставленной профессиональной задачи и характеристикам имеющегося аппаратного обеспечения.	Оценка продукта учебной/профессиональной деятельности обучающегося по критериям на учебной/производственной практике
ОК 6	Работать в коллективе и в команде, эффективно общаться с коллегами, руководством, потребителями.	Обсуждение технического задания с потребителем проведено в соответствии с пунктами «Опросного листа клиента»	Оценка результатов формализованного наблюдения за учебной/профессиональной деятельностью обучающегося на учебной/производственной практике
		План выполнения работ составлен в соответствии с техническим заданием и одобрен руко-	Оценка продукта деятельности обучающегося (план работ) по критериям (со-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

		водством.	ответствие тех. заданию) на учебной/производственной практике
		Распределение работ среди членов команды выполнено в соответствии с планом выполнения работ. Техническое задание выполнено согласно выработанному плану работ	Оценка продукта деятельности обучающегося (техническое задание) по критериям (соответствие пунктам плана) на учебной/производственной практике
ОК 7	Брать на себя ответственность за работу членов команды (подчиненных), за результат выполнения заданий.	Составленный план-график работы каждого члена команды соответствует цели задания Результат выполнения задания каждого члена команды скорректирован в соответствии с планом-графиком Берет ответственность за результат выполнения задания в соответствии с заданными полномочиями	Экспертная оценка продукта учебной деятельности (коллективного проекта) по критериям на его защите
ОК 8	Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.	Задачи профессионального развития определены в соответствии с условиями поставленной профессиональной (производственной задачи) План личностного развития и самообразования составлены в соответствии с имеющейся на текущий момент самооценкой Запланированное повышение квалификации соответствует запросам потенциальных работодателей, предъявляемым к специалисту «техник-программист»	Оценка результатов формализованного наблюдения за учебной/профессиональной деятельностью обучающегося на учебной/производственной практике
ОК 9	Ориентироваться в условиях частой смены технологий в профессиональной деятельности.	Перечисляет современные программные и аппаратные ресурсы, соответствующие условиям поставленной профессиональной задачи Формулирует основные характеристики имеющихся программных и аппаратных ресурсов Осваивает предложенную технологию по имеющейся сопроводительной документации	Оценка результатов стандартизованного тестирования на итоговом испытании (экзамене/диф.зачете по УД или МДК, сертификационном экзамене по модулю)
ОК 10	Исполнять воинскую обязанность, в том числе с применением полученных профессиональных знаний (для юношей).	Приводит произвольные примеры применения полученных знаний при исполнении воинской обязанности	Экспертная оценка результатов стандартизованного тестирования на итоговом испытании

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

РАЗДЕЛ 2 КОНСПЕКТЫ ТЕОРЕТИЧЕСКИХ ЗАНЯТИЙ

Технологии разработки ПП

Технологии (с греческого: ремесло + наука) - совокупность знаний о способах и средствах проведении производственных процессов.

В одном крайнем случае один человек осуществляет поэтапную разработку программы со своего терминала в непринужденной обстановке. Естественно, он создает сравнительно небольшую программу, не требующую особой оценки.

В другом обычном случае разрабатывается очень сложное программное обеспечение, предназначенное для функционирования в реальном масштабе времени и требующее трудозатрат объемам в тысячи человеко-часов.

Эти две взаимно-противоположные ситуации характеризуются различной степенью формализации и проведении процесса разработки программных средств.

Степень формализованности и проведения процесса разработки программного обеспечения напрямую зависит от целей его создания, его величины, численности группы разработчиков и других факторов. От того, насколько правильно и удачно с точки зрения технологии разработки программного обеспечения построено приложение, зависит качество и жизнеспособность конечного продукта.

Под *технологией разработки программного обеспечения* (ТРПО) понимается совокупность обобщенных и систематизированных знаний, или наука об оптимальных способах (приемах) проведения процесса разработки программного обеспечения, обеспечивающего в заданных условиях получение программной продукции с заданными свойствами.

Технология разработки программного обеспечения представляет собой инженерный подход к разработке программных средств ЭВМ, охватывающий методологию программирования, проблемы обеспечения надежности программ, оценки рабочих характеристик и качества проектов.

Технология разработки программного обеспечения рассматривает вопросы управления проектированием систем программного обеспечения, а также средства и стандарты разработки программ.

Технология разработки программного обеспечения определяет некоторую профессиональную культуру работы специалистов (не только программистов), обеспечивающую заданный уровень производительности труда и качества получаемой в результате программной продукции.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Технология разработки программного обеспечения охватывает процесс разработки программного обеспечения от появления потребности в нем до его изготовления, передачи пользователю, модификации в процессе эксплуатации и прекращения его использования вследствие морального старения.

В идеале технология разработки программного обеспечения должна удовлетворять основным ниже перечисленным требованиям.

1. Необходима стандартизация языков проектирования программ, оформления и испытания программных модулей, а также гарантии их качества. Это позволит значительно сократить дублирующие разработки, внедрить сборочное программирование и вести накопление на предприятиях и в стране высококачественного программного продукта для его многократного использования в качестве типовых комплектующих изделий.

2. Вести постоянный контроль и обеспечение качества программ.

3. Программы не должны содержать непроверенных путей и ситуаций функционирования, которые приводят к неожиданным результатам.

4. Пользователю или покупателю программ необходимо дать четкое представление о возможностях данной программы и технологических условиях эксплуатации, при которых гарантируются определенные функции и качества.

5. Технология разработки программного обеспечения должна обеспечивать отторжимость программного изделия от его разработчика, т.е. человеческий фактор в программировании быть сведен к минимуму.

6. Технология разработки программного обеспечения и средства ее поддержки (автоматизации) должны обеспечивать целенаправленную работу, прежде всего коллектива программистов, а не отдельных личностей. Она должна побуждать коллектив работать только правильно и слаженно; должна автоматически блокировать любые не санкционированные технологией действия.

7. Необходимо вести аккуратное документирование всех этапов разработки. Документация должна также заноситься и храниться на магнитных носителях. Доступ к этой информации должен быть открытым, простым и автоматизированным.

8. Работа пользователя должна обеспечиваться развитой информационно-справочной системой.

9. Средства автоматизации технологии должны охватывать все этапы работы коллектива программистов.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

10. Технология разработки программного обеспечения должна быть простой в освоении, с автоматически включаемыми средствами подсказки.

11. Технология разработки программного обеспечения должна иметь средства автоматической фиксации в хронологическом порядке всех действий, выполняемых в процессе коллективного изготовления программного изделия - должны вестись и храниться в системе журналы (протоколы, дневники) разработки. Эти средства должны позволять восстанавливать любое состояние процесса разработки на любом интервале изготовления программного эксплуатации.

Объектно-ориентированная разработка программ

Разработанный программный продукт решает задачи реального мира, предметы и понятия которого при объектно-ориентированной разработке программ заменяются их моделями, т.е. определенными формальными конструкциями, представляющими их в программной системе.

Модель содержит не все признаки и свойства представляемого ею предмета (понятия), а только те, которые существенны для разрабатываемой программной системы. Это значит, что модель «беднее», а, следовательно, проще отображаемого ею предмета (понятия). Но главное даже не в этом, а в том, что модель есть формальная конструкция. Формальный характер моделей позволяет определить формальные зависимости и формальные операции над ними, что значительно упрощает разработку программного продукта.

Объектно-ориентированная разработка программ помогает справиться с такими сложными проблемами, как:

— уменьшение сложности программного обеспечения;

— повышение надежности программного обеспечения;

— обеспечение возможности модификации отдельных компонентов программного обеспечения без изменения остальных его компонентов;

— обеспечение возможности повторного использования отдельных компонентов программного обеспечения.

Систематическое применение объектно-ориентированного подхода к разработке программ позволяет разрабатывать хорошо структурированные, надежные в эксплуатации, достаточно просто модифицируемые программные системы. В настоящее время объектно-ориентированная разработка программ является одним из наиболее интенсивно развивающихся направлений в теоретическом и прикладном программировании.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Объектно-ориентированная разработка программного обеспечения использует инструментальные средства, поддерживающие объектно-ориентированные методологии (технологии) и связана с применением объектно-ориентированных моделей при разработке программных систем и их компонентов.

Можно назвать разные объектно-ориентированные методологии, например, SA/SD (Structured Analysis/Structured Design), JSD (Jackson Structured Development), OSA (Object-Oriented System Analysis).

Одной из наиболее продвинутых и популярных объектно-ориентированных методологий является OMT (Object Modeling Technique). Ее графический язык (система обозначений для диаграмм) получил достаточно широкое распространение и используется в некоторых других объектно-ориентированных методологиях, а также в большинстве публикаций по объектно-ориентированным методологиям.

Контрольные вопросы.

1. Что включает в себя понятие технологии разработки программного обеспечения.
2. Каким требованиям должна удовлетворять технология разработки ПП?
3. в чем заключается объектно-ориентированная разработка программ?

ОТЛАДКА И ТЕСТИРОВАНИЕ

Процесс отладки включает:

- действия, направленные на выявление ошибок (тестирование);
- диагностику и локализацию ошибок (определение характера ошибок и их местонахождение);
- внесение исправлений в программу с целью устранения ошибок.

Определение и принципы тестирования

Тестирование программного средства (ПС) - это процесс выполнения программ на некотором наборе данных, для которого заранее известен результат применения или известны правила поведения этих программ. Указанный набор данных называется *тестовым* или просто *тестом*. Тестирование программ является одной из составных частей более общего понятия - «отладка программ». Под отладкой понимается процесс, позволяющий получить программу, функционирующую с требующимися характеристиками в заданной области изменения входных данных.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Из перечисленных видов работ самым трудоемким и дорогим является тестирование, затраты на которое приближаются к 45 % общих затрат на разработку ПС.

Невозможно гарантировать отсутствие ошибок в программе. В лучшем случае можно попытаться показать наличие ошибок. Если программа правильно ведет себя для большого набора тестов, нет оснований утверждать, что в ней нет ошибок. Если считать, что набор тестов способен с большой вероятностью обнаружить возможные ошибки, то можно говорить о некотором уровне уверенности (надежности) в правильности работы программы, устанавливаемом этими тестами. Сформулируем следующее высказывание: *если ваша цель - показать отсутствие ошибок, вы их найдете не слишком много. Если же ваша цель - показать наличие ошибок, вы найдете значительную их часть.*

Надежность невозможно внести в программу в результате тестирования, она определяется правильностью этапов проектирования. Наилучшее решение проблемы надежности - с самого начала не допускать ошибок в программе. Однако вероятность того, что удастся безупречно спроектировать большую программу, мала. *Роль тестирования состоит в том, чтобы определить местонахождение немногочисленных ошибок, оставшихся в хорошо спроектированной программе*. Попытки с помощью тестирования достичь надежности плохо спроектированной программы безнадежны.

Тестирование оказывается довольно необычным процессом (поэтому и считается трудным), так как этот процесс разрушительный. Ведь цель проверяющего (тестовика) - заставить программу сбиться.

Программы, как объекты тестирования, имеют ряд особенностей, которые отличают процесс их тестирования от общепринятого, применяемого при разработке аппаратуры и других технических изделий. Особенностями тестирования ПП являются:

отсутствие эталона (программы), которому должна соответствовать тестируемая программа;

высокая сложность программ и принципиальная невозможность исчерпывающего тестирования;

практическая невозможность создания единой методики тестирования (формализация процесса тестирования) в силу большого разнообразия программных продуктов (ПП) по их сложности, функциональному назначению, области использования и т.д.

Тестирование - это процесс многократного выполнения программы с целью выявления ошибок. Целью тестирования является обнаружение максимального числа ошибок.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Поэтому тестовый прогон, в результате которого не выявлено ошибок, считается неудачным (неэффективным).

Существуют несколько эмпирических правил проведения тестирования программ, обобщающих опыт тестировщиков.

1. Процесс тестирования более эффективен, если проводится не автором программы. По своей сути тестирование - это процесс деструктивный (разрушительный). Именно этим и объясняется, почему многие считают его трудным. Особенно трудным и малоэффективным он является для самого автора программы, так как после выполнения конструктивной части при проектировании и написания программы, ему трудно перестроиться на деструктивный образ мышления и, создав программу, тут же приступить к пристрастному выявлению в ней ошибок. Поэтому для проведения тестирования создаются специальные группы тестирования. Это не означает, что программист не может тестировать свою программу. Речь идет о повышении эффективности тестирования.

2. Необходимой частью тестового набора данных должно быть описание предполагаемых значений результатов тестовых прогонов. Тестирование как процесс многократного выполнения программы проводится на многочисленных входных наборах данных. Чтобы определить правильность полученных в результате очередного тестового прогона данных, необходимо знать ожидаемый результат. Таким образом, тестовый набор данных должен включать в себя два компонента: описание входных данных, описание точного и корректного результата, соответствующего набору входных данных. Этот принцип сложно, а в некоторых случаях и невозможно реализовать на практике. Сложность его заключается в том, что при тестировании программы (модуля) необходимо для каждого входного набора данных рассчитать вручную ожидаемый результат или найти допустимый интервал изменения выходных данных. Процесс этот трудоемкий даже для небольших программ, так как он требует ручных расчетов, следуя логике алгоритма программы. Из рассмотренного принципа, который трудно реализуем, но которого следует придерживаться логически, вытекает следующий.

3. Необходимо изучить результаты каждого теста. Из практики следует, что значительная часть обнаруженных ошибок могла быть выявлена в результате первых тестовых прогонов, но они были пропущены вследствие недостаточно тщательного анализа их результатов.

4. Тесты для неправильных и непредусмотренных входных данных должны разрабатываться также тщательно, как для правильных и предусмотренных. Согласно этому прин-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ципу при обработке данных, выходящих за область допустимых значений, в тестируемой программе должна быть предусмотрена диагностика в виде сообщений. Если сообщение о причине невозможности обработки по предложенному алгоритму отсутствует, и программа завершается аварийно или ведет себя непредсказуемо, то такая программа не может считаться работоспособной и требует существенной доработки. Тестовые наборы данных из области недопустимых входных значений обладают большей обнаруживающей способностью, чем тесты, соответствующие корректным входным данным.

5. Необходимо проверять не только, делает ли программа то, для чего она предназначена, но и не делает ли она того, чего не должна делать. Это утверждение логически вытекает из предыдущего. Необходимо любую программу проверить на нежелательные побочные эффекты.

6. Следует тщательнее проверять те участки программ, где обнаруживается больше ошибок. Утверждается, что вероятность наличия необнаруженных ошибок в какой-либо части программы пропорциональна числу ошибок, уже обнаруженных в этой части. Возможно, что те части программы, где при тестировании обнаружено большее число ошибок, либо были слабо проработаны с точки зрения системного анализа, либо разрабатывались программистами более низкой квалификации.

Основные определения

Тестирование (testing) - процесс выполнения программы или ее части с целью найти ошибки.

Доказательство (proof) - попытка найти ошибки в программе безотносительно к внешней для программы среде. Большинство методов доказательства предполагает формулировку утверждений о поведении программы и доказательство математических теорем о правильности программы. Доказательства могут рассматриваться как форма тестирования, хотя они и не предполагают прямого выполнения программы.

Контроль (verification) - попытка найти ошибки, выполняя программу в тестовой, или моделируемой, среде.

Испытание (validation) - попытка найти ошибки, выполняя программу в заданной реальной среде.

Аттестация (certification) - авторитетное подтверждение правильности программы. При тестировании с целью аттестации выполняется сравнение с некоторым заранее определенным стандартом.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Отладка (debugging) не является разновидностью тестирования. Хотя слова «отладка» и «тестирование» часто используются как синонимы, но под ними подразумеваются разные виды деятельности.

Тестирование - это деятельность, направленная на обнаружение ошибок.

Отладка направлена на установление точной природы известной ошибки, а затем на исправление этой ошибки. Эти два вида деятельности связаны, т.к. результаты тестирования являются исходными данными для отладки.

Тестирование модуля, или автономное тестирование (module testing, unit testing) - контроль отдельного программного модуля, обычно в изолированной среде (изолированно от всех остальных модулей).

Тестирование сопряжений (integration testing) - контроль сопряжений между частями системы (модулями, компонентами, подсистемами).

Тестирование внешних функций (external function testing) - контроль внешнего поведения, определенного внешними спецификациями.

Комплексное тестирование (system testing) - контроль и/или испытание системы по отношению к исходным целям.

Тестирование приемлемости (acceptance testing) - проверка соответствия программы требованиям пользователя.

Контрольные вопросы.

1. Что такое отладка программы и для чего она нужна?
2. Раскрыть процесс отладки программы.
3. Что такое тестирование программы?
4. В чем заключается роль тестирования?
5. Перечислите эмпирические правила проведения тестирования программ.
6. Выучить основные определения.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ОРГАНИЗАЦИЯ РАБОТЫ С ФАЙЛАМИ РАЗЛИЧНЫХ ТИПОВ. ЭТАПЫ РАЗРАБОТКИ ПРОГРАММЫ С ОБРАЩЕНИЕМ К ФАЙЛАМ

Понятие файла

Переменная файлового типа, или коротко файл, в языке Паскаль представляет последовательность однотипных компонент, соответствующих последовательности записей на внешнем носителе. В отличие от массива, количество компонент заранее не оговаривается, и компоненты файла не имеют индексов.

С точки зрения программирования, все файлы можно разделить на три класса:

• типизированные;

• бестиповые;

• текстовые.

Файлы, состоящие из компонентов одного типа (целые, вещественные, массивы и т.д.), число которых заранее не определено и может быть любым, называются типизированными. Они заканчиваются специальным символом «конец файла», хранятся в двоичном виде, содержимое подобных файлов нельзя просмотреть обычным текстовым редактором, для просмотра подобных файлов нужно писать специальную программу.

В бестиповых файлах информация считывается и записывается блоками определенного размера. В подобных файлах хранятся данные любого вида и структуры.

Текстовые файлы состоят из любых символов. При записи информации в текстовый файл все данные преобразуются к символьному типу, в котором и хранятся. Просмотреть данные в подобном файле можно с помощью любого текстового редактора. Информация в текстовом файле хранится построчно. В конце каждой строки хранится специальный символ «конец строки». Конец самого файла обозначается символом «конец файла».

Работу с текстовыми файлами обеспечивает модуль System Турбо Паскаля. Этот модуль содержит процедуры и функции для основных операций ввода и вывода информации, работу со строками и т. д. Модуль System подключается к программе автоматически, если в этом есть необходимость, и не требует использования оператора uses.

Существуют два способа доступа к файлам - последовательный и прямой. Текстовые файлы допускают только последовательный доступ. Это означает, что для нахождения определенного элемента такого файла нужно просмотреть все, что ему предшествовало. При последовательном доступе программа не может в любой момент времени считать из файла произвольную порцию информации или произвести запись в произвольное место

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

файла. Так, например, чтобы прочитать третий элемент текстового файла, нужно сначала прочитать первый и второй элементы. Прямой доступ - при котором выполняется чтение/запись произвольного элемента по заданному адресу. В бестиповых и типизированных файлаз можно использовать оба метода.

При определении переменной файлового типа также в программе появляется скрытый (неявный) *текущий указатель файла.* Его назначение – указывать на конкретный элемент файла (обеспечивать доступ к нему).

В языке программирования Паскаль все действия с файлом (чтение, запись) производятся поэлементно. Действия совершаются именно над тем элементом файла, на который указывает текущий указатель файла. После того как действие будет завершено, указатель перемещается к следующему элементу. Все элементы файла пронумерованы, начиная с нуля.

Этапы разработки программы с обращением к файлам.

Для работы с файлами в программе следует описать файловую переменную. Для работы с текстовым файлом файловая переменная (например, f) описывается с помощью служебного слова text.

```
var f:text;
```

Для описания типизированных файлов можно описать файловую переменную следующим образом:

```
var f:file of тип;
```

Бестиповый файл описывается с помощью служебного слова file.

Рассмотрим несколько примеров описания файловых переменных.

```
type
massiv=array[1..25]of real;
ff=file of real;
var
a:text; {Файловая переменная a для работы с текстовым файлом}
b:ff; {Файловая переменная f для работы с файлом вещественных чисел}
c:file of integer; {Файловая переменная c для работы с файлом целых чисел}
d:file of massiv; {Файловая переменная d предназначена для работы с типизированным файлом, элементами которого являются массивы из 25 вещественных чисел. }
```

С файловой системой TURBO PASCAL связано понятие буфера файла (ввода – вывода). Ввод и вывод данных осуществляется через буфер. Буфер – это область в памя-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ти, которая выделяется для каждого файла. При записи в файл вся информация сначала направляется в буфер и там накапливается до тех пор, пока весь объем буфера не будет заполнен. Только после этого или после специальной команды сброса происходит передача данных на внешнее устройство. При чтении из файла данные вначале считываются в буфер, причем данных считывается не столько, сколько запрашивается, а сколько поместится в буфер. Механизм буферизации позволяет более быстро и эффективно обмениваться информацией с внешними устройствами.

ОПЕРАЦИИ НАД ФАЙЛАМИ

Типизированные файлы

Начинается работа над файлами с процедур открытия файла:

Assign(f, <имя файла>:String);

Reset(f); или Rewrite(f);

Процедура Assign() устанавливает соответствие между файловой переменной f и < именем файла > на внешнем носителе. Процедуры Reset(f) и Rewrite(f) инициируют (подготавливают) файл для работы. При этом, если файл на внешнем носителе отсутствует, то следует использовать оператор Rewrite(f), который создает новый файл с именем <имя файла>. Если требуется работать с уже существующим файлом, то необходимо использовать оператор Reset(f). После выполнения процедур открытия файла указатель всегда установлен в начало файла (на компонент с номером 0). Если открытие производится оператором Rewrite(f), то признак конца файла совмещен с началом (т.е. файл пуст). Если перед этим в файле имелась некоторая информация, то она стирается. Запись значений переменных в файл производится покомпонентно с помощью оператора

Write(f,x);

Write(f,y,z);

После записи каждой переменной значения указатель увеличивается на единицу, что соответствует его перемещению к следующему компоненту файла. Если перед записью указатель находился напротив признака конца файла, то при записи каждой переменной признак конца смещается на длину этой записи и количество компонентов в файле возрастает на единицу.

Количество компонентов, записанных в файле, можно определить с помощью функции FileSize(f). Чтение значений переменных из компонентов файла производится с помощью оператора

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Read(f,x);

Read(f,y,z);

При чтении каждой переменной указатель увеличивается на единицу.

Если производится попытка чтения при указателе, установленном на конец файла, то происходит останов программы по ошибке чтения. Распознать ситуацию, когда указатель установлен на конец файла, можно с помощью функции Eof(f), возвращающей значение True, если достигнут конец файла, и False, в противном случае.

Организовать чтение с проверкой этого условия можно, например, следующим образом:

if not Eof(f) then Read(f,x);

При необходимости чтения или записи заданного компонента файла нужно предварительно указатель установить на номер этого компонента. Это делается с помощью процедуры

Seek(f,<номер компонента>).

Например, при выполнении следующей группы операторов

Seek(f,2);

Read(f,x);

Write(f,y);

Переменная x будет прочитана из компонента с номером 2 (третьего), а переменная y запишется в компонент с номером 3 (четвертый).

Текущее положение указателя можно узнать с помощью функции FilePos(f).

Truncate(f) - Процедура отсекает конец файла, начиная с текущей позиции включительно.

После окончания работы с файлом его следует обязательно закрыть с помощью процедуры

Rename(f, NewName) - позволяет переименовать физический файл на диске, связанный с файловой переменной f. Переименование возможно после закрытия файла.

Erase(f) - уничтожает физический файл на диске, который был связан с файловой переменной f. Файл к моменту вызова процедуры Erase должен быть закрыт.

Close(f).

Если этот оператор отсутствует, то из-за специфики обмена данными с файлом, часть информации, помещенной в файл, может быть утеряна.

Текстовые файлы

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Текстовые файлы – файлы на диске, состоящие из символов ASCII. Текстовые файлы внутренне разделены на строки, длины которых различны. Для разделения строк используются символы EOF (End of File). В любой момент времени доступна только одна запись файла. Другие записи становятся доступными лишь в результате последовательного продвижения по файлу.

Assign (f, <имя файла>:String); – устанавливает связь между именем файла в программе (файловой переменной) и физическим именем файла, принятым в ОС.

Reset (f) - открывает существующий файл для чтения. При работе с текстовым файлом следует помнить, что в отличие от типизированного, после открытия файла процедурой Reset (f) разрешается только чтение

Rewrite (f)– создает и открывает новый файл для записи на внешнем устройстве (если файл ранее существовал, вся предыдущая информация из него стирается).

Close (f) - закрывает открытый файл.

Readln (f , st)- чтение строки st из файла f и переход на начало следующей ;

Writeln (f, st)- запись строки st в файл f и маркера конца строки ;

Append (f) - процедура, открывающая файл f для добавления строк в конец файла. Для сохранения предыдущей информации в текстовом файле его следует открывать процедурой Append (f).

Eoln (st)- логическая функция, результат выполнения которой равен TRUE, если достигнут маркер конца строки st.

Rename(f, NewName) - позволяет переименовать физический файл на диске, связанный с файловой переменной f. Переименование возможно после закрытия файла.

Erase(f) - уничтожает физический файл на диске, который был связан с файловой переменной f. Файл к моменту вызова процедуры Erase должен быть закрыт.

Функция SeekEOLn(var f: Text): Boolean возвращает значение True, если до конца строки остались только пробелы.

Функция SeekEOF(var f: Text): Boolean возвращает значение True, если до конца файла остались строки, заполненные пробелами.

Пример. Создать текстовый файл, в который записать 3 предложения. Прочитать этот файл, вывести его содержимое на экран. Определить длину каждого предложения.

```
Program File_text;
Var   f : text; st : string; n: byte;
begin
```


Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
assign   (f, 'file1.txt'); {связать с файлом file1.txt  файловую переменную f  }
rewrite  (f); { создать новый файл  с именем file1.txt }
writeln ( f, 'Очень полезно изучать'); { записать предложения в файл}
writeln ( f, ' всем  студентам ');
writeln (f, ' язык  Pascal ');
close (f);    { закрыть файл для записи }
reset (f); { открыть файл для чтения }
while not eof (f) do { пока не конец файла  f}
begin
readln (f, st);    {читаем строку из файла f }
writeln(st);         { выводим на экран }
n:= length (st);     {определяем длину строки }
writeln ('  длина =',n);
end;
close (f); { закрыть файл  для чтения}
end .
```

Контрольные вопросы.

1. Что представляет собой файл.
2. Какие виды файлов можно обрабатывать средствами языка Паскаль?
3. Как осуществляется доступ к элементам текстового файла и типизированного файла?
4. какие возможны операции с файлами в языке Паскаль?
5. Приведите отличия текстового файла от типизированного при обработке.
6. Какие действия с файлами различных типов общие?

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

МОДУЛИ В ПАСКАЛЬ. НАЗНАЧЕНИЕ, СТРУКТУРА, ОРГАНИЗАЦИЯ, ПРИМЕР ИСПОЛЬЗОВАНИЯ

Стандартные модули Паскаля

В Турбо Паскале имеется 8 стандартных модулей, в которых содержится множество различных типов, констант, процедур и функций. Этими модулями являются SYSTEM, DOS, CRT, GRAPH, OVERLAY, TURBO. Модули Паскаля GRAPH , TURBO, GRAPH выделены в отдельные TPU -файлы, а остальные входят в состав библиотечного файла TURBO . TPL . Лишь один модуль Паскаля SYSTEM подключается к любой программе автоматически, все остальные становятся доступны только после указания их имен в списке подключаемых модулей.

Модуль Паскаля SYSTEM. В него входят все процедуры и функции стандартного Паскаля, а также встроенные процедуры и функции, которые не вошли в другие стандартные модули (например, INC , DEC , GETDIR и т.п.). Модуль Паскаля SYSTEM подключается к любой программе независимо от того, объявлен ли он в предложении USES или нет, поэтому его глобальные константы, переменные, процедуры и функции считаются встроенными в Турбо Паскаль.

Модуль Паскаля CRT. В нем сосредоточены процедуры и функции, обеспечивающие управление текстовым режимом работы экрана. С его помощью можно перемещать курсор в любую точку экрана, менять цвет выводимых символов и фона, создавать окна. Кроме того, в данный модуль включены также процедуры «слепого» чтения клавиатуры и управления звуком.

Модуль Паскаля GRAPH . Содержит набор типов, констант, процедур и функций для управления графическим режимом работы экрана. Этот модуль позволяет создавать различные графические изображения и выводить на экран надписи стандартными или созданными программистом шрифтами.

Модуль Паскаля DOS . В модуле собраны процедуры и функции, открывающие доступ к средствам дисковой операционной системы MS - DOS .

Модуль Паскаля OVERLAY . Данный модуль необходим при разработке громоздких программ с перекрытиями. Турбо Паскаль обеспечивает создание программ, длина которых ограничивается лишь основной оперативной памятью. Операционная система MS - DOS оставляет программе около 580 Кбайт основной памяти. Память такого размера достаточна для большинства исполняемых программ, тем не менее, использование программ с перекрытиями снимает это ограничение.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Модуль CRT в Turbo Pascal.

Библиотека CRT позволяет менять цвета и яркость экрана, производить его очистку, управлять звуковым генератором компьютера.

Подключение модуля CRT производится директивой USES CRT, которая указывается в разделе описаний. CRT - аббревиатура, означающая "Электронно-лучевая трубка".

Таблица *настройки цветов в модуле CRT Turbo Pascal*

Константа	Число	Цвет	Процедуры
Black	0	Черный	TextColor, TextBackGround
Blue	1	Синий	TextColor, TextBackGround
Green	2	Зеленый	TextColor, TextBackGround
Cyan	3	Голубой	TextColor, TextBackGround
Red	4	Красный	TextColor, TextBackGround
Magenta	5	Фиолетовый	TextColor, TextBackGround
Brown	6	Коричневый	TextColor, TextBackGround
LightGray	7	Ярко-серый	TextColor, TextBackGround
DarkGray	8	Темно-серый	TextColor
LightBlue	9	Ярко-синий	TextColor
LightGreen	10	Ярко-зеленый	TextColor
LightCyan	11	Ярко-голубой	TextColor
LightRed	12	Ярко-красный	TextColor
LightMagenta	13	Ярко-фиолетовый	TextColor
Yellow	14	Желтый	TextColor
White	15	Белый	TextColor
Blink	128	Мерцание	TextColor (как слогаемое)

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Основные директивы модуля CRT

Uses CRT - подключение модуля CRT.

TextColor - устанавливает цвет символов.

TextBackGround - устанавливает цвет фона (только 8 неярких цветов).

HighVideo - устанавливает включение яркости.

LowVideo - устанавливает выбор низкой яркости.

NormVideo - восстановление того цветового оформления, которое было на момент начала работы программы.

ClrScr - производит очистку экрана.

ReadKey - обеспечивает задержку изображения и выход из программы при нажатии любой клавиши, например Esc, Enter, Пробел.

Sound(1000) - включение звукового сигнала частотой 1000 Гц.

Sound(1500) - включение звукового сигнала частотой 1500 Гц.

Delay(1000) - установка (задержка) звучания на время 1 с (1000 мс).

Delay(2000) - установка (задержка) звучания на время 2 с.

NoSound - отключение звукового сигнала.

Window(X1, Y1, X2, Y2 : Byte) - определяет текстовое окно на экране. X1, Y1 - координаты левого верхнего угла, X2, Y2 - правого нижнего угла.

Пример подключения модуля CRT:

```
USES CRT;
BEGIN
TextColor(14); {Задаёт желтые символы текста}
TextBackGround(5); {Задаёт фиолетовый фон}
HighVideo; {Устанавливает включение яркости}
ClrScr; {Очищает экран}
```

Пример: TextColor(14+128); {Задаёт мерцание символов желтого цвета}

Программирование клавиатуры

Дополнительные возможности управления клавиатурой реализуются двумя функциями: KeyPressed и ReadKey.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Функция KeyPressed.

Возвращает значение типа Boolean, указывающее состояние буфера клавиатуры: False означает, что буфер пуст, а True - что в буфере есть хотя бы один символ, еще не прочитанный программой.

В MS-DOS реализуется так называемый асинхронный буферизованный ввод с клавиатуры. По мере нажатия на клавиши соответствующие коды помещаются в особый буфер, откуда они могут быть затем прочитаны программой. Стандартная длина буфера рассчитана на хранение до 16 кодов символов. Если программа достаточно долго не обращается к клавиатуре, а пользователь нажимает клавиши, буфер может оказаться переполненным. В этот момент раздается звуковой сигнал и «лишние» коды теряются. Чтение из буфера обеспечивается процедурами Read/ReadLn и функцией ReadKey. Замечу, что обращение к функции KeyPressed не задерживает исполнения программы: функция немедленно анализирует буфер и возвращает то или иное значение, не дожидаясь нажатия клавиши.

Функция ReadKey.

Возвращает значение типа Char. При обращении к этой функции анализируется буфер клавиатуры: если в нем есть хотя бы один не прочитанный символ, код этого символа берется из буфера и возвращается в качестве значения функции, в противном случае функция будет ожидать нажатия на любую клавишу. Ввод символа с помощью этой функции не сопровождается эхо-повтором и содержимое экрана не меняется.

Пусть, например, в какой-то точке программы необходимо игнорировать все ранее нажатые клавиши, коды которых еще не прочитаны из буфера, т.е. необходимо очистить буфер. Этого можно достичь следующим способом:

```
Uses CRT;
var
C: Char;
begin
while KeyPressed do
C := ReadKey;
.....
end.
```

При использовании процедуры ReadKey необходимо учесть, что в клавиатурный буфер помещаются так называемые расширенные коды нажатых клавиш. Если нажимается

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

любая алфавитно-цифровая клавиша, расширенный код совпадает с ASCII-кодом соответствующего символа. Например, если нажимается клавиша с латинской буквой «а» (в нижнем регистре), функция ReadKey возвращает значение chr (97), а если «А» (в верхнем регистре) - значение chr (65). При нажатии функциональных клавиш F1...F10, клавиш управления курсором, клавиш Ins, Home, Del, End, PgUp, PgDn в буфер помещается двухбайтная последовательность: сначала символ #0, а затем расширенный код клавиши. Таким образом, значение #0, возвращаемое функцией ReadKey, используется исключительно для того, чтобы указать программе на генерацию расширенного кода. Получив это значение, программа должна еще раз обратиться к функции, чтобы прочитать расширенный код клавиши.

Т.е. код сканирования клавиши. Этот код определяется порядком, в соответствии с которым микропроцессор клавиатуры Intel 8042 периодически опрашивает (сканирует) состояние клавиш.

Следующая простая программа позволит определить расширенный код любой клавиши. Для завершения работы программы нажмите клавишу Esc.

```
Uses CRT;
var
C: Char;
begin
repeat
C := ReadKey;
if C<>#0 then
WriteLn(ord(C))
else
WriteLnCO1 ,ord(ReadKey) :8)
until C=#27 {27 - расширенный код клавиши Esc}
end.
```

Если Вы воспользуетесь этой программой, то обнаружите, что нажатие на некоторые клавиши игнорируется функцией ReadKey. Это прежде всего так называемые сдвиговые клавиши - Shift, Ctrl, Alt. Сдвиговые клавиши в MS-DOS обычно используются для переключения регистров клавиатуры и нажимаются в сочетании с другими клавишами. Именно таким способом, например, различается ввод прописных и строчных букв. Кроме того, функция игнорирует переключающие клавиши Caps Lock, Num. Lock, Scroll Lock, a

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

также «лишние» функциональные клавиши F11 и F12 клавиатуры IBM AT, не имеющие аналога на клавиатуре ранних моделей IBMPC/XT (в этих машинах использовалась 84-клавишная клавиатура, в то время как на IBM AT - 101-клавишная).

В табл. 2 приводятся расширенные коды клавиш, возвращаемые функцией ord(ReadKey). Для режима ввода кириллицы приводятся коды, соответствующие альтернативному варианту кодировки.

Таблица 2 Расширенные коды клавиш

Код		Клавиша или комбинация клавиш	Код		Клавиша или комбинация клавиш
Первый байт	**Второй байт**		**Первый байт**	**Второй байт**	
Алфавитно-цифровые клавиши					
8	-	Backspace (Забой)	9	-	Tab (Табуляция)
13	-	Enter	32	-	Пробел
33	-	!	34	-	"
35	-	#	36	-	$
37	-	%	38	-	&
39	-	'	40	-	(
41	-	)	42	-	*
43	-	+	44	-	,
45	-	-	46	-	.
47	-	/	4S...57	-	0...9
58	-		59	-	;
60	-	<	61	-	=
62	-	>	63	-	?
64	-	@	65...90	-	A...Z
91	-	[	92	-	\
93	-	]	94	-	^
95	-		96	-	'
97...122	-	a...z	123	-	{
124	-	I	125	-	}
126	-	~	128...159	-	А...Я
160... 175	-	а...п	224...239	-	р...я
Управляющие клавиши и их сочетания со сдвиговыми					
0	3	Ctrl-2	0	15	Shift-Tab
0	16...25	Alt-Q...Alt-P (верхний ряд букв)	0	30...38	Alt-A...Alt-L (средний ряд букв)
0	44...50	Alt-Z...Alt-M (нижний	0	59...68	F1...F10

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Код		Клавиша или комбинация клавиш	Код		Клавиша или комбинация клавиш
Первый байт	Второй байт		Первый байт	Второй байт	
		ряд букв)			
0	- 71	Ноте	0	72	Курсор вверх
0	73	PgUp	0	75	Курсор влево
0	77	Курсор вправо	0	79	End
0	80	Курсор вниз	0	81	PgDn
0	82	Ins	0	83	Del
0	84...93	Shift-Fl...Shift-F10	0	94...103	Ctrl-F1... Ctrl-F10
0	104...113	Alt-Fl...Alt-F10	0	114	Ctrl-PrtScr
0	115	Ctrl-курсор влево	0	116	Ctrl-Курсор вправо
0	117	Ctrl-End	0	118	Ctrl-PgDn
0	119	Ctrl-Home	0	120...131	Alt-1. ..Alt-= (верхний ряд клавиш)
0	132	Ctrl-PgUp			

Модули в языке программирования Паскаль

Стандартный Паскаль не предусматривает механизмов раздельной компиляции частей программы с последующей их сборкой перед выполнением. Вполне понятно стремление разработчиков коммерческих компиляторов Паскаля включать в язык средства, повышающие его **модульность**.

Модуль Паскаля – это автономно компилируемая программная единица, включающая в себя различные компоненты раздела описаний (типы, константы, переменные, процедуры и функции) и, возможно, некоторые исполняемые операторы инициирующей части.

Основным принципом **модульного программирования** является принцип «разделяй и властвуй». **Модульное программирование** – это организация программы как совокупности небольших независимых блоков, называемых **модулями**, структура и поведение которых подчиняются определенным правилам.

Использование модульного программирования позволяет упростить тестирование программы и обнаружение ошибок. Аппаратно-зависимые подзадачи могут быть строго отделены от других подзадач, что улучшает мобильность создаваемых программ.

Термин «модуль» в программировании начал использоваться в связи с внедрением модульных принципов при создании программ. В 70-х годах под модулем понимали какую-либо процедуру или функцию, написанную в соответствии с определенными правилами.

Модули представляют собой прекрасный инструмент для разработки библиотек прикладных программ и мощное средство модульного программирования. Важная особен-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ность модулей заключается в том, что компилятор размещает их программный код в отдельном сегменте памяти. Длина сегмента не может превышать 64 Кбайт, однако количество одновременно используемых модулей ограничивается лишь доступной памятью, что позволяет создавать большие программы.

Структура модулей Паскаль

Всякий модуль Паскаля имеет следующую структуру:

Unit <имя_модуля>;

interface <интерфейсная часть>;

implementation < исполняемая часть >;

begin

<инициирующая часть>;

end .

Здесь UNIT – зарезервированное слово (единица); начинает заголовок модуля;

<имя_модуля> - имя модуля (правильный идентификатор);

INTERFACE – зарезервированное слово (интерфейс); начинает интерфейсную часть модуля;

IMPLEMENTATION – зарезервированное слово (выполнение); начинает исполняемую часть модуля;

BEGIN – зарезервированное слово; начинает инициирующую часть модуля; причем конструкция begin <инициирующая часть> необязательна;

END – зарезервированное слово – признак конца модуля.

Таким образом, модуль Паскаля состоит из заголовка и трех составных частей, любая из которых может быть пустой.

Заголовок модуля Паскаля и связь модулей друг с другом

Заголовок модуля Паскаля состоит из зарезервированного слова unit и следующего за ним имени модуля. *Для правильной работы среды Турбо Паскаля и возможности подключения средств, облегчающих разработку больших программ, имя модуля Паскаля должно совпадать с именем дискового файла, в который помещается исходный текст модуля.* Если, например, имеем заголовок модуля Паскаля

Unit primer ;

то исходный текст этого модуля должен размещаться на диске в файле primer .pas .

Имя модуля Паскаля служит для его связи с другими модулями и основной программой. Эта связь устанавливается специальным предложением:

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

uses<список модулей>

Здесь **USES** – зарезервированное слово (использует);

<список модулей> - список модулей, с которыми устанавливается связь; элементы списка – имена модулей через запятую.

Если в Паскале модули используются, то предложение **uses <список модулей>** должно стоять сразу после заголовка программы , т.е. должно открывать раздел описаний основной программы. В модулях Паскаля могут использоваться другие модули. В модулях предложение uses <список модулей> может стоять сразу после слова interface или сразу после слова implementation . Допускается и два предложения uses , т.е. оно может стоять и там, и там.

Интерфейсная часть

Интерфейсная часть открывается зарезервированным словом **INTERFACE** . В этой части содержатся объявления всех глобальных объектов модуля (типов, констант, переменных и подпрограмм), которые должны быть доступны основной программе и (или) другим модулям Паскаля. При объявлении глобальных подпрограмм в интерфейсной части указывается только их заголовок, например:

```
Unit complexn;
Interface Type Complex= record
  Re, im: real;
End;
Procedure AddC(x,y: complex, var z: complex);
Procedure MulC (x,y: complex, var z: complex);
```

Если теперь в основной программе написать предложение Uses complexn ; то в программе станут доступными тип cmplex и две процедуры – AddC и MulC из модуля complexn .

Отметим, что объявление подпрограмм в интерфейсной части автоматически сопровождается их компиляцией с использованием дальней модели памяти. Таким образом, обеспечивается доступ к подпрограммам из основной программы и других модулей Паскаля.

Следует учесть, что все константы и переменные, объявленные в интерфейсной части модуля Паскаля, равно как и глобальные константы и переменные основной программы, помещаются компилятором Турбо Паскаля в общий сегмент данных (максимальная длина сегмента 65536 байт).

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Порядок появления различных разделов объявлений и их количество может быть произвольным. Если в интерфейсной части объявляются внешние подпрограммы или подпрограммы в машинных кодах, их тела (т.е. зарезервированное слово EXTERNAL , в первом случае, и машинные коды вместе со словом INLINE – во втором) должны следовать сразу за их заголовками в исполняемой части модуля (не в интерфейсной!). В интерфейсной части модулей Паскаля нельзя использовать опережающее описание.

Исполняемая часть модуля Паскаля

Исполняемая часть модуля Паскаля начинается зарезервированным словом **IMPLEMENTATION** и содержит описания подпрограмм, объявленных в интерфейсной части. В ней могут объявляться локальные для модуля объекты – вспомогательные типы, константы, переменные и блоки, а также метки.

Описанию подпрограммы, объявленной в интерфейсной части модуля Паскаля, в исполняемой части должен предшествовать заголовок, в котором можно опустить список формальных параметров и тип результата для функции, так как они уже описаны в интерфейсной части. Но если заголовок подпрограммы приводится в полном виде, т.е. со списком параметров и объявлением типа результата для функции, то он должен полностью совпадать с заголовком подпрограммы в интерфейсной части, например:

```
Unit complexn;
{-------------------------------}
Interface
Type
Complex= record
  Re, im: real;
End;
Procedure AddC(x,y: complex, var z: complex);
{-------------------------------}
Implementation
  Procedure AddC;
    z.re:= x.re + y.re;
    z.im:= x.im + y.im;
  end ;
end .
```

Инициирующая часть модуля Паскаля

Инициирующая часть завершает модуль Паскаля. Она может отсутствовать вместе с начинающим ее словом BEGIN или быть пустой – тогда вслед за BEGIN сразу следует признак конца модуля.

В инициирующей части размещаются исполняемые операторы, содержащие некоторый фрагмент программы. Эти операторы исполняются до передачи управления основной про-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

грамме и обычно используются для подготовки ее работы. Например, в инициирующей части могут инициироваться переменные, открываться файлы, устанавливаться связи с другими компьютерами и т.п.:

```
Unit fileText;
{------------------------------}
Interface
  Procedure print(s: string);
{------------------------------}
implementation
var f: text;
const
  name= ‘output.txt’;
procedure print;
begin
  writeln(f, s)
end ;
{--------------------------------}
{начало инициирующей части}
begin
  assign(f, name);
  rewrite ( f );
{конец инициирующей части}
end .
```

Не рекомендуется делать инициирующую часть пустой, лучше ее опустить.

Компиляция модулей в Паскаль

В среде Турбо Паскаль имеются средства, управляющие способом компиляции модулей и облегчающие разработку больших программ. Определены три режима компиляции: **COMPILE , MAKE , BUILD**. Режимы отличаются способом связи компилируемого модуля или основной программы с другими модулями, объявленными в предложении USES .

При компиляции модуля или основной программы в режиме **COMPILE** все, упоминаемые в предложении USES модули, должны быть предварительно откомпилированы, и результаты компиляции должны быть помещены в одноименные файлы с расширением TPU (от англ. Turbo Pascal Unit). Файл с расширением TPU создается автоматически при компиляции модуля Паскаля.

В режиме **MAKE** компилятор проверяет наличие TPU -файлов для каждого объявленного модуля. Если какой-либо файл не найден, система ищет одноименный файл с расширением PAS , т.е. файл с исходным текстом модуля Паскаля. Если таковой файл найден, система приступает к его компиляции. Кроме того, в этом режиме система следит за возможными изменениями исходного текста любого используемого модуля. Если в PAS -файл внесены изменения, то независимо от того, есть ли в каталоге соответствующий TPU

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

-файл или нет, система откомпилирует его перед компиляцией основной программы. Более того, если изменения внесены в интерфейсную часть, то будут откомпилированы все другие модули, обращающиеся к нему. Режим MAKE существенно облегчает процесс разработки крупных программ с множеством модулей Паскаля: программист избавляется от необходимости следить за соответствием TPU -файлов их исходному тексту, т.к. система делает это автоматически.

В режиме **BUILD** существующие TPU -файлы игнорируются, система пытается отыскать и откомпилировать соответствующие PAS - файлы для каждого модуля Паскаля. После компиляции можно быть уверенным, что учтены все сделанные в текстах модулей Паскаля исправления и изменения.

Подключение модулей Паскаля к основной программе и их компиляция происходит в порядке их объявления в предложении USES . При переходе к очередному модулю Паскаля система предварительно ищет все модули, на которые он ссылается. Ссылки модулей Паскаля друг на друга могут образовывать древовидную структуру любой сложности, однако запрещается явное или косвенное обращение модуля к самому себе. Например, недопустимы следующие объявления:

Unit A; Unit B;
interface interface
uses B;Uses A;
……. ……
implementation implementation
…….. ……
end. end.

Это ограничение можно обойти, если «спрятать» предложение USES в исполняемые части зависимых модулей:

Unit A; Unit B;
interface interface
……. ……
implementation implementation
uses B;Uses A;
…….. ……
end. end.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Дело в том, что Турбо Паскаль разрешает ссылки на частично откомпилированные модули, что приблизительно соответствует опережающему описанию подпрограммы. Если интерфейсные части независимы (это обязательное условие!), Турбо Паскаль сможет идентифицировать все глобальные объекты в каждом модуле, после чего откомпилирует тела модулей обычным способом.

ПРИМЕР

1. Разработать модуль, содержащий подпрограмму суммирования элементов массива.

Разбиваем текст программы на две части: подпрограмму размещаем в модуле, а тестирующую программу оставляем в качестве основной программы. Так как все структурные типы параметров должны быть предварительно объявлены, описываем тип массива в модуле.

```
{Модуль должен размещаться в файле Summa.pas}
Unit Summa;
Interface {объявление внешних ресурсов}
Type mas=array[1..10] of integer;
Function sum(b:mas;n:integer):integer;
Implementation .
Function sum; {описание функции}
Var s,i: integer;
begin
s:=0;
for i:=1 to n do s:=s+b[i];
sum:=s;
end;
end.
```

Программа использует из модуля два ресурса: описание типа mas для объявления массива A и функцию Sum.

```
Program ex;
Uses Summa; {указание используемого модуля}
Var a:mas; {используем ресурс mas}
i,n: integer;
Begin
readln(n);
for i:=1 to n do re ad (a [i]);
ReadLn;
WriteLn('Сумма= sum(a,n)); {используем ресурс sum}
end.
```

2. Теперь напишем небольшой модуль. Назовем его IntLib и вставим в него две простые подпрограммы для целых чисел - процедуру и функцию:

```
unit IntLib;
 interface
 procedure ISwap(var I,J : integer);
```


Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
function IMax(I,J : integer) : integer;
implementation
procedure ISwap;
var
Temp : integer;

begin
Temp := I; I := J; J := Temp
end; { конец процедуры ISwap }
function IMax;
begin
if I > J
then IMax := I
else IMax := J
end; { конец функции IMax }
end. { конец модуля IntLib }
```

Наберите этот модуль, запишите его в файл INTLIB.PAS. Перешлем его в каталог модулей (если такой имеется), или оставим в том же каталоге, где находится следующая программа, которая использует модуль IntLib:

```
program IntTest;
uses IntLib;
var
A,B : integer;
begin
Write('Введите два целочисленных значения: ');
Readln(A,B);
ISwap(A,B);
Writeln('A = ',A,' B = ',B);
Writeln('Максимальное значение равно ',IMax(A,B));
end. { конец программы IntTest }
```

Контрольные вопросы.

1. Какие модули относятся к стандартным модулям?
2. Опишите функции каждого стандартного модуля SYSTEM, DOS, CRT, GRAPH, OVERLAY, TURBO.
3. Какие еще могут быть модули в Паскаль?
4. Структура модуля в Паскаль.
5. Расскажите о заголовке модуля и имени файла, содержащего модуль.
6. Как подключить модули к файлу с решением задачи?
7. Компиляция модулей в Паскаль.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ИНТЕРФЕЙС СРЕДЫ ПРОГРАММИРОВАНИЯ VISUAL STUDIO 2010

Структура проекта VB

Перед рассмотрением вопроса проектирования интерфейса приложения на Visual Basic, необходимо представлять, из чего вообще состоит этот проект?

В Visual Basic любой проект состоит из следующих файлов:

- файл каждой формы (расширение frm). Это обычный ASCII текстовый файл, в котором записан весь код, помещённый в форму, а также свойства всех помещённых на форму элементов управления и самой формы тоже.
- файл каждой формы, содержащий бинарную информацию (например картинку в PictureBox) (расширение frx)
- файл проекта, содержащий информацию о проекте (расширение vbp)
- информация о рабочей области проекта (workspace) (расширение vbw)

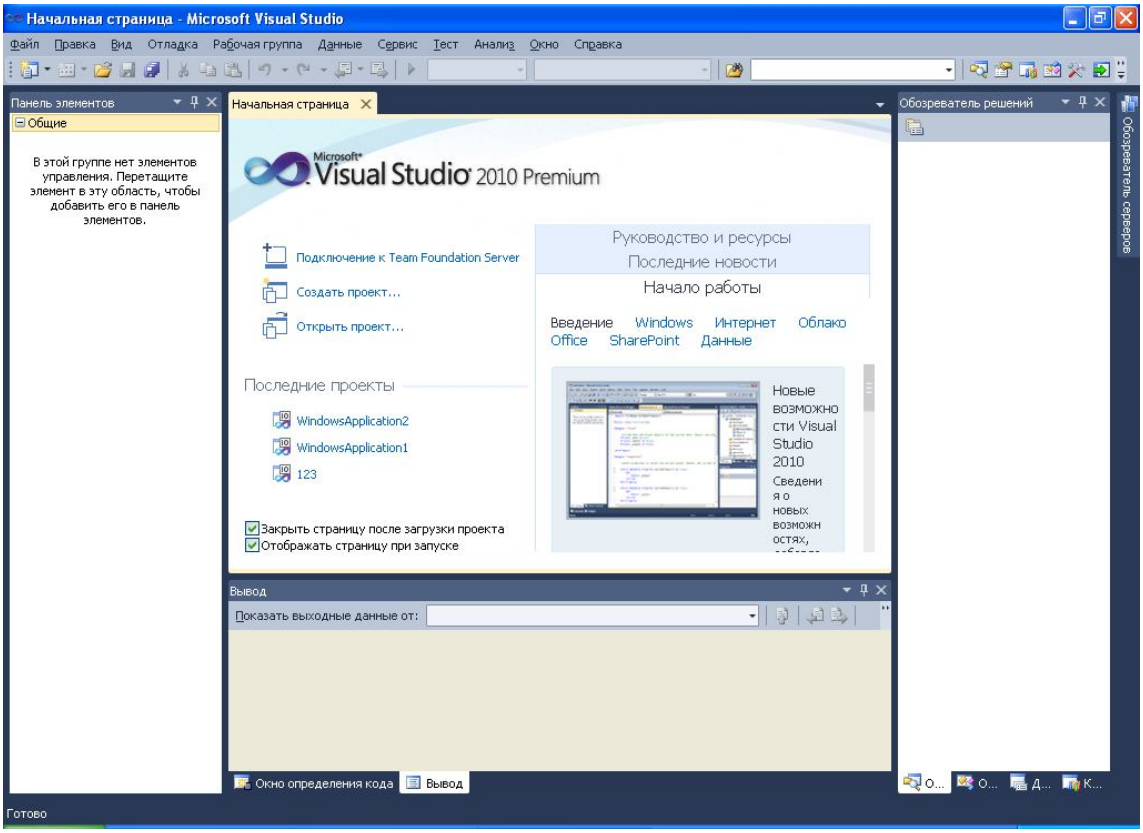
Это необходимый минимум. (Хотя, бывают и исключения, например, когда в проекте не используются формы. Тогда вместо frm файла, будет bas файл.)

Далее перечислим дополнительные файлы, которые могут быть подключены к проекту:

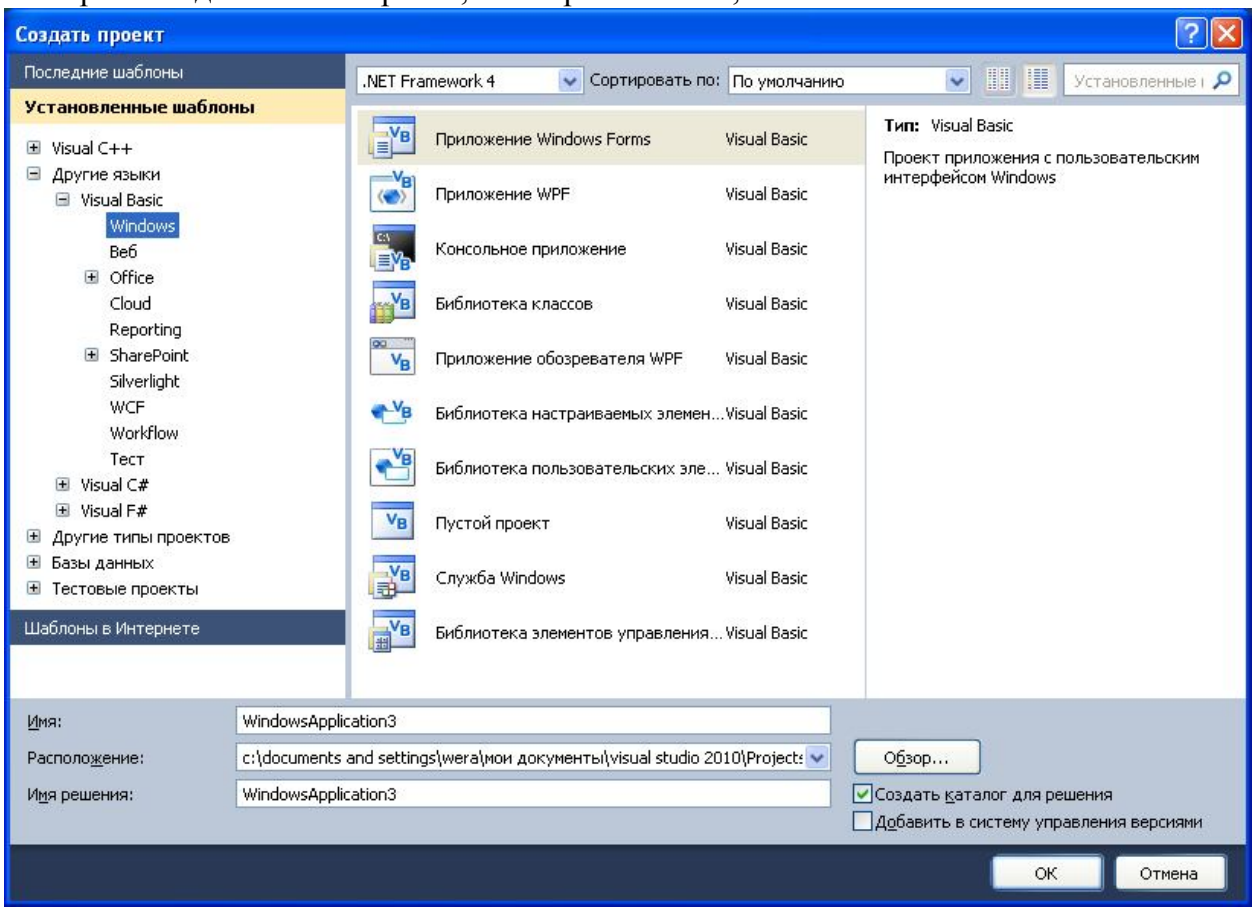
- файл каждого модуля (расширение bas) Это текстовый файл.
- файл каждого модуля классов (расширение cls). Это текстовый файл.
- файл каждого дополнительного элемента управления (расширение ctl) Это тоже текстовый файл.
- файл ресурсов (расширение res)
- другие файлы (ocx, tlb, и т.д...)

Выполнив запуск программы из главного меню операционной системы откроется окно с начальной странице для выбора готового объекта или для создания нового.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

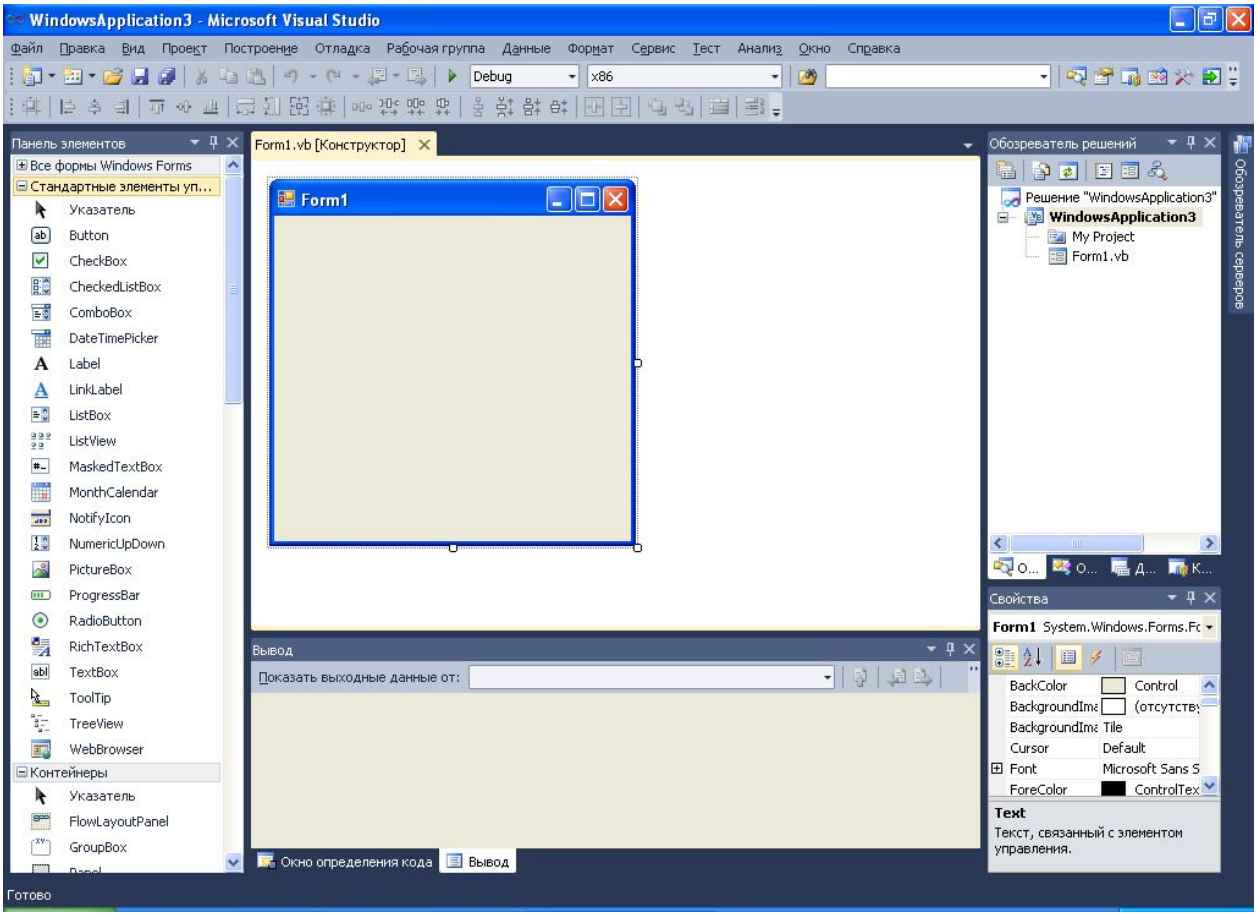


Если выбрать создать новый проект, то откроется окно,

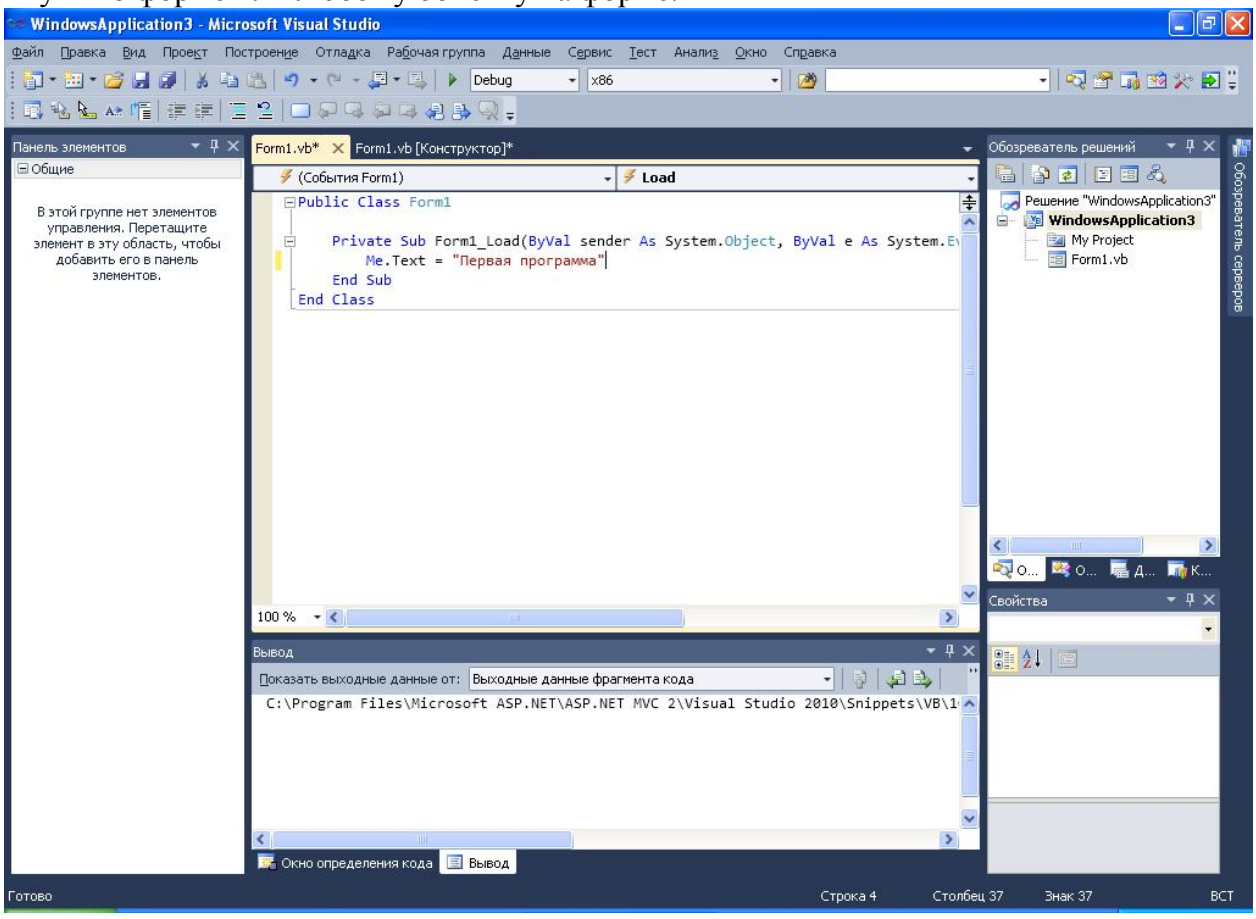


в котором выбираем тип создаваемого проекта.
Окно проекта.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)



Окно программного кода проекта. Чтобы открыть окно кода проекта можно дважды щелкнуть по форме или любому объекту на форме.

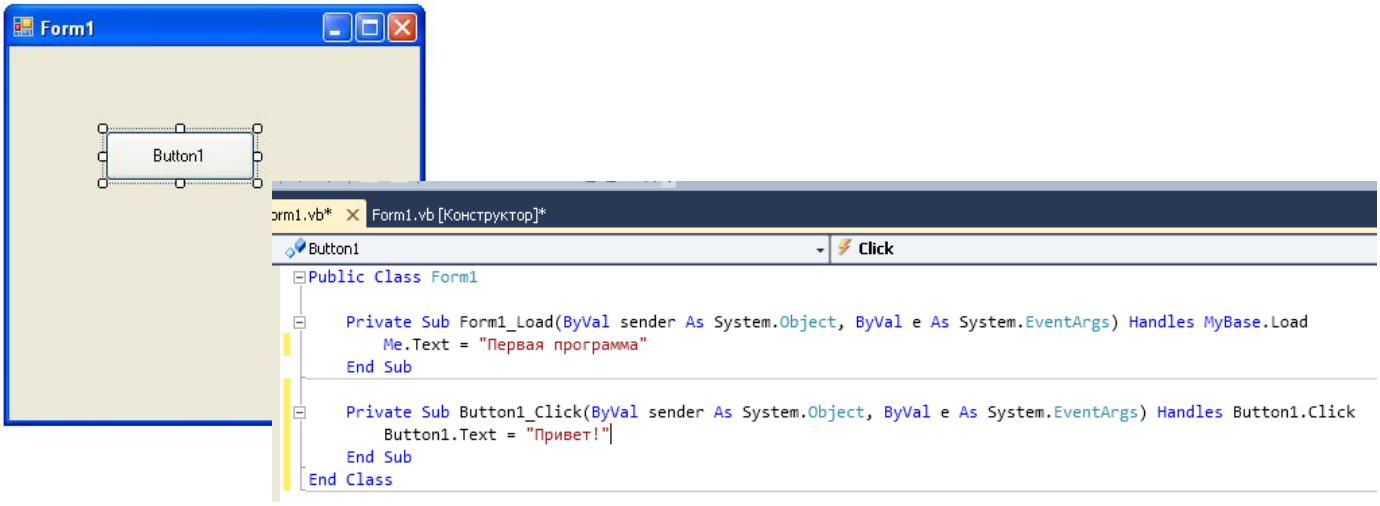


Внутри конструкции Public Class Form1

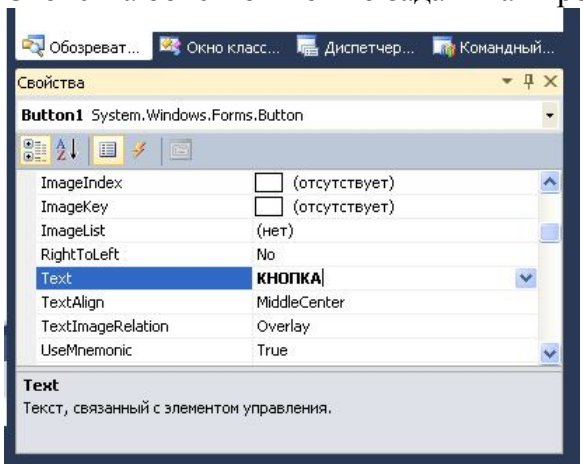
Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

End Class

и происходит размещение процедур и функций проекта. Конструкции подпрограмм формируются автоматически при двойном нажатии на объект на форме. Например, если разместить на форме кнопку и дважды по ней щелкнуть мышью, то получим конструкцию процедуры Private Sub …. End Sub



Свойства объектов можно задать как программным способом, так и в окне свойств.



Задание для самостоятельного выполнения.

- Создать проект.
- Разместить на форме 5 различных элементов.
- Изменить для каждого от 4 до 7 свойств, используя окно свойств.
- Сохранить проект в папке со своим именем на рабочем столе.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ВВОД ДАННЫХ В VISUAL BASIC

К системным функциям относятся функции, действие которых напрямую зависит от работы системы Windows. Это:

• функция InputBox – для ввода данных пользователем через системное окно;

• функция MsgBox – для выдачи сообщений пользователю через системное окно.

Работа этих функций сопровождается появлением на экране одного из двух окон: Окна ввода (InputBox) и Окна сообщений (MsgBox)

Окно InputBox состоит из четырех элементов:

1) строка заголовка (title) ;

2) приглашение к вводу (prompt) ;

3) поле ввода со значением, предлагаемым по умолчанию (default);

4) координаты левого верхнего угла окна на экране;

5) две кнопки (OK и Cancel).

Функция вызова окна InputBox имеет следующий синтаксис с соответствующими аргументами:

P = InputBox (prompt [,title] [,default] [,Xpos] [,Ypos]),

где P – возвращаемое значение функции;

Xpos и Ypos – координаты левого верхнего угла окна.

Все необязательные параметры указаны в квадратных скобках. При щелчке пользователем на кнопке OK функция InputBox возвращает строку, введенную пользователем. При щелчке пользователем на кнопке Cancel возвращается пустая строка.

Например:

```
Public Class Form1
 Dim yourName As String
 Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
  yourName = InputBox("Введите число лет", "Возраст")
  Label1.Text = "привет"
  Label2.Text = "тебе уже"
```


Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
  Label3.Text = yourName + " !"
 End Sub
End Class
```

Окно MsgBox

Для вывода различных сообщений в Visual Basic имеется окно MsgBox.

Вид окна может быть различным, но в его состав всегда входят:

- текст сообщения (prompt);
- заголовок (title);
- пиктограмма;
- набор кнопок.

Синтаксис команды: **MsgBox (prompt [,type] [,title])**

где type – целая константа (или несколько констант), определяющая, какие кнопки (комбинации кнопок) и пиктограммы будут отображаться в окне сообщения.

Параметр type определяет внешний вид MsgBox. Значение параметра формируется из нескольких частей, которые можно складывать: button, icon, default, modal, extras.

Для категории параметра button, icon, default, modal можно использовать только одну из допустимых констант. Для категории extras допускается применение комбинации значений.

Ниже приведена таблица констант параметра type окна MsgBox.

Константа	Значение	Описание
Категория Button		
VbOkOnly	0	Только кнопка ОК
VbOkCancel	1	Кнопки ОК и Отмена
VbAbortRetryIgnore	2	Кнопки Стоп, Повторить, Пропустить
VbYesNoCancel	3	Кнопки Да, Нет и Отмена
VbYesNo	4	Кнопки Да и Нет
VbRetryCancel	5	Кнопки Повторить и Отмена
Категория Icon		
VbCritical	16	Отображает пиктограмму Critical Message
VbQuestion	32	Отображает пиктограмму Warning Query
VbExclamation	48	Отображает пиктограмму Warning Message
VbInformation	64	Отображает пиктограмму Information Message
Категория Default		
VbDefaultButton1	0	По умолчанию активна первая кнопка
VbDefaultButton2	256	По умолчанию активна вторая кнопка
VbDefaultButton3	512	По умолчанию активна третья кнопка
VbDefaultButton4	768	По умолчанию активна четвертая кнопка

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Категория Modal		
VbApplicationModal	0	Модальное диалоговое окно приложения
VbSystemModal	4096	Модальное диалоговое окно системы
Категория Extras		
VbMsgBoxHelpButton	16384	Дополнительная кнопка для справки
VbMsgBoxSetForeground	65536	Отображение диалогового окна в фоновом режиме
VbMsgBoxRight	524288	Текст выровнен по правому краю
VbMsgBoxRtReading	1048576	Текст отображается справа налево (еврейский, арабский)

Функция MsgBox( )

Если вам необходимо не только что-либо сообщить пользователю, но и получить информацию о том, какое решение принял пользователь, можно воспользоваться функцией MsgBox. Функция MsgBox( ) возвращает некоторое значение (ответ пользователя о том, какое он принял решение).

Синтаксис функции MsgBox(): **Возвращаемое_значение= MsgBox (prompt [,type] [,title])**

Возвращаемое_значение позволяет определить, какую кнопку нажал пользователь. В таблице, которая приведена ниже, содержатся значения, возвращаемые функцией MsgBox:

Константа	Значение	Нажата кнопка
VbOk	1	OK
VbCancel	2	Отмена
VbAbort	3	Стоп (Прервать)
VbRetry	4	Повторить
VbIgnore	5	Пропустить
VbYes	6	Да
VbNo	7	Нет

Контрольные вопросы.

1. Что представляет собой Visual Studio?
2. Что такое Visual Basic?
3. Что представляет собой программный код проекта?
4. Сколько файлов содержит проект?
5. Какие окна пользователь видит перед собой при открытии проекта?
6. Как открыть окно программного кода?
7. Нужно ли самостоятельно прописывать заголовок событийной процедуры?
8. Опишите функцию InputBox.
9. Опишите функцию MsgBox
10. Опишите процедуру MsgBox.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ТИПЫ ДАННЫХ И ОПЕРАЦИИ РАБОТЫ С НИМИ.

Правила написания программ. В каждой строке располагается один или несколько, разделенных **:** операторов.

Для переноса продолжения строки на другую ставится *пробел* и *знак подчеркивания*.

В вещественных числах целыая часть от дробной разделяется *точкой*.

Комментарии - не выполняемые редактором тексты, строка комментария начинается с **'** ***текст*** или **Rem** ***текст.***

В VB все данные относятся к определенному типу, основные из них

Тип данных	**Размер (байт)**	**Описание**	**Символы описания типов**
Integer	2	Целые числа	%
Long	4	Длинные целые числа	&
Single	4	Вещественные одинарной точности	!
Double	8	Вещественные двойной точности	#
String	10 байт + длина строки	Строки символов переменной длины	$
String *	1байт/символ	Строки символов постоянной длины	$
Boolean	2	Логмческие (булевые)	
Currency	8	Числа в денежном формате	@
Date	8	Знгачения даты и времени	
Byte	1	Целые числа (0-255)	
Object	4	Объекты (ссылки на объекты)	
Variant	16	Тип определяется содержимым	

Описание типов данных. Для описания типов переменных используется оператор

Dim имя As mun [, имя As mun], . . .

где *имя* – имя переменной, набор символов не более 255 символов, должны начинаться с буквы, ***не может*** содержать пробел, . ! @ & $ #; не допускается использование повторяющихся имен на одном уровне области определения,

mun - тип переменной (*Integer, Long, Currency, Single, Double, Date, String*, *String**, *Object, Variant)*. По умолчанию переменная получает тип *Variant.*

Оператор требования **обязательного описания переменных,** устанавливается в начале процедуры

Option Explicit

Данные могут изменяться в ходе программы (переменные) и не изменяться (константы). Строковые константы беруться в кавычки. По умолчанию численная переменная равна 0, строковая – пустая строка.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Операции VB

Арифметические операции		**Операции сравнения**		**Логические операции**	
^	Возведение в степень	=	Равно	*Not*	Логическое отрицание
* /	Умножение, деление	<>	Не равно	*And*	Логическое «И»
\	Целочисленное деление	<	Меньше	*Or*	Логическое «ИЛИ»
Mod	Деление по модулю	>	Больше	*Eqv*	Эквивалентность
+ -	Сложение, вычитание	<=	Меньше или равно	*Результат логических операций: False (Ложь) или True (Истина)*	
		>=	Больше или равно		
& или +	Объединение строк				

Функции *преобразования*

Val(*строка*)	преобразует строку цифровых символов (до первого нецифрового символа) в число
Str(*число*)	преобразует число в строку
Chr(*код символа*)	преобразует код символа в символ

***Арифметические* функции**

Математическое описание	**Программное описание**
cosx, sinx, tgx, arctgx,	*Cos(x), Sin(x), Tan(x), Atn(x)*
e^x, *lnx*, √x	*Exp(x), Log(x), Sqr(x)*
Вычисление по модулю |x|	*Abs(x)*
Генерация слачайных чисел от 0 до 1	*Randomize, Rnd*
Определение знака	*Sgn(x)*
Выделение целой части	*Fix(x), Int(x)*

Оператор присваивания. Оператор вычисляет выражение и присваивает его переменной

Имя переменной = выражение

Задание для самостоятельного выполнения

Наберите программу вычисления квадратного корня из введенного числа и проанализируйте используемые переменные и процедуры.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
' Программа вводит через текстовое поле число, при нажатии
' командной кнопки извлекает из него квадратный корень и выводит
' результат на метку Label1. В случае ввода не числа сообщает
' пользователю об этом, очищая текстовое поле. Есть еще одна кнопка
' Очистка для обнуления текстового поля и метки.
Public Class Form1

  Private Sub Form1_Load(ByVal sender As System.Object,
              ByVal e As System.EventArgs) Handles MyBase.Load
    Me.Text = "Извлечение квадратного корня"
    Button1.Text = "Извлечь корень"
    Button2.Text = "Очистка"
    TextBox1.Clear()    ' Очистка текстового поля
    Label1.Text = ""
    Label1.TextAlign = ContentAlignment.MiddleCenter
  End Sub

  Private Sub Button1_Click(ByVal sender As System.Object,
                  ByVal e As System.EventArgs) Handles Button1.Click

    ' Обработка события "щелчок на кнопке" Извлечь корень
    If Not IsNumeric(TextBox1.Text) Then
      MessageBox.Show("Следует вводить числа", "Ошибка",
                  MessageBoxButtons.OK, MessageBoxIcon.Error)
      TextBox1.Clear()    ' Очистка текстового поля
      TextBox1.Focus()    ' Установить фокус на текстовом поле
      Exit Sub
    End If

    Dim X, Y As Single
    ' Преобразование из строковой переменной в Single
    X = Convert.ToSingle(TextBox1.Text)
    Y = Math.Sqrt(X)
    Label1.Text = "Корень из " + X.ToString + " равен " + Y.ToString
  End Sub

  Private Sub Button2_Click(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles Button2.Click

    ' Обработка события "щелчок на кнопке" Очистка:
    Label1.Text = ""
    TextBox1.Clear()  ' Очистка текстового поля
    TextBox1.Focus()
  End Sub
End Class
```


Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ИТОГОВОЕ ЗАНЯТИЕ ПО ТЕМЕ «ВИЗУАЛЬНОЕ ПРОГРАММИРОВАНИЕ»

На итоговом занятии можно провести рубежный контроль знаний в форме теста и решения задачи с использованием изученных элементов. Такое проверочное задание необходимо для дифференцированного подхода к обучающимся, по уровню знаний.

Пример тестовых вопросов для контроля.

1. Для чего служит окно “Properties”?
    A. Для отображения свойств выбранного объекта
    B. Для изменения свойств выбранного объекта
    C. Для отображения списка форм в проекте
    D. Для показа макета проекта
2. Как отобразить окно свойств, если оно не видно в рабочей среде?
    A. Меню "View" - "Toolbox"
    B. Меню "View" - "Properties Window"
    C. Меню "View" - "Form Layout Window"
    D. Меню "View" - "Project Explorer"
3. Какое расширение получает файл проекта при сохранении его на диске?
    A. frm
    B. bas
    C. vbp
    D. exe
4. Что такое проект в Visual Basic?
    A. Набор файлов различных форматов
    B. Программа на языке Visual Basic
    C. Набор окон рабочей среды
    D. Алгоритм выполнения программы
5. Какое расширение получает файл формы при сохранении его на диске?
    A. *.vbp
    B. *.bas
    C. *.frm
    D. *.exe
6. Сколько форм может содержать проект?
    A. Одну
    B. Три
    C. Сколько угодно
    D. Десять
7. Какая функция выводит на экран текущее время?
    A. DateString
    B. TimeString
    C. Time.ToString
    D. Date.ToString

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

8. Какой из этих функций мы загрузим текст файла указанный в TextBox1?
    A. System.IO.StreamReader
    B. System.File.ToString
    C. System.File.Read
    D. System.IO.StreamWriter
9. Команда делает видимой форму 2 и позволяет мышью переключаться на первоначальную форму
    A. Form2.Show
    B. Form2.Windowstate
    C. Form2.ShowDialog
    D. Form2.Hide
10. Функция Now служит для
    A. Определения системного часового пояса пользователя,
    B. Возвращения текущего системного значения даты и времени,
    C. Изменения текущего системного значения даты и времени,
    D. Получения и изменения системного значения даты и времени
11. Чтобы процедуру можно было вызвать в другом модуле следует определить ее с помощью ключевого слова
    A. Static,
    B. Public,
    C. Private,
    D. Sub
12. Чтобы добавить в проект новую форму необходимо выполнить
    A. Меню "Проект" - "Add Form" - в окне "Add Form" - выбрать значок "Form",
    B. На панели инструментов выбрать кнопку "Add Form",
    C. Меню "File" - "Add Form" - в окне "Add Form" - выбрать значок "Form",
    D. Меню "Построение" - "Add Form" - в окне "Add Form" - выбрать значок "Form"
13. С каким объектом связана событийная процедура
    A. Form1,
    B. Imege1,
    C. Button1,
    D. Label2

```
Private Sub Button1_Click()
Image1.Width = Image1.Width * 2
Image1.Height = Image1.Height * 2
Form1.Text = "проверка знаний"
Label2.Color = Red
End Sub
```


Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

14. Чем отличаются между собой подпрограммы-процедуры и подпрограммы-функции
    A. Принципами выполнения входящих операций,
    B. Набором исполняемых в подпрограмме операторов,
    C. Принципами передачи параметров в подпрограмму,
    D. Методом вызова и количеством возвращаемых результатов
15. Свойства объекта определяют
    A. Состояние объекта,
    B. Поведение объекта,
    C. Что может происходить с объектом,
    D. Выдаваемые результаты
16. Проект в Visual Basic представляет собой
    A. Набор файлов различных форматов,
    B. Программа на языке Visual Basic,
    C. Набор окон рабочей среды,
    D. Алгоритм выполнения программы
17. Подпрограмма, которая начинает выполняться после реализации определенного события, называется...
    A. Свойством объекта,
    B. Методом объекта,
    C. Графическим интерфейсом,
    D. Событийной процедурой
18. Какая переменная является параметром цикла?

Код Visual Basic

```
1 Byti = 10 : bytst = 2 : bytk = 1
2 For bytj = bytk to byti step bytst
3 bytS = bytj
4 Label1.Text=Str( bytS)
5 Next bytj
6
```

    а) bytk
    б) bytj
    в) bytst
    г) byti
19. Ключевые слова ByRef и ByVal используются для задания
    A. Способа передач параметров,
    B. Области видимости процедуры,
    C. Имен переменных,
    D. Результатов подпрограммы
20. Какие файлы записываются на диск при сохранении проекта?

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

A. Файл элементов проекта,
B. Файл процедур элементов,
C. Файл программного кода,
D. Файл свойств формы

21. При сохранении проекта на диске его файл получает расширение
A. .frm,
B. .bas,
C. .vbp,
D. .com

22. Для чего используется компонент OpenFileDialog?
A. ввод и редактирование данных
B. открытие файлов заданного типа
C. открытие окна ввода данных
D. осуществление диалога

23. Какие из перечисленных ниже - целые типы данных в Visual Basic?
A. Long
B. Smallint
C. Single
D. Currency

24. Событие MouseMove...
A. Нажатие на мышь.
B. Удаление мыши с объекта.
C. Перемещение мыши по объекту.
D. Двойной щелчок мышью.

25. Для чего используется компонент ColorDialog?
A. задание цвета шрифта данных,
B. открытие окна задания цвета объекта,
C. открытие окна свойств данных,
D. осуществление диалога о цвете.

26. Что будет напечатано в результате выполнения программы?

Код Visual Basic

```
   intX = -3 : intY =
1  10
2  IF intX * intY >10
3  THEN
4  intX = intY * intX
5  ELSE
6  IF  intX * intY<0
7  THEN
8  intY = intY-5
9  End if
10 End if
   intS = intY - intX
```

Код Visual Basic

A. 8
B. 2
C. -25
D. 35

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Пример типовых задач для проверки освоенных знаний и умений

Создать проект для решения следующей задачи.

1. Ввести с клавиатуры текстовую строку и вывести на форму в текстовое поле. После нажатия на кнопку «Заново», введенный текст удаляется из окна и выводится на форму. Следующий текст добавляется к имеющемуся на форме.

1.1 Добавить кнопку на форму «Закончить ввод» при нажатии на которую кнопка «Заново» становится недоступной (свойство Enabled).

2. На форме разместить кнопку для запроса числовых данных и вывода их на форму. Элементами Checkbox и Radio Button задать выполнение операций целочисленного деления и взятия остатка от целочисленного деления. Оформить вывод результатов так чтобы были представлены значения введенных величин, знак операции, знак равенства и результат (например, 25 mod 3=1)

3. Организовать ввод чисел с помощью Input Box до первого встретившегося нуля. Вывести эти числа на форму в Textbox. Определить максимальное и минимальное из них с использованием цикла While и Repeat. Разместить найденные числа на разных вкладках на форме. Вывести на форму пояснения по имеющейся информации.

4. Создать на форме меню с тремя пунктами и с двумя командами в каждом пункте. Организовать смену фона формы на картинку и отмену этого действия. Организовать смену цвета и типа шрифта в меню с использованием диалоговых окон. Вывести на форму пояснения по имеющейся информации.

5. Написать программу перевода числа в двоичную систему счисления и обратно с использованием подпрограмм – функций. На форме предусмотреть меню, вкладки или переход на вторую форму. Оформить проект изображением как фоном. Число вводить в Textbox и предусмотреть многократный ввод числа. Вывести на форму пояснения по имеющейся информации.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

РАЗДЕЛ 3 ЛАБОРАТОРНЫЕ ЗАНЯТИЯ

Я слышу и забываю.

Я вижу и запоминаю.

Я делаю и понимаю.

(Конфуций)

Лабораторные работы №1- № 4.

Задание

1.По готовой программе составьте список структурных элементов программного кода.

2. Опишите входные и выходные данные каждого структурного элемента.

```
uses crt; const m=3;  n=3;
 a:array [1..m,1..n] of integer=((1,3,4),
                                 (182,93,2),
                                  (2,3,4));
var i,j,k,l,r,q,max:integer;
    b:array[1..m] of string[15];
    c:array[1..n] of string[25];
label m1;
begin
clrscr;
{Определяем количество разрядов в максимальном числе, следовательно название столбца должно быть длинее самого большого числа. Если же число отрицательное, то делаем его положительным}
r:=0;
for i:=1 to n do
   for j:=1 to m do
      begin
      q:=a[i,j];
      if q<0 then q:=q*(-1);
      while q div 10 >=1 do  {Определяем количество разрядов}
           begin
               r:=r+1;
               if r>max then max:=r;
               q:=q div 10;
           end;
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
      end;
for i:=1 to n do
   begin
      m1:
      writeln('Введите имя ',i,' столбика');
      readln(b[i]);
      if length(b[i])<max then
      begin
         writeln('Название столбика должно быть ',max,' и более символов');
         goto m1;
      end;
   end;
max:=0;
for i:=1 to m do
   begin
      writeln('Введите имя ',i,' строки');
      readln(c[i]);
      if length(c[i])>max then max:=length(c[i]);
   end;
{Рисуем шапку}
write(' ');
for k:=1 to max do
   write('_');
for k:=1 to n do
   begin
      write(' ');
      for l:=1 to length(b[k]) do
      write('_');
   end;
writeln;
write('|');
for k:=1 to max do
   write(' ');
for k:=1 to n do
   begin
      write('|');
      write(b[k]);
      if k=n then
      write('|');
   end;
writeln;
write(' ');
for k:=1 to max do
   write('-');
for k:=1 to n do
   begin
      write(' ');
      for l:=1 to length(b[k]) do
      write('-');
   end;
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
for j:=1 to n do
   begin
      writeln;
      if length(c[j])<=max then  {Пишем названия строк, при этом отступаем}
      begin
          write('|',c[j]);
          for l:=1 to max-length(c[j]) do
              write(' ');
          write('|');
      end
      else
          write('|',c[j]);
      for i:=1 to m do
         begin
             write(a[j,i]:length(b[i]));
             write('|');
         end;
      if i=m then
        begin
            writeln;
            write(' ');
            for k:=1 to max do
               write('-');
            for k:=1 to n do
               begin
                   write(' ');
                   for l:=1 to length(b[k]) do
                      write('-');
               end;
        end;
   end;
readln;
end.
```

Задачи на подпрограммы

1. Написать процедуру (функцию) нахождения НОД(a,b)
2. Напишите функцию логического типа, проверяющую являются ли все цифры числа различными.
3. Даны три точки (x1,y1), (x2,y2), (x3,y3). Составить программу, выдающую площадь треугольника или сообщение что точки не служат вершинами треугольника. Для решения задачи составить две подпрограммы: для определения являются ли точки вершинами треугольника и вычисление площади треугольника.
4. Оформить предложенный код как процедуру и выполнить отладку и тестирование готовой программы.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
Uses Crt;
Type
  rec = record
        FirstName:String;
        LastName :String;
        Age      :Byte;
  end;
Var
  Baza: Array[0..100] Of rec;
  I: Integer;
begin
For I:=0 To 4 Do
Begin
  Write('FirstName='); ReadLn(Baza[I].FirstName);
  Write('LastName='); ReadLn(Baza[I].LastName);
  Write('Age='); ReadLn(Baza[I].Age);
End;
WriteLn('FirstName LastName Age');
For I:=0 To 4 Do
  WriteLn(Baza[I].FirstName,' ',Baza[I].LastName,' ',Baza[I].Age);
End.
```

5. Выполнить отладку и тестирование программы и опишите структурные элементы.

```
Program Procedure_Function;
Uses Crt;
Var R:Real;
    n,X1,Y1:Integer;

Procedure WriteXY(Var X,Y:Integer;M:Real);
Begin
 GotoXY(X,Y); Write(M);
 X:=WhereX;
 Y:=WhereY;
End;
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
Function XStepN(X:Real;N:Integer):Real;
Var I:Integer;
    Y:Real;
Begin
Y:=1;
 For I:=1 To N Do Y:=Y*X;
 XStepN:=Y;
End;

Begin
 X1:=1; Y1:=5;
 R:=XStepN(3,4); WriteXY(X1,Y1,R);
 R:=XStepN(2,3); WriteXY(X1,Y1,R);
End.
```

6. Организовать подпрограммы рисующие цифры от **'0'** до **'9'** в заданных координатах (левый верхний угол цифры), в текстовом режиме и выполнить их вызов в основной программе так, чтобы получилось какое-то число. Пример вызова процедуры: **Number0(X,Y); Number9(X,Y);** .

Трассировка программ

Меню Debug:

Breakpoints - выдает список установленных точек останова

Evaluate/Modify... - Ctrl+F4 - вычислить/изменить значения (можно наблюдать за значениями переменных, изменять их значения и вычислять выражения с несколькими переменными)

Add watch... - Ctrl+F7 - добавление переменных для наблюдения в отдельном окне наблюдений

Add breakpoint... - добавить точку останова программы

Горячие клавиши для трассировки:

F7 - Trace - трассировка со входом в функции и процедуры

F8 - Step - трассировка с пропуском функций и процедур

Ctrl+F2 - сброс трассировки

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Лабораторные работы № 5 - № 11.

Рассмотрим пример решения задачи с файлами, в котором разберем ввод и вывод данных из файла.

Память на жестком диске компьютера разбита на параграфы, каждый размером 16 Кб. Файл может занимать только целое кол-во параграфов, даже если параграф занят не весь, то он все равно отдается файлу полностью. Файл INPUT.TXT содержит целые числа, указанные через пробел, отвечающие размерам файлов в байтах. Неохо-димо подсчитать общий объём, занимаемый ими на жестком диске. Ответ записать в файл OUTPUT.TXT .

```
program RW;
var f: text;
   i,s: longint;
Begin
  s:= 0;

Assign(f,'INPUT.TXT');
 Reset(f);
  while not Eof(f) do begin
    Read(f,i);
    s:= s + (i div 16384 + 1);
                  end;
 s:= s * 16384;
Close(f);

Assign(f,'OUTPUT.TXT');
 Rewrite(f);
 Write(f,s);
Close(f);

End.
```

Задачи для самостоятельного решения

Во всех заданиях составить две программы. Первая должна формировать типизированный файл. Вторая – считать данные из этого файла, выполнить соответствующие вычисления и записать их результаты в текстовый файл.

1. Создать типизированный файл, куда записать n целых чисел. Из исходного файла сформировать массивы четных и нечетных чисел. Определить наибольший отрицательный компонент файла и наименьший положительный.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

2. Создать типизированный файл, куда записать n целых чисел. На основе исходного файла создать массив утроенных четных чисел. Упорядочить его по убыванию элементов.

3. Создать типизированный файл, куда записать n целых чисел. Сформировать массив положительных чисел, делящихся на семь без остатка, используя элементы исходного файла. Упорядочить массив по возрастанию элементов.

4. Создать типизированный файл, куда записать n вещественных чисел. Из компонентов исходного файла сформировать массивы, из чисел, больших 10 и меньших двух. Вычислить количество нулевых компонентов файла.

5. Создать типизированный файл, куда записать n целых чисел. Из файла создать массив, элементы которого являются простыми числами и расположены после максимального элемента.

6. Создать типизированный файл, куда записать n целых чисел. Из файла целых чисел сформировать массив, записав в него только четные компоненты, находящиеся до минимального элемента.

7. Создать типизированный файл, куда записать n вещественных чисел. Сделать массив из элементов исходного файла, внеся в него числа, превосходящие среднее значение среди положительных значений файла.

8. Создать типизированный файл, куда записать n целых чисел. Из исходного файла сформировать массив, записав в него числа, расположенные в файле до максимального элемента и после минимального.

9. Создать типизированный файл, куда записать n целых чисел. Массив создать из исходного файла. Внести в него простые и совершенные числа, расположенные в файле между минимальным и максимальным элементами.

10. Создать типизированный файл, куда записать n целых чисел. Из исходного файла сформировать массив, в котором вначале расположить четные, а затем нечетные числа. Определить номера наибольшего нечетного и наименьшего четного компонентов.

11. Создать типизированный файл, куда записать n целых чисел. В файле поменять местами минимальный среди положительных элементов и третий по счету простой элемент.

12. Создать типизированный файл, куда записать n целых чисел. Из файла переписать все простые, расположенные после максимального элемента в новый файл.

13. Создать типизированный файл, куда записать n целых чисел. Найти среднее арифметическое среди положительных чисел, расположенных до второго простого числа.

14. Создать типизированный файл, куда записать n целых чисел. Поменять местами последнее совершенное и третье отрицательное числа в файле.

15. Создать типизированный файл, куда записать n целых чисел. Все совершенные и простые числа из исходного файла записать в массив, который упорядочить по возрастанию.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Лабораторные работы №12 - №18.

Модуль , реализующий функцию возведения в степень

```
unit step; {заголовок модуля}
interface
    { описание видимых программных элементов модуля }
     function powerint(a,n:integer):longint;
     function powerreal(a,b:real):real;
implementation
    { операторы инициализации элементов модуля }
    function powerint(a,n:integer):longint;
    begin
        if n=0 then powerint:=1
              else powerint:=a*powerInt(a,n-1);
    end;
    function powerreal (a, b : real) : real;
    begin
        if a > 0 then powerReal := exp (b * ln (a))
                 else
            if a < 0 then powerReal := exp (b * ln (abs (a)))
     else
        powerReal := 0
     end;
end.
```

Задание.

1) Написать модуль, реализующий функцию и процедурупоиска максимального элемента и среднего арифметического элементов одномерного массива.
2) Написать модуль, реализующий функцию, определяющую содержит ли введенное слово цифры.
3) Написать процедуру удаления лишних пробелов и добавления точки в конце строки, если ее там нет.Оформить процедуру в модуле.

Работа со встроенными модулями.

Следующая простая программа позволит определить расширенный код любой клавиши. Для завершения работы программы нажмите клавишу Esc.

```
Uses CRT;
var
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
C: Char;
begin
repeat
C := ReadKey;
if C<>#0 then
WriteLn(ord(C))
else
WriteLnCO1 ,ord(ReadKey) :8)
until C=#27 {27 - расширенный код клавиши Esc}
end.
```

Задание.

1) Написать программу, реализующую диалог с пользователем и выход из программы осуществить по нажатию на Esc.
2) Оформить фон и текст различными цветами.

Лабораторные работы №19 - №37.

Не стоит изучать язык, который не меняет вашего представления о программировании.

Реализация основных алгоритмических конструкций

Оператор If...Then...Else (Visual Basic)

```
' Multiple-line syntax:
If condition [ Then ]
   [ statements ]
[ ElseIf elseifcondition [ Then ]
   [ elseifstatements ] ]
[ Else
   [ elsestatements ] ]
End If

' Single-line syntax:
If condition Then [ statements ] [ Else [ elsestatements ] ]
```

Части

condition

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Обязательный. Выражение. Должен принимать значение True или False или должен быть типом данных, который можно преобразовть в Boolean.

Then

В однострочном синтаксисе является обязательным параметром, а в многострочной — необязательным.

statements

Необязательный. Один или несколько операторов следующих за If...Then, которые выполняются, если результатом вычисления condition является True.

elseifcondition

Требуется, если имеется ElseIf. Выражение. Должен принимать значение True или Falseили должен быть типом данных, который можно преобразовть в Boolean.

elseifstatements

Необязательный. Один или несколько операторов следующих за ElseIf...Then, которые выполняются, если результатом вычисления elseifcondition является True.

elsestatements

Необязательный. Один или несколько операторов, которые выполняются, если нет предшествующего выражения condition или elseifcondition, которое имеет значение True.

End If

Завершает блок If...Then...Else.

Оператор множественного выбора

Оператор Select Case параметр

Case значения параметра

операторы

Case значения параметра

операторы

Case значения параметра

операторы

End Select

Пример.

```
Select Case x
        Case Is > 7.1, Is < 1.1
           y = 2 * Math.Cos(3 * x)
           TextBox1.Text = "y="
        Case 3.3 To 5.3
           y = 2 * Math.Atan(3) + Math.Log10(7 * x)
           TextBox1.Text = "y="
        Case Else
           y = Math.Tan(6) - x
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
        TextBox1.Text = "y="
    End Select
```

Задание.

1. Для заданного целого положительного K и значения вещественного числа X вычислить Y = F(X) по формуле:

$$Y=\begin{cases} X^K+X+1, & K=2 \quad или \quad K=3; \\ \dfrac{1}{|X+1|}, & K=4 \quad или \quad K=5 \quad или \quad K=6; \\ \sqrt{|X+K|}+\sqrt{|X-K|}, & K>7 \quad или \quad K<2. \end{cases}$$

2. Составить код решения произвольного квадратного уравнения.
3. Определить максимальное (минимальное) из 4 чисел.

Циклический алгоритм

Do {While|Until} condition … Loop

Повторяет блок инструкций, пока условие Boolean равно True или до тех пор, пока условие станет True.

```
Do { While | Until } condition
   [ statements ]
   [ Exit Do ]
   [ statements ]
Loop
-or-
Do
   [ statements ]
   [ Exit Do ]
   [ statements ]
Loop { While | Until } condition
```

Термин	Определение
Do	Обязательный. Начало определения цикла Do.
While	Требуется, если не используется Until. Повторяет цикл до тех пор, пока condition равно False.
Until	Требуется, если не используется While. Повторяет цикл до тех пор, пока condition равно True.
condition	Необязательный. Выражение Boolean. Если condition равно Nothing, Visual Basic обрабатывает его как False.
statements	Необязательный. Один или несколько операторов, повторяемых, пока condition равно или пока не станет равно True.
Exit Do	Необязательный. Передача управления из цикла Do.
Loop	Обязательный. Завершение определения цикла Do.

While или Until можно использовать для указания condition, но не оба одновременно.

Можно проверить condition только один раз — в начале или в конце цикла. Если проверить condition в начале цикла (в инструкции Do), цикл может никогда не выполниться, даже один раз. Если проверить в конце цикла (в инструкции Loop), цикл всегда выполняется по крайней мере один раз.

Условие обычно является результатом сравнения двух значений, но оно может быть любым выражением, значение которого при вычислении имеет тип Тип данных Boolean

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

(Visual Basic) (True или False). Сюда же относятся значения других типов данных, например, числовых типов, преобразованные в тип Boolean.

Циклы Do могут вкладываться друг в друга. Также можно вложить друг в друга различные виды управляющих структур.

Exit Do

Оператор Exit Do дает альтернативный способ выхода из Do…Loop. Exit Do немедленно передает управление оператору, следующему за оператором Loop.

Exit Do часто используется после оценки некоторого условия, например в структуре If...Then...Else. Выход из цикла может потребоваться при обнаружении условия, которое делает бесполезным или невозможным продолжение итераций, например ошибочное значение или запрос на завершение. Exit Do , в частности, применяется для тестирования условия, которое может вызвать бесконечный цикл, т. е. цикл, повторяемый много раз или бесконечно. Exit Do можно использовать для выхода из цикла.

Оператор While... End While (Visual Basic)

Выполняет блок операторов, пока заданное условие True.

```
While condition
   [ statements ]
   [ Exit While ]
   [ statements ]
End While
```

Термин	Определение
condition	Обязательный. Выражение Boolean. Если condition равно Nothing, Visual Basic обрабатывает его как False.
statements	Необязательный. Одна или несколько инструкций, следующих за While, которые выполняются каждый раз, когда condition равно True.
Exit While	Необязательный. Передача управления за пределы блока While.
End While	Обязательный. Завершает определение блока While.

Пока условие остается равным True, для повторения операторов неограниченное число раз используйте структуру While...End While. Для большей гибкости с условием, с которым происходит проверка или результат, для которого вы его проверяете, предпочтительнее использовать Оператор Do...Loop (Visual Basic). Если вы хотите повторить инструкцию заданное количество раз, Оператор For... Next (Visual Basic) обычно является лучшим выбором.

Если condition равно True, все statements работают до тех пор, пока не встретится оператор End While. Затем управление передается оператору While и condition проверяется заново. Если condition по-прежнему равно True, процесс повторяется. Если — False, управление передается оператору, следующему за оператором End While.

Оператор For Each... Next (Visual Basic)

Повторяет группу операторов применительно к каждому элементу коллекции.

```
For Each element [ As datatype ] In group
   [ statements ]
   [ Continue For ]
   [ statements ]
   [ Exit For ]
   [ statements ]
Next [ element ]
```

Термин	**Определение**
element	Обязателен в операторе For Each. Необязателен в операторе Next. Переменная. Используется для циклического прохода (итерации) элементов

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

коллекции.

datatype	Обязателен, если еще не объявлен элемент element. Тип данных element.
group	Обязательный. Объектная переменная. Указывает на коллекцию, применительно к элементам которой будут повторяться операторы statements.
statements	Необязательный. Один или несколько операторов, стоящих между операторами For Each и Next и выполняющихся применительно к каждому элементу group.
Continue For	Необязательный. Передача управления в начало цикла For Each.
Exit For	Необязательный. Передача управления из цикла For Each.
Next	Обязательный. Завершение определения цикла For Each.

Цикл For Each...Next используется при необходимости повтора набора инструкций для каждого элемента коллекции или массива.

Visual Basic вычисляет коллекцию только один раз — перед началом цикла. Если в блоке операторов изменяется element или group, то эти изменения не оказывает влияния на повторение цикла.

Вложенные циклы.

Циклы For Each могут вкладываться друг в друга. При этом каждый цикл должен иметь уникальную переменную element.

Также можно вложить друг в друга различные виды управляющих структур. Дополнительные сведения см. в разделе Вложенные структуры управления (Visual Basic).

Если инструкция Next внешнего уровня вложенности встретилась раньше, чем Next внутреннего уровня, то компилятор сообщает об ошибке. При этом компилятор может обнаружить эту ошибку только в том случае, если в каждой инструкции Next указан element.

Exit For

Оператор Exit for вызывает выход из цикла For…Next и передачу управления на оператору, следующему за Next.

Любое число операторов Exit For можно разместить в цикле For Each. При использовании вложенных циклов For Each оператор Exit For вызывает выход из самого внутреннего цикла и передает управление следующему уровню вложения.

Exit For часто используется после оценки некоторого условия, например в структуре If...Then...Else. Можно использовать Exit For при следующих условиях:

Продолжать выполнение итераций не нужно или невозможно. Это может быть вызвано ошибочным значением или запросом на прерывание.

Исключение перехватывается в Try...Catch...Finally. Можно использовать Exit For в конце блока Finally.

Там бесконечный цикл, то есть цикл, которые может быть запущен большое или даже бесконечное количество раз. При обнаружении таких условий для выхода из цикла можно использовать Exit For. Дополнительные сведения см. в разделе Оператор Do...Loop (Visual Basic).

Оператор Continue For передает управление непосредственно следующей итерации цикла. Дополнительные сведения см. в разделе Оператор Continue (Visual Basic).

Задание

1. Составить конспект.
2. Выполнить вычисление факториала заданного числа. Ввод числа организовать с помощью InputBox(), вывод результатов – с помощью MsgBox ().
3. Вычислить приближенно корни уравнения $x^2 + 7x - 1 = 0$. Ввод числе организовать с помощью InputBox(), вывод результатов – с помощью MsgBox (). (первый корень в промежутке [0;1], второй - [-7;-8])

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Работа с массивами в Visual Basic.

Массив - это набор однотипных переменных, объединенных одним именем и доступных через это имя и порядковый номер переменной в наборе. Элементы массива обладают непрерывной нумерацией определённого диапазона.

В Visual Basic массивы определяются следующим образом: Dim myArray (10) As Long

Индекс (число в скобках) указывает размерность массива. Нижняя граница массива начинается с нуля. [0,1,2.....9,10].

Чтобы задать определённую размерность можно использовать зарезервированное слово To:

Dim myArray (5 To 10) As Long

Здесь определяется массив, размерность которого 6 элементов (5,6,7,8,9,10).

Общий синтаксис определения массива следующий:

Dim ИмяМассива{НомПерв1 To НомПосл1, НомПерв2 To НомПосл2, ...) [As [New] ИмяТипа]

Многомерные массивы

Массивы можно делать многомерными. Например, объявим массив - таблицу поля шахматной доски:

Dim chessTable (1 To 8, 1 To 8) As String

Этот массив представляет собой таблицу с восьми ячейками по вертикали и горизонтали.

Массивы переменной размерности (динамические)

Динамические массивы - это такие массивы, размерность которых может меняться в ходе работы программы.

Visual Basic предоставляет довольно мощные средства для работы с такими массивами. Опеределяется такой массив следующим образом:

Dim myArray () As Byte

В отличие от массивов статичных размеров, когда обращаться к элементам можно сразу после его объявления, к элементам динамического массива сразу обращаться нельзя, т.к. они ещё не инициализированы. Для начала нужно указать его новую размерность. Для это в VB есть оператор ReDim. Работает он следующим образом:

ReDim myArray (4)

Теперь массив myArray имеет одну размерность с индексами от 0 до 4 (т.е. всего 5 элементов). Теперь к такому массиву можно обращаться точно так же, как и к статичному. Если в дальнейшем возникнет необходимость снова изменить размерность массива, можно ещё раз использовать ReDim.

Оператор ReDim заново инициализирует (сбрасывает) все элементы массива к значению по умолчанию (как помните, для чисел - это 0, для строк ""). Но как же быть, если мы хотим изменить размеры массива, сохранив все старые элементы? Для сохранения значений элементов нужно после оператора ReDim поставить слово Preserve. Например:

ReDim Preserve myArray (3) ' сохраняем старые элементы

Заполнение одномерного массива случайными числами. Для этого используется функция

Rnd[(number)] - Возвращает Single значение, содержащее случайное число от 0 до 1.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim n, i As Integer, m(10) As Integer
        n = Val(InputBox("Введите размерность массива"))
        TextBox1.Text = " " + (n + 1).ToString + ":  "
        For i = 0 To n Step 1
            m(i) = Int((10 * Rnd())) ' Генерирует случайное число от 1 до 10
            TextBox1.Text = TextBox1.Text + " " + m(i).ToString
        Next i
    End Sub
```

Заполнение массива с клавиатуры

```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        Dim m, i, s As Integer, a(10) As Integer
        m = Val(InputBox("Введите размерность массива"))
        TextBox2.Text = " " + (m + 1).ToString + ":  "
        For i = 0 To m Step 1
            a(i) = Val(InputBox("Введите элемент массива"))
            TextBox2.Text = TextBox2.Text + " " + a(i).ToString
        Next i
End Sub
```

Заполнение двумерного массива.

```
    Dim s(5, 5) As Integer, i, j As Byte
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        Label2.Text = "5x5"
        For i = 1 To 5
            For j = 1 To 5
                s(i, j) = Int((21 * Rnd()))
                TextBox1.AppendText("    ")
                TextBox1.AppendText(s(i, j))
            Next
            TextBox1.Text = TextBox1.Text + vbNewLine
        Next
    End Sub
```

Вывод двумерного массива на форму в виде таблицы

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
  Dim m, i, j As Integer, a(10) As Integer
  Me.Text = "Двумерный массив случайных чисел"
  m = Val(InputBox("Введите размерность
массива"))
  Label1.Text = ""
  For i = 1 To m Step 1
    For j = 1 To m
      a(i) = Int((10 * Rnd()))
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
      Label1.Text = Label1.Text + " " + a(i).ToString
    Next
    Label1.Text = Label1.Text + " " + vbNewLine
  Next
End Sub
```

Задание.

1. Заполните одномерный массив с клавиатуры и сообщите его минимальный и максимальный элемент.
2. Заполните одномерный массив R(25) случайным образом и сообщите количество элементов массива, больших среднего арифметического значения элементов массива.
3. Заполните одномерный массив E(16) по формуле E(i) = 3*i*i-i+1 и заменить элементы с четным индексом на сумму всех элементов массива. Вывести новый массив на форму.
4. Заполнить массив действительных чисел, округлив каждый элемент до 0,01. Вывести массив на форму в столбец.
5. Заполнить автоматически одномерный массив T(15) и отсортировать его по убыванию. Вывести отсортированный массив на форму вместе с исходным.
6. Заполнить с клавиатуры двумерный массив A(3, 4) и отсортировать его строки по возрастанию.

Вывести отсортированный массив на форму вместе с исходным.

Основы работы со строками в Visual Basic

Тип данных String является последовательностью символов, каждый из которых, в свою очередь, является экземпляром типа данных Char. В этом разделе рассмотрены базовые понятия строк в Visual Basic.

```
Dim MyString As String
MyString = "This is an example of the String data type"
```

Любой текст, который присваивается переменной типа String, должен быть заключен в кавычки (""). Это означает, что кавычки в пределах строки не могут быть представлены кавычками.

В следующем примере демонстрируется, как правильно включать в строку кавычки:

```
' The value of myString is: He said, "Look at this example!"
myString = "He said, ""Look at this example!"""
```

В примере двойные кавычки перед словом Look становятся одной кавычкой в строке. Тройные кавычки в конце предложения представляют собой одинарную кавычку и символ окончания строки.

Символы в строках

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Строку можно представить как последовательность значений типа Char, кроме того, тип String имеет встроенные функции, которые позволяют манипулировать строками подобно массивам. Как и все массивы в .NET Framework, они являются массивами, в которых индексация ведется от нуля. Можно обратиться к определенному символу в строке с помощью свойства Chars, которое предоставляет механизм доступа к символу по занимаемой им позиции в строке. Например:

```
Dim myString As String = "ABCDE"
Dim myChar As Char
' The value of myChar is "D".
myChar = myString.Chars(3)
```

В приведенном выше примере свойство строки Chars возвращает четвертый символ в строке, D, и присваивает его переменной myChar. Размер строки можно получить, используя свойство Length. Если требуется выполнить несколько манипуляций со строкой, характерных для массивов, можно преобразовать ее в массив экземпляров Char, используя функцию строки ToCharArray. Например:

```
Dim myString As String = "abcdefghijklmnop"
Dim myArray As Char() = myString.ToCharArray
```

Переменная myArray теперь содержит массив значений Char, каждый из которых представляет символ из myString.

В отличие от других основных типов, тип String является ссылочным. Когда в функцию или подпрограмму в качестве аргумента передается ссылочный тип данных, на самом деле передается ссылка на адрес в памяти, по которому расположены данные, а не фактическое значение строки. Точно так же в предыдущем примере имя переменной остается без изменений, но оно указывает на другой экземпляр класса String, которому присвоено новое значение.

Смена регистра

При написании приложения, которое поддерживает данные, вводимые пользователем, неизвестно, какой регистр будет использовать пользователь для ввода данных. Поскольку методы, которые сравнивают строки и знаки, зависят от регистра, необходимо преобразовать регистр строк, введенных пользователями, перед их сравнением в значения констант. Регистр строки меняется легко. В следующей таблице описаны два метода изменения регистра. Для каждого метода имеются перегруженные варианты, учитывающие язык и региональные параметры.

Название метода	**Применение**
String.ToUpper	Преобразовывает регистр всех символов строки к верхнему регистру.
String.ToLower	Преобразовывает регистр всех символов строки к нижнему регистру.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
Dim MyString As String = "Hello World!"
Console.WriteLine(MyString.ToUpper())
' This example displays the following output:
'       HELLO WORLD!
```

Сокращение и удаление знаков

При разборе предложения на отдельные слова может оказаться, что некоторые слова на обоих концах содержат пробелы. В этом случае для удаления ряда пропусков или других знаков из указанного места строки используются методы сокращения из класса System.String. В следующей таблице представлены доступные методы сокращения.

Название метода	**Применение**
String.Trim ()	Удаление пробелов из начала и конца строки.
String.TrimEnd ()	Удаление знаков, указанных в массиве знаков, из конца строки.
String.TrimStart ()	Удаление знаков, указанных в массиве знаков, из начала строки.
String.Remove ()	Удаление указанного числа знаков из указанной позиции индекса в строке.

Метод String.Remove удаляет указанное число знаков, начиная с указанного места в существующей строке. В этом методе предполагается, что индексация начинается с нуля.
MyString.Remove(5,10)

```
Dim MyString As String = "Hello, World!"
Dim MyChar() As Char = {"r","o","W","l","d","!"," "}
Dim NewString As String = MyString.TrimEnd(MyChar)
Console.WriteLine(NewString)
```

В следующей таблице перечислено несколько полезных методов, которые возвращают строковые объекты.

Название метода	**Применение**
String.Format	Создание форматированной строки из набора объектов ввода.
String.Concat	Создание строк из двух и более строк.
String.Join	Создание новой строки с помощью объединения массива строк.
String.Insert	Создание новой строки с помощью вставки строки в указанную позицию существующей строки.
String.CopyTo	Копирование указанных знаков в строке в указанную позицию в массиве знаков.

Format

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Метод String.Format используется для создания форматированных строк и соединения строк, представляющих несколько объектов. Этот метод автоматически преобразует в строку любой переданный объект. Например, если приложению необходимо отобразить для пользователя значение Int32 и значение DateTime, легко создается строка для представления этих значений с помощью метода Format.

В следующем примере метод Format используется для создания строки, содержащей целочисленную переменную.

```
Dim numberOfFleas As Integer = 12
Dim miscInfo As String = String.Format("Your dog has {0} fleas. " & _
                                       "It is time to get a flea collar. " & _
                                       "The current universal date is: {1:u}.", _
                                       numberOfFleas, Date.Now)
Console.WriteLine(miscInfo)
```

В этом примере значение DateTime.Now отображает текущие дату и время в соответствии с языком и региональными параметрами, связанными с текущим потоком.

Concat

Метод String.Concat используется для простого создания нового объекта строки из двух или более существующих объектов. Он позволяет использовать независимый от языка способ сцепления строк. Этот метод принимает любой класс, производный от System.Object. В следующем примере создается строка из двух существующих объектов строки и знака разделения.

```
Dim helloString1 As String = "Hello"
Dim helloString2 As String = "World!"
Console.WriteLine(String.Concat(helloString1, " "c, helloString2))
' The example displays the following output:
'     Hello World!
```

Join

Метод String.Join создает новую строку из массива строк и разделительной строки. Этот метод полезен в случае необходимости сцепления нескольких строк и создания списка, отделенного, например, запятой.

В следующем примере используется пробел для привязки массива строк.

```
Dim words() As String = {"Hello", "and", "welcome", "to", "my" , "world!"}
Console.WriteLine(String.Join(" ", words))
' The example displays the following output:
'     Hello and welcome to my world!
```

Атрибут Insert

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Метод String.Insert создает новую строку с помощью вставки строки в указанную позицию другой строки. Этот метод использует индекс с отсчетом от нуля. В следующем примере строка вставляется в пятую позицию индекса MyString, и создается новая строка с этим значением.

```
Dim sentence As String = "Once a time."
 Console.WriteLine(sentence.Insert(4, " upon"))
 ' The example displays the following output:
 '     Once upon a time.
```

CopyTo

Метод String.CopyTo копирует часть строки в массив знаков. Можно указать начальный индекс строки и число копируемых знаков. Для копирования этим методом необходимы исходный индекс, массив знаков, индекс назначения и число знаков. Все индексы отсчитываются от нуля.

В следующем примере метод CopyTo используется для копирования знаков слова "Hello" из объекта строки в первую позицию индекса массива знаков.

```
Dim greeting As String = "Hello World!"
Dim charArray() As Char = {"W"c, "h"c, "e"c, "r"c, "e"c}
Console.WriteLine("The original character array: {0}", New String(charArray))
greeting.CopyTo(0, charArray,0 ,5)
Console.WriteLine("The new character array: {0}", New String(charArray))
' The example displays the following output:
'     The original character array: Where
'     The new character array: Hello
```

Задание

1. На форме разместить поля для ввода Имени, Фамилии и Отчества пользователя. После ввода данных, по нажатию на кнопку выдать сообщение «Уважаемый ФИО Вы наблюдаете работу программы по обработке срок)))»
2. Написать программу, запрашивающую текст, и выводящую на экран первое, последнее слово и количество слов в тексте с использованием CheckBox (`If CheckBox1.Checked = False Then I`)
3. На форме разместить поле для ввода пароля с целью регистрации. Правильный пароль запросить InputBox-ом и поместить в переменную строкового типа. Организовать проверку введенного пароля и выдачу сообщения в диалоговом окне: «Пароль введен не правильно» (не более 3 раз), «Вы так и не вспомнили пароль», «Пароль

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

введен правильно». Пароль не должен учитывать регистр, т.е. – Мама, маМа, Мама, мама – неразличимы.

4. Дана строка, содержащая по крайней мере один символ пробела. Вывести подстроку, расположенную между первым и вторым пробелом исходной строки. Если строка содержит только один пробел, то вывести пустую строку.
5. Составить программу, которая выводит на экран слово, удваивая каждую букву исходного слова.

Процедуры и функции в Visual Basic

Процедура Sub — это последовательность операторов Visual Basic, заключенных между операторами Sub и End Sub. Процедура Sub выполняет задачу и возвращает контроль коду вызова, но она не возвращает значения в код вызова.

При каждом вызове процедуры ее операторы выполняются, начиная с первого исполняемого оператора после оператора Sub и заканчивая первым из операторов End Sub, Exit Sub или Return.

Процедуру Sub можно определять в модулях, классах и структурах. По умолчанию она является Public, что означает, что ее можно вызывать из любого места в приложении, которое имеет доступ к модулю, классу или структуре, в котором она определена.

Для объявления процедуры Sub используется следующий синтаксис:

```
Sub Имя[(список_параметров)]
Statements (операторы) of the Sub procedure.
End Sub
```

Каждый параметр процедуры объявляется аналогично объявлению переменной: задается имя параметра и тип данных.

Для каждого параметра в списке параметров синтаксис выглядит следующим образом:

[Optional] [ByVal | ByRef] [ParamArray] имя_параметра As тип_данных

ByVal - Указывает способ передачи аргумента, при котором вызванная процедура может изменить значение переменной, содержащейся в аргументе вызывающего кода.

Передача аргументов по значению и по ссылке

В Visual Basic можно передать аргумент в процедуру по значению или по ссылке. Это называется *механизмом передачи*. Он определяет, может ли процедура изменять элемент программирования, содержащийся в аргументе кода вызова. Объявление процедуры определяет механизм передачи для каждого параметра путем указания ключевого слова ByVal (по значению) или ByRef (по ссылке).

Необходимо осторожно выбирать механизм передачи для каждого аргумента.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

- Защита. Наиболее важным критерием выбора между двумя механизмами передачи является требуемая степень доступности исходных переменных для изменения. Преимущество передачи аргумента ByRef заключается в том, что процедура может возвратить значение в вызывающий код через этот аргумент. Преимущество передачи аргумента ByVal состоит в том, что в этом случае переменная защищена от изменений, вносимых процедурой.
- Производительность. Хотя механизм передачи также может повлиять на производительность кода, разница практически не заметна. Единственным исключением является передача типа значения ByVal. В этом случае Visual Basic копирует все содержимое аргумента. Таким образом, большие аргументы типа значения (например структуру) целесообразнее передавать ByRef.

Объявление процедуры определяет механизм передачи для каждого параметра. Вызывающий код не может изменить механизм передачи ByVal, но если параметр объявлен с ключевым словом ByRef, то в вызывающем коде можно обеспечить передачу аргумента по ByVal, заключив при вызове имя аргумента в скобки.

По умолчанию в Visual Basic для передачи аргументов используется передача по значению. Чтобы сделать код легче читаемым, используйте ключевое слово ByVal. Включение ключевого слова ByVal или ByRef с каждым объявленным параметром — *это хороший стиль программирования*.

Когда передавать аргумент по значению

- Если элемент кода вызова, содержащийся в аргументе, является неизменяемым, объявите соответствующий параметр ByVal (Visual Basic). Никакой код не может изменить значение неизменяемого элемента.
- Если элемент является изменяемым, но процедура не должна изменить его значение, объявите параметр ByVal. Только вызывающий код может изменить значение изменяемого элемента, который передается по значению.

Когда передавать аргумент по ссылке

- Если процедуре необходимо изменить базовый элемент в коде вызова, объявите соответствующий параметр ByRef (Visual Basic).
- Если правильное выполнение кода зависит от изменения базового элемента в коде вызывающей процедуры, объявите параметр ByRef. Если параметр передается не по значению или код вызова переопределяет ByRef механизма передачи, заключив аргумент в круглые скобки, вызов процедуры может привести к непредсказуемым результатам.

Процедуры Function

Процедура Function — это последовательность операторов Visual Basic, заключенных между операторами Function и End Function. Процедура Function выполняет задачу и

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

возвращает управление вызвавшему ее коду. Вместе с управлением вызвавшему коду возвращается значение.

При каждом вызове процедуры ее инструкции выполняются, начиная с первого исполняемого оператора после оператора Function и заканчивая первым из операторов End Function, Exit Function или Return.

Можно определить процедуру Function в модуле, классе или структуре. По умолчанию эта процедура является глобальной (Public), что позволяет вызывать ее из любого места в приложении, из которого доступны модуль, класс или структура, в которых она определена.

Процедура Function может принимать аргументы, например константы, переменные или выражения, которые передаются ей вызывающим кодом.

Для объявления процедуры Function используется следующий синтаксис:

[модификаторы] Function имя_функции[(список_параметров)] As тип_возвращаемого_значения

' Statements of the Function procedure.

End Function

Тип данных

Каждой процедуре Function, как и любой переменной, назначается тип данных. Этот тип данных указывается предложением As оператора Function и определяет тип значения, которое функция будет возвращать вызывающему коду. Например:

```
Function yesterday() As Date
End Function
Function findSqrt(ByVal radicand As Single) As Single
End Function
```

Значение, которое процедура Function отправляет вызывающему коду, называется возвращаемым значением. Процедура возвращает значение одним из двух способов:

- Значение присваивается собственному имени функции в одном или нескольких операторах процедуры. Управление не возвращается вызывающей программе до тех пор, пока не будет выполнен оператор Exit Function или End Function. Это показано в приведенном ниже примере.

Function имя_функции[(список_параметров)] As тип_возвращаемого_значения

имя_функции = выражение

End Function

- Для определения возвращаемого значения используется оператор Return, после которого управление немедленно возвращается вызывающей программе. Это показано в приведенном ниже примере.

Function имя_функции[(список_параметров)] As тип_возвращаемого_значения

' The following statement immediately transfers control back to the calling code and returns the value of выражение.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Returnвыражение

End Function

Преимущество первого способа (с присвоением имени функции возвращаемого значения) заключается в том, что управление возвращается в вызывающую программу только после того, как будет выполнен оператор Exit Function или End Function. Это позволяет разработчику определить предварительное значение и изменять его впоследствии при необходимости.

Возвращение значения с помощью инструкции Return

1. Поместите инструкцию Return в точку, где завершена задача процедуры.
2. Дополните ключевое слово Return выражением, представляющее собой значение, которое следует вернуть вызывающему коду.
3. В одной и той же процедуре допускается несколько инструкций Return.

Возвращение значения с помощью Exit Function или End Function

1. Хотя бы в одном месте процедуры Function назначьте имени процедуры значение.
2. При выполнении инструкции Exit Function или инструкции End Function Visual Basic возвращает самое последнее значение, назначенное имени процедуры.
3. В одной и той же процедуре можно использовать несколько инструкций Exit Function, кроме того, инструкции Return и Exit Function можно сочетать.
4. В процедуре Function допускается только одна инструкция End Function.

Задание.

1. С использованием функции вычислить и сообщить минимальный элемент каждой строки массива m×n и максимальный элемент каждого столбца матрицы.

Решение

```
Public Class Form1
   Public m, n As Integer, d(20, 20) As Integer
   Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
      n = Val(InputBox("введите количество строк массива"))
      m = Val(InputBox("введите количество столбцов массива"))
      Label1.Text = Label1.Text + " размером " + n.ToString + " x " + m.ToString
      Call z_m()
   End Sub
   Public Sub z_m()
      Label2.Text = ""
      For i = 1 To n
         For j = 1 To m
            d(i, j) = Rnd() * 10
            Label2.Text = Label2.Text + "    " + d(i, j).ToString
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
        Next
        Label2.Text = Label2.Text + vbNewLine
      Next
    End Sub
    Public Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
      Call z_m()
    End Sub
    Function max(ByVal v(,) As Integer, ByVal mst As Integer) As Integer
      max = v(mst, 1)
      For i = 2 To m
        If d(mst, i) > max Then max = d(mst, i)
      Next
    End Function

    Private Sub RadioButton2_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RadioButton2.CheckedChanged
      If RadioButton2.Checked = True Then
        TextBox1.Text = ""
        For j = 1 To n
          TextBox1.Text = TextBox1.Text + "   " + max(d, j).ToString
        Next
      End If
    End Sub

    Private Sub RadioButton3_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RadioButton3.CheckedChanged
      If RadioButton3.Checked = True Then
        TextBox1.Text = ""
        Label2.Text = ""
      End If
    End Sub

End Class
```

2. В готовом коде запрограммируйте работу элемента RadioButton по отысканию минимальных значений столбцов матрицы.
3. С использованием функции заменить в строке все буквы «а» на «+».

Обработка файловых структур в Visual Basic

Для работы с файлами в vb-2010 используется StreamWriter и StreamReader.

```
Imports System
Imports System.IO  ' работа с системой
Public Class Form1
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim str$ = TextBox1.Text 'ну это понятно
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
    Dim strimwrit As New StreamWriter("c:\tekst.txt", True) ' создаем новый объект зада-
ем путь, True это как аппенд в ВБ-6
    strimwrit.Write(str) ' записиваем
    strimwrit.Close()
  End Sub
  Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
    Dim strimrid As New StreamReader("c:\tekst.txt",
System.Text.Encoding.GetEncoding(1251))
    TextBox2.Text = strimrid.ReadToEnd ' читать до конца файла , тоесть выводится все
содержимое файла
    strimrid.Close()
  End Sub
End Class
```

Для работы с файлами и директориями используем пространство имен *System.IO*
Пространство имен *System.IO* содержит типы, позволяющие осуществлять чтение и запись в файлы и потоки данных, а также типы для базовой поддержки файлов и папок.
Действиями с файлами :

```
IO.File.Create("D:/File.txt") ' Создаем новый файл
IO.File.Delete("D:/File.txt") ' Удаляем файл
IO.File.Copy("D:File.txt", "C:/File.txt") ' Копируем файл из диска D в C
IO.File.Move("D:File.txt", "D:/Работа/File1.txt") ' Перемещаем и изменяем имя файла
MsgBox(IO.File.GetCreationTime("D:/File.txt")) ' Дата создания файла
MsgBox(IO.File.GetAttributes("D:/File.txt")) ' Тип файла
Dim Файл As New IO.FileInfo("D:/File.txt")
MsgBox(Файл.Length & " байт") ' Размер файла в байтах, в следующих уроках мы бу-
дем узнавать размеры файлов в килобайтах и мегабайтах
```

Существует ли файл :

```
If IO.File.Exists("D:/File.txt") Then
MsgBox("Файл существует")
Else
MsgBox("Файл не существует")
End If
```

Сведения о файлах :

```
MsgBox(IO.Path.GetExtension("D:/File.txt")) ' Расширение файла результат будет .txt
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

MsgBox(IO.Path.GetFileName("D:/File.txt")) ' Имя файла без пути результат будет File.txt

MsgBox(IO.Path.GetFileNameWithoutExtension("D:/File.txt")) ' Имя файла без расширения результат будет File

MsgBox(IO.Path.GetFullPath("D:/File.txt")) ' Полное имя файла с путем результат будет D:/File.txt

MsgBox(IO.Path.GetPathRoot("D:/File.txt")) ' Имя корневого каталога результат будет D

MsgBox(IO.Directory.GetCurrentDirectory) ' Получаем путь каталога где находиться программа

MsgBox(Application.ExecutablePath) ' Получаем путь и имя исполняемого файла

Запись в файл(способов очень много, но я напишу всего два) :

Добавьте на форму текстовое поле TextBox, в него мы будем вводить текст.

IO.File.WriteAllText("D:/File.txt", TextBox1.Text) ' Запись текста из TextBox1 в File.txt

' Другой способ

FileOpen(1, "D:/File.txt", OpenMode.Output)

Print(1, TextBox1.Text)

FileClose(1)

Чтение текста из файла :

TextBox1.Text = My.Computer.FileSystem.ReadAllText("D:/File.txt", System.Text.Encoding.GetEncoding(1251)) ' Считываем все строки в TextBox1 с кодировкой, указывайте такую кодировку и не будет проблем с отоброжение.

TextBox1.Text = IO.File.ReadAllText("D:/File.txt", System.Text.Encoding.GetEncoding(1251)) ' Тоже самое

Задание.

Дан файл целых чисел. Определить максимальный и минимальный из этих чисел. Дописать в файл их разность.

Панель элементов: диалоговые окна

Классы всех стандартных диалоговых окон являются производными от абстрактного класса CommonDialog. Важнейший метод этого класса ShowDialog предназначен для отображения диалогового окна. По возвращаемому значению этого метода можно узнать, какую кнопку нажал пользователь в диалоговом окне — ОК или Cancel. Метод ShowDialog имеет следующий синтаксис: Public Function ShowDialogO As DialogResult

При проверке возвращаемое значение сравнивается с константами DialogResult .OK и DialogResult.Cancel.

ColorDialog

При использовании диалогового окна выбора цвета (Col orDialog) программа обычно запрашивает свойство Color и назначает его свойству ForeColor или BackColor элемента или формы.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

FontDialog

Диалоговое окно выбора шрифта (FontDialog) хорошо знакомо всем пользователям текстовых редакторов Windows. В свойстве Font объекта диалогового окна возвращается объект Font, обычно назначаемый свойству Font элемента или формы.

FontDialog

Диалоговое окно выбора шрифта (FontDialog) хорошо знакомо всем пользователям текстовых редакторов Windows. В свойстве Font объекта диалогового окна возвращается объект Font, обычно назначаемый свойству Font элемента или формы.

Создание диалоговых окон

Чтобы вывести собственное диалоговое окно, создайте форму, задайте ее свойствам ControlBox, MinimizeBox и MaximizeBox значение False, а свойству Modal — значение True. Форму следует выводить методом ShowDialog в режиме модального диалогового окна. Если при этом задать свойство TopMost равным True, диалоговое окно будет располагаться поверх всех окон на экране

После вызова ShowDialog программа может узнать, какая кнопка была нажата на форме, при помощи свойства Dial ogResult кнопки или самой формы (нажатие кнопки с заданным свойством DialogResul t приводит к автоматическому закрытию формы, на которой эта кнопка находится).

Пример

```
Public Class Form1
   Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
      Dim Bmp As Bitmap
      If OpenFileDialog1.ShowDialog() = DialogResult.OK Then

         PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
         Bmp = New Bitmap(OpenFileDialog1.FileName)
         PictureBox1.Image = CType(Bmp, Image)
      End If
   End Sub
   Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
      Dim myDialog As New ColorDialog()
      Dim Temp As Color = Button2.BackColor
      If myDialog.ShowDialog() = DialogResult.OK Then
         Temp = myDialog.Color
         Me.BackColor = Temp
         Button2.BackColor = Temp
         Button1.BackColor = Color.Bisque
         Button2.ForeColor = Color.Aqua
      End If
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
    End Sub
End Class
```

Чтобы выбрать свойства шрифта с помощью компонента FontDialog

1. Отобразите диалоговое окно с помощью метода ShowDialog.
2. С помощью свойства DialogResult определите, как было закрыто диалоговое окно.
3. Для задания требуемого шрифта воспользуйтесь свойством Font.

Задание.
Добавьте кнопку на форму для изменения оформления шрифта на форме (на всех её элементах).

Создание меню в Microsoft Visual Basic 2010

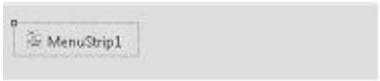
Добавление Меню с помощью элемента MenuStrip

Элемент управления MenuStrip - инструмент, который добавляет меню к вашим программам, которые вы можете настроить с установкой свойств в окне Свойств. С MenuStrip, вы можете добавить новые меню, изменить и переупорядочить существующее меню, и удалять старые меню. Вы можете также создать меню стандартной конфигурации автоматически, и вы можете дополнить свои меню специальными эффектами, как например ключи доступа, отметка элемента и горячие клавиши . Но MenuStrip создает только видимую часть меню и команд . Вам все еще нужно написать процедуры - события, которые обрабатывают выбранный пункт меню. Используя MenuStrip создайте меню Часы, содержащее команды, которые показывают текущую дату и время.

Создание меню

1. Запустите Visual Studio .
2. На меню Файл, щелкают Новый Проект с именем MyMenu .
3. Щелкните MenuStrip по вкладке Панелей инструментов Menus & Toolbars tab, а затем перетащите элемент на вашу форму .

Не задумывайтесь о расположении - Визуал Студия переместит элемент управления и переопределит размер его автоматически . Ваша форма выглядит вот так:



Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Объект menu strip не появляется на форме, а ниже неё . Невидимые объекты, такие как меню и таймеры, показываются в Интегрированной Среде Разработки в отдельной области называемой область элементов, и вы можете выбрать их, устанавливать их свойства, или удалять их из этой области .

В добавление к объекту полосы меню в область элементов, Visual Studio показывает визуальный

представление меню, которое вы создали в верхней части формы . Таким образом помечена облать меню, чтобы вы щелкнули там и ввели название вашего меню . После того, как вы вводите первое название меню, вы можете ввести заголовки меню нижнего уровня и другие имена меню нажимая стрелку и печатая дополнительные имена . Вы можете вернуться к этому встроенному меню позже и отредактировать.

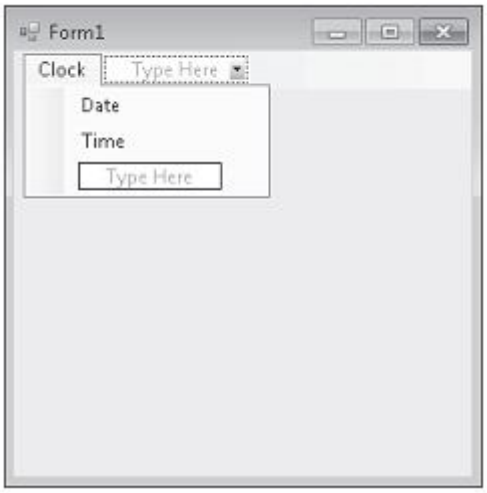
5. Щелкните по обозначению меню и напишите Часы, а затем нажмите ENTER .

Слово Часы заносится как имя вашего первого меню, и две дополнительные отметки Пищи Здесь Type Here появляются, в которых вы можете создать меню нижнего уровня.

6. Напишите Дата, чтобы создать команду Даты в меню Часы, а затем нажмите ENTER.

7. Напишите Время, чтобы создать команду Времени в меню, а затем нажмите ENTER.

Вы сейчас имеете меню Часы с двумя командами в нем: Дата и Время .



8. Щелкните по форме, чтобы закрыть Проектировщика Меню .

Проектировщик Меню закрывается, и ваша форма открывается в IDE с новым меню Часы.

Добавление Ключей доступа к Командам Меню

В большинстве приложений Вы можете получить доступ и выполнить команды меню, используя клавиатуру. что Вы прессу, чтобы выполнить команду в открытом меню называют клавишами доступа. Вы можете задать клавишу доступа пункта меню, для этого подчеркнуть буквы команды на меню. Добавить ключ доступа к меню можно активизируя Проектировщика Меню, и затем напечатать амперсанд (&) перед соответствующей буквой в имени меню. Когда Вы открываете меню во время выполнения программы, Ваша программа автоматически показывает ключ доступа.

Соглашения меню

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

В соответствии с соглашением, каждая команда названия и меню в Приложении Windows имеет начальную заглавную букву. Файл и Редактировать (Правка), часто первые два пункта меню, а Помощь обычно - последнее. Другие общие имена меню - Просмотр, Формат, и Окно. Независимо от того меню или команды Вы используете в своих приложениях, заботитесь, чтобы работа с ними была ясной и последовательной . Меню и команды должны быть удобными в использовании и иметь общее насколько возможно с подобными пунктами в других основанных Windows приложениях.

Создание пунктов меню, использует следующие принципы:

- Использование короткие, определенные заголовки, состоящие из одного или двух слов
- Назначают каждому пункту меню ключ доступа. Используйте первую букву пункта, если возможно, или клавиша доступа, которая обычно назначается.
- Пункты меню на том же самом уровне должен иметь уникальную клавишу доступа.
- Если команда используется в качестве указания вкл\выкл, поместите галочку слева от пункта, когда команда активна. Вы можете добавить галочку, устанавливая свойство Checked для команды меню в True в окне Properties (Свойств).
- Место с многоточием (...) после команды меню, которая требует, чтобы пользователь указал больше информации до выполнения команды. Троеточие указывает на то, что Вы откроете диалоговое окно, если выберет е этот пункт.

Добавить ключ доступа к меню Clock.

Запрограммируйте команды меню.

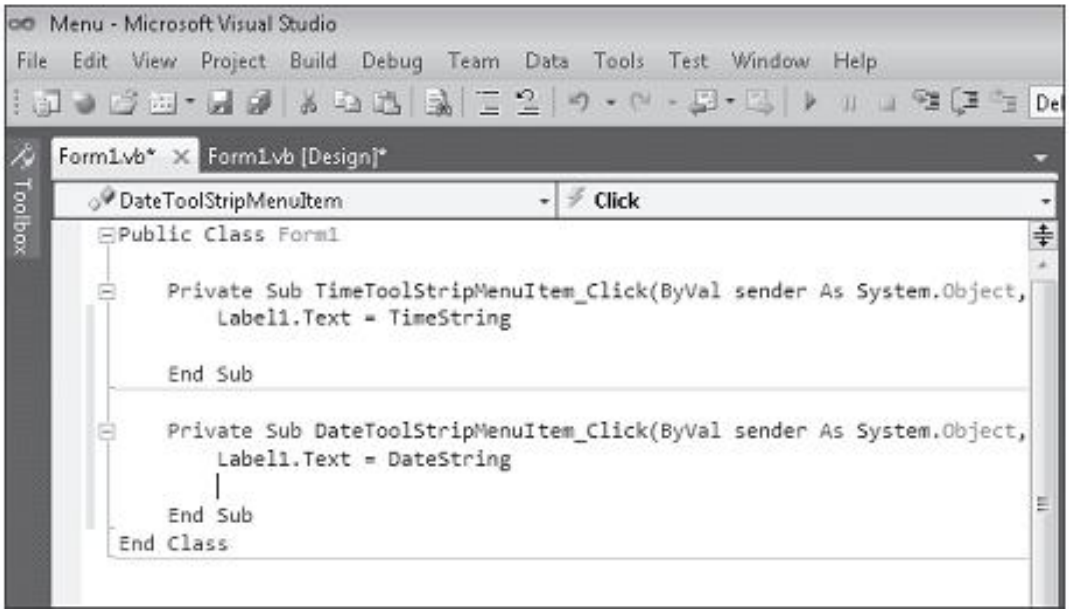


Таблица «Системные свойства Часов»

Свойства и методы	Описание
TimeString	Это свойство устанавливает или возвращает текущее

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

	время по системным часам.
DateString	Это свойство устанавливает или возвращает текущую дату по системным часам.
Now	Это свойство возвращает закодированное значение, представляющее текущую дату и время.
Hour (date)	Это свойство извлекает часть часа указанного в значении дата/время (0 до 23).
Minute (date)	Это свойство извлекает часть указанного значения дата/время (0 до 59).
Second (date)	Этот метод извлекает вторую часть указанной даты/время (0 до 59).
Month (date)	Этот метод извлекает целое число, представляющее месяц (1 - 12).
Year (date)	Этот метод извлекает часть года указанного в дата/время
Weekday (date)	Этот метод извлекает целое число, представляющее день недели (1, воскресенье, 2 понедельник, и так далее).

Предоставить контекстную справку для элементов управления с помощью класса HelpProvider

Следующий пример описывает, как использовать класс **HelpProvider** для отображения всплывающего окна справки для элементов управления. Используйте метод **SetHelpString** класса **HelpProvider** для задания текста всплывающей подсказки для элементов управления. Нажмите кнопку **Справка** в строке заголовка формы Form1 и выберите элемент управления, отображается строка справки.

1. Добавьте в форму Form1 два элемента управления **TextBox** .
2. Добавьте следующий код в обработчик событий **Form1_Load** .
3. 'Set the Help string for the TextBox control on the form.
4. HelpProvider1.SetHelpString(Me.TextBox1, "Enter your UserName")
5. HelpProvider1.SetHelpString(Me.TextBox2, "Enter your Password in this TextBox")
6. 'Set the Help string for the Button control on the form. HelpProvider1.SetHelpString(Me.Button1, "Click submit after you type your UserName and Password")

7. Щелкните правой кнопкой мыши **форму Form1**и выберите команду **Свойства**.
8. В диалоговом окне **Свойства** установите для **Свойства HelpButton** значение **True**.
9. Значение **False MaximizeBox** .
10. **MinimizeBox** задать значение **False**.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Создание справочных файлов.

Надо ли говорить, насколько важна справочная система для приложения? Она является одной из важнейших составляющих успеха приложения! Разумеется, сама программа должна иметь интуитивно понятный интерфейс, чтобы пользователь мог догадаться для чего нужна та или иная кнопка. Но справка конечно же необходима.

Алгоритм создания файла справки.

1. Запустите Help Workshop и выполните команду **File > New**.

Далее, из появившегося списка выберите строку **Help Contents** и нажмите кнопку **OK**.

Так мы создали файл содержания.

2. Теперь, необходимо задать имя справочного файла. Пусть это будет, например, **MyHelp.hlp**.

Введите это имя в поле **Default Filename**. А в поле **Default Title** введите строку **Создание справочных файлов**

3. Создание структуры файла в Help Workshop, чем-то напоминает создание меню в Visual Basic.

Добавление разделов осуществляется кнопками **Add Above...** и **Add Below...**

Нажмите кнопку **Add Above...**, установите переключатель **Heading** и в поле **Title** введите строчку **Создание справочных файлов**.

4. Далее, нажмите кнопку **Add Below...**, установите переключатель **Heading** и в поле **Title** введите строчку **Создаем файл содержания**.
5. Теперь, нажмите кнопку **Move Right**, для того, чтобы поставить созданный только что нами раздел ниже по иерархии, чем первый (я же сказал, что это очень похоже на создание меню в VB).
6. Снова жмем кнопку **Add Below...**, но на этот раз оставим переключатель в покое (т.е оставим выбранным**Topic**). Далее, в поле Title введите строку **Создать файл содержания**, а в поле TopicID - строку **IDH_CreateContentsFile**.

7. Теперь, таким же образом добавляем следующие разделы:

Цитата:

Создаем файлы разделов

- Создать файлы разделов - IDH_WriteTopics

- Написать текст, добавить сноски, и сохранить файл - IDH_WriteAddSave

Создаем файл проекта

- Создать файл проекта, задать параметры проекта - IDH_CreateSet

- Откомпилировать проект - IDH_Compile

- Протестировать справочный файл - IDH_Test

Выберите команду **File > Save** и сохраните файл под именем **MyHelp.cnt**.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Итак, файл содержания создан. Сверните окно программы. Теперь пришло время создать файл разделов. Этот файл представляет собой RTF-документ и содержит текст, графику, макросы и ссылки, которые будут воспроизводиться в справочном файле.

1. Для создания файла разделов, нам понадобится Microsoft Word. Запустите Word и создайте в нем новый документ.

2. Введите в первую строку документа следующий текст: **Создать файл содержания**.

3. Введите ниже:

Цитата:

Создание справочного файла начинается с создания файла содержания. Help Workshop позволяет производить это графически, что существенно облегчает задачу. Создание разделов напоминает работу с редактором меню VB.

1. Запустите Help Workshop и выполните команду File > New.

2. Выберите строку Help Contents и нажмите кнопку OK.

3. Задайте имя справочного файла в поле Default Filename.

4. Добавьте необходимые разделы с помощью кнопок Add Above... и Add Below...

4. Теперь, необходимо добавить сноски. Установите курсор перед заголовком **Создать файл содержания**.

5. Выберите команду **Вставка > Ссылка > Сноска** и в поле **Другой** введите знак #. Нажмите **OK**

6. Редактор переключился на текст сноски. Введите строку **IDH_CreateContentsFile**. Как вы думаете, что это? Идетификатор данного раздела.

7. Теперь, надо задать название раздела. Установите курсор между знаком фунта и заголовком в верхней часте раздела.

На этот раз, добавьте сноску **$**, а в качестве текста сноски, введите строку **Создать файл содержания**.

8. Последняя сноска - + с текстом **auto**.

Теперь, по такому же образцу добавьте следующие разделы (каждый раздел должен начинаться с новой страницы).

После того, как весь текст будет введен, можно сохранить файл. Сохраните его с именем **MyHelp.rtf**.

Файл разделов создан! Теперь, пришло время создать файл проекта.

1. Разверните окно программы и выполните команду **File > New**

Далее, из появившегося списка выберите строку **Help Project** и нажмите кнопку **OK**.

Так мы создали файл проекта.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

2. Когда программа предложит сохранить проект, введите имя **MyHelp.hpj** и нажмите кнопку **OK**.

3. Нажмите кнопку **Options**.

4. В открывшемся окне, в поле **Help Title**, введите строку **Создание справочного файла**.

5. Перейдите на вкладку **Compression**, установите переключатель **Custom**, а потом - флажок **Hall Compression**.

6. Теперь, перейдите на вкладку **Files** и введите в поле **Help File** строку **MyHelp.hlp**.

7. Нажмите кнопку **Change**, находящуюся рядом с полем **RTF Files**. Теперь, в открывшемся окне нажмите кнопку **Add** и выберите файл **MyHelp.rtf**.

8. Нажмите кнопку **Browse**, рядом с полем **Contents**. В открывшемся окне, выберите файл **MyHelp.cnt**.

9. Закройте окно **Options** кнопкой **OK**

10. Нажмите кнопку **Windows** и в поле **Create a Window Named**, окрывшегося окна, введите строку **Main**. Нажмите кнопку **OK**.

11. После этого, откроется окно **Windows Properties**. Кликните вкладку **Buttons** и установите флажок **Browse**. Нажмите **OK**.

12. Нажмите кнопку **Map**.

13. Для каждого из разделов, перечисленных ниже, нажмите кнопку **Add** и введите соответствующий идентификатор в поле **Topic ID**, а также идентификатор контекста в поле **Mapped Numeric Value**.

13. Нажмите **OK**.

14. Ну, и наконец-то, нажмите кнопку **Save and Compile**. На время компиляции окно программы сворачивается. После завершения компиляции, окно восстанавливается и в нем отображаются результаты компиляции.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

РАЗДЕЛ 4 КОНТРОЛЬ ЗНАНИЙ
РАЗДЕЛ 4. КОНТРОЛЬ ЗНАНИЙ

Текущий контроль знаний и умений

Текущий контроль предусматривает систематическую проверку качества знаний и умений студентов по всем изучаемым в данном семестре дисциплинам. Формы текущего контроля знаний и умений отражены в календарно-тематических планах преподавателей.

Контроль знаний и умений студентов - один из важнейших элементов учебного процесса. От его правильной организации во многом зависит эффективность управления учебно-воспитательным процессом и качество подготовки специалиста.

Интенсивность и регулярность работы студентов зависит, главным образом, от частоты и регулярности проведения контроля. От этого же зависит и длительность сохранения в памяти усвоенных знаний. При организации и проведении контроля следует учитывать определенные свойства памяти.

Организация проведения контроля включает следующие этапы:

- разработка целей контроля;
- разработка содержания контрольных заданий;
- выбор организационных форм контроля, адекватных целям и содержанию;
- разработка методов контроля;
- разработка критериев оценки, результатов выполнения контрольных заданий и требований к их анализу.

Назначение контроля и предъявляемые к нему требования

Текущий контроль – оценка знаний и умений обучаемых.

Наиболее важная цель контроля - мотивирование регулярной и целенаправленной работы студентов. Интенсивность и регулярность работы студентов зависит, главным образом, от частоты и регулярности проведения контроля. От этого же зависит и длительность сохранения в памяти усвоенных знаний.

Чем чаще проходит контроль, тем лучше студент адаптируется к контрольной процедуре: его нервное напряжение значительно снижается.

Другая цель контроля - дать возможность студенту сопоставить свою работу с требованиями преподавателя, выяснить недочеты, недоработки и внести, если нужно, необходимые коррективы в свою подготовку.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Контроль знаний и умений студентов выполняет в процессе обучения следующие функции: проверочную (диагностическую), обучающую, развивающую, воспитательную и методическую.

Проверочная (диагностическая)	Данные контроля констатируют не только результаты и оценку учебной деятельности отдельных студентов и преподавателей, но и состояние учебно-воспитательной работы всего учебного заведения, подсказывают меры, необходимые для его совершенствования.
Обучающая	В ходе проведения контрольных заданий происходят повторение и закрепление, совершенствование приобретенных ранее знаний путем их уточнения и дополнения. Контроль способствует формированию умений и навыков, рационально организовывать учебную деятельность.
Развивающая	Контроль содействует развитию внимания, памяти, мышления, воображения, умению сопоставить и систематизировать имеющиеся знания, делать выводы, обобщения, приводить доказательства.
Воспитательная	Контроль дисциплинирует студента, воспитывает у него чувство ответственности за свою работу, приучает к систематическому труду, стимулирует регулярную активную учебную деятельность.
Методическая	Контроль позволяет оценить методы преподавателя, увидеть его сильные и слабые стороны, выбрать оптимальные варианты обучающей деятельности (контроль учит не только студента, но и преподавателя).

Для организации текущего контроля в УМК предусмотрены контрольные вопросы по темам, самостоятельные работы, а также на каждой лабораторной работе каждый студент получает оценку за выполненные задания в соответствии с указанными баллами в каждом задании на лабораторную работу.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Проверка теоретических знаний по теме «Процедуры и функции»

По предложенной программе ответьте на вопросы

1. Какое значение по умолчанию имеет параметр: y, i, T1 ?
2. В какой подпрограмме переменные передаются по значению?
3. Почему после выполнения функции «Факториал» меняется значение глобальной переменной y?
4. Укажите для подпрограммы «Сумма» параметры и аргументы.
5. Перечислите обязательные и необязательные параметры для процедур программного кода

По предложенной программе

```
Public Class Form1
    Dim y As Integer
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        Me.Text = "ПОДПРОГРАММЫ"
        Label2.Text = "Результаты работы подпрограммы"
        y = Val(InputBox("введите число"))
    End Sub
    Function factorial(ByRef n As Integer) As Integer
        factorial = 1
        For i As Integer = 1 To n
            factorial = i * factorial
        Next i
        n = 2
    End Function
    Function symma(ByVal n As Integer, Optional ByRef T1 As Boolean = True) As Integer
        Dim s As Integer
        s = 0
        For i As Integer = 0 To n
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

```
            s = i + s
        Next i
        Return s
    End Function
    Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CheckBox1.CheckedChanged
        Label1.Text = vbNewLine
        Label1.Text = "сумма " + y.ToString + "=" + symma(y).ToString
    End Sub
    Private Sub CheckBox2_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CheckBox2.CheckedChanged
        If CheckBox2.Checked = True Then
            Label1.Text = vbNewLine
            Label1.Text = "Факториал  " + y.ToString + "= " + factorial(y).ToString
        End If
    End Sub
End Class
```

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

РУБЕЖНЫЙ КОНТРОЛЬ

В качестве рубежного контроля по дисциплине выступают контрольные работы по темам и ежемесячная аттестация по дисциплине.

Контрольная работа №1

Вариант № 1

1. Создать файл wares, содержащий сведения об экспортируемых товарах (10 записей): указывается наименование товара, страна, импортирующая товар, и объём поставляемой партии в штуках. Найти страны, в которые экспортируется данный товар, и общий объём его экспорта.
2. Создать текстовый файл f, состоящий из 2-3 строк текста произвольной длины. Перезаписать содержание файла f в файл h, так, чтобы в файле h все строки были по 20 символов (кроме последней) и в начале каждой строки стоял ее номер.

Вариант № 2

1. Создать типизированный файл book, содержащий сведения о книгах(10 записей): фамилия автора, название и год издания. Найти названия книг данного автора, изданных после 1999 года.

2. Содать текстовый файл, состоящий из 2-3 предложений. Определить количество слов в нем

Вариант № 3

1. Создать типизированный файл toys, содержащий сведения об игрушках (10 записей): указывается название игрушки (например, кукла, кубики, мяч, конструктор), её стоимость в гривнах и возрастные границы детей, для которых игрушка предназначена (например, для детей от двух до пяти лет). Узнать цену самого дорогого конструктора;
2. Создать текстовый файл, состоящий из 2-3 предложений. Определить, есть ли в нем английские слова и сколько.

Вариант № 4

1. Создать типизированный файл avto, содержащий сведения об автомобилях (10 записей): марка автомобиля, его номер и фамилия владельца. Найти фамилии владельцев и номера автомобилей данной марки.
2. Создать текстовый файл f, содержащий любое арифметическое выражение, например, (2x+3y)*((x+y)-3(xy-4)+5). Определить, совпадает ли в нем количество открывающихся и закрывающихся скобок, если нет, то каких не хватает и сколько.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Вариант № 5

1. Создать типизированный файл f, компонентами которого являются целые случайные числа. Записать в файл g все четные числа файла из f, а в файл h – все нечетные. Порядок следования чисел сохраняется.
2. Создать текстовый файл f, состоящий из произвольных слов и чисел, записанных в одну строку. Перезаписать содержание файла f в файл h, так, чтобы в файле h были записаны только числа, разделенные пробелом.

Вариант № 6

1. Создать текстовые файлы f и g, компонентами которых являются случайные целые числа. Записать в файл h сначала компоненты файла f, а затем компоненты файла g с сохранением порядка.
2. Создать типизированный файл avto, содержащий сведения об автомобилях (10 записей): марка автомобиля, его номер и фамилия владельца. Найти фамилии владельцев и номера автомобилей данной марки.

Контрольная работа №2

Вариант 1

Создать модуль, содержащий функции одного компонента ряда и всего ряда целиком и процедуру расчета погрешности.

g(x,i)=-1/((i+3)*x^i)

f(x)=1 + Сумма(буква сигма,от i=1 до бесконечности)g(x,i)

-создать программу выводящую на экран сумму элементов;

-создать программу, выводящую заданное число элементов в виде таблицы;

-создать программу, выводящую погрешность

Вариант 2

Реализовать в виде модуля набор подпрограмм для выполнения следующих операций с квадратными матрицами: 1) сложение двух матриц; 2) умножение одной матрицы на другую; 3) нахождение транспонированной матрицы; 4) вычисление определителя матрицы.

Матрицу описать следующим образом:

```
Const NMax = 10;
Type Matrica = Array [1..NMax, 1..Nmax] Of Real;
```

Используя этот модуль, решить следующие задачи:

1. Решить систему линейных уравнений *N*-го порядка (2<=*N*<=10) методом Крамера.
2. Задан массив величин типа Matrica. Отсортировать этот массив в порядке возрастания значений определителей матриц.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

РАЗДЕЛ 5 МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ

Методические рекомендации для студентов по организации самостоятельной работы

Введение

В современный период востребованы высокий уровень знаний, социальная мобильность, профессионализм специалистов, готовность к самообразованию и самосовершенствованию. В связи с этим должны измениться подходы к планированию, организации учебно-воспитательной работы, в том числе и самостоятельной работы студентов. Прежде всего, это касается изменения характера и содержания учебного процесса, переноса акцента на самостоятельный вид деятельности, который является не просто самоцелью, а средством достижения глубоких и прочных знаний, инструментом формирования у студентов активности и самостоятельности.

Целью методических рекомендаций является повышение эффективности учебного процесса, в том числе благодаря самостоятельной работе, в которой студент становится активным субъектом обучения, что означает:

- способность занимать в обучении активную позицию;
- готовность мобилизовать интеллектуальные и волевые усилия для достижения учебных целей;
- умение проектировать, планировать и прогнозировать учебную деятельность;
- привычку инициировать свою познавательную деятельность на основе внутренней положительной мотивации;
- осознание своих потенциальных учебных возможностей и психологическую готовность составить программу действий по саморазвитию.

Виды самостоятельной работы студентов

Репродуктивная самостоятельная работа	Самостоятельное прочтение, просмотр, конспектирование учебной литературы, прослушивание лекций, магнитофонных записей, заучивание, пересказ, запоминание, Интернет-ресурсы, повторение учебного материала и др.
Познавательно-поисковая самостоятельная работа	Подготовка сообщений, докладов, выступлений на семинарских и практических занятиях, подбор литературы по дисциплинарным проблемам, написание рефератов, контрольных, курсовых работ и др.
Творческая самостоятельная работа	Написание рефератов, научных статей, участие в научно-исследовательской работе, подготовка дипломной работы (проекта). Выполнение специальных заданий и др., участие в студенческой научной конференции. Организация и контроль самостоятельной работы

Для успешного выполнения самостоятельной работы студентов необходимо планирование и контроль со стороны преподавателей.

Аудиторная самостоятельная работа выполняется студентами на лекциях, лабораторных занятиях, и, следовательно, преподаватель должен заранее выстроить систему самостоятельной работы, учитывая все ее формы, цели, отбирая учебную и научную информацию и средства (методических) коммуникаций, продумывая роль студента в этом процессе и свое участие в нем.

Вопросы для самостоятельной работы студентов, указанные в рабочей программе дисциплины, предлагаются преподавателями в начале изучения дисциплины. Студенты имеют право выбирать дополнительно интересующие их темы для самостоятельной работы.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1. Основные мотивы самостоятельной работы студентов

Активная самостоятельная работа студентов возможна только при наличии серьезной и устойчивой мотивации. Самый сильный мотивирующий фактор – подготовка к дальнейшей эффективной профессиональной деятельности. Среди внутренних факторов, способствующих активизации самостоятельной работы выделяют следующие:

1. *Полезность* выполняемой работ заключается в том, что результаты самостоятельной работы могут быть использованы на семинарских и практических занятиях, лабораторном практикуме, при подготовке публикации. Другим вариантом использования фактора полезности является активное применение результатов работы в профессиональной подготовке. Так, например, если студент получил задание на дипломную работу на одном из младших курсов, он может выполнять самостоятельные задания по ряду дисциплин гуманитарного и социально-экономического, естественно-научного и общепрофессионального циклов дисциплин, которые затем войдут в его выпускную квалификационную работу.
2. *Творческая деятельность.* Это может быть участие в научно-исследовательской, опытно-конструкторской или методической работе, проводимой на той или иной кафедре.
3. *Участие* в олимпиадах по учебным дисциплинам, конкурсам научно-исследовательских или прикладных работ и т.д.
4. *Участие* в научно – практических конференциях.
5. *Подготовка публикаций* для сборников тезисов и докладов научно-практических конференций, журналов, учебных пособий и т.д.
6. *Участие в грантовых конкурсах.*

2. Способы самостоятельной работы при чтении учебной и научной литературы

При остром недостатке времени у студентов встает вопрос об оптимизации обучения, то есть такой организации учебного процесса, которая обеспечила бы условия для продуктивного самообучения и самовоспитания. Важнейшую роль здесь играет овладение способами самостоятельной работы. Речь идет о том, что чтобы прежде всего научиться рациональному использованию времени при работе с книгой.

В этом особенно нуждаются первокурсники, которые еще недостаточно владеют навыками умственной деятельности, обеспечивающей успешное обучение.

Начинать самостоятельные занятия следует с первых же дней учебы в вузе. Первые дни семестра важны, чтобы включиться в работу, установить определенный равномерный ритм на весь семестр. Чтобы выполнить весь объем самостоятельной работы, необходимо заниматься самостоятельно по 4-5 часов ежедневно, кроме выходных дней.

Под ритмом работы понимают ежедневные занятия в одни и те же часы, при чередовании их с перерывами для отдыха. Вначале для организации ритмичной работы требуется сознательное напряжение воли, затем принуждение снимается, возникает привычка и работа становиться потребностью.

Ритмичная работа позволяет студенту заниматься много, не уставая, не снижая производительности и не перегружая себя. Для этого необходимо:

- Сменять один вид работы другим, что позволяет сохранять высокую работоспособность, поскольку при однообразных видах занятий человек утомляется больше, чем при работе разного характера.
- Заниматься несколькими предметами в один и тот же день не всегда целесообразно, поскольку при каждом переходе нужно вновь концентрировать внимание и затрачивать время.
- Умение сосредотачиваться – необходимое условие для умственного труда, иначе работа оказывается малопродуктивной и даже бесполезной.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

- Начинать занятия немедленно, как только сел за стол. Следует начинать с уверенностью, что вскоре придет сосредоточенное состояние, но если внимание наступает не сразу или нарушается на время, нужно выяснить и устранить причины этого.
- Нужно научиться не прерывать внимания, пока читаемое не получит логического завершения, пока не будет пройден какой либо этап. Нередко внимание отвлекается посторонними мыслями, которые во время занятий следует решительно отгонять. Перерыв в занятиях следует приурочить к концу изучения параграфа, раздела или главы книги, та как в этом случае не будет потери времени при возобновлении работы. Умение сосредоточиться, углубиться в работу приобретается в результате практики, создающей определенные навыки.
- Повысить производительность умственного труда может порядок на рабочем месте и обстановка, благоприятствующая работе.

Большая часть самостоятельной работы студента состоит в **изучении литературы**. Одна из задач студента – научиться самостоятельно работать с книгой, а это требует определенных затрат энергии и времени. Поэтому надо научиться делать эту работу рационально, то есть необходимо учиться читать.

Как работать с учебной и научной книгой

Методы эффективной работы с книгой в целях развития интеллекта можно условно разделить на две группы:

1. Правильная организация процесса чтения
2. Повышение скорости чтения и восприятия.

В комплексе оба метода могут в 2-3 раза сократить время прочтения различных материалов.

При чтении текста мозг формирует «свою трактовку содержания» прочитанного. Происходит перекодирование сообщения на языке собственных мыслей читателя. Мозг выделяет «ядерное», сущностное значение из текста. Эффективность такой перекодировки зависит от осмысления и внимательности чтения.

Как показали эксперименты, знание и умелое применение некоторых упражнений дают возможность извлекать «ядерное» значение в тексте быстро и надежно. Эти упражнения основаны на использовании дифференциального алгоритма чтения. Центральное место в этом алгоритме занимает «блок доминанта». Это слово в переводе с латинского языка означает «господствующий, основной, главный». Что же такое доминанта применительно к тексту?

Доминанта – главная смысловая часть текста. Она выражается своими словами, на языке собственных мыслей, является результатом переработки текста, его осмысления в соответствии с индивидуальными особенностями читателя, выявления основного замысла автора.

Дифференциальный алгоритм чтения в соответствии с блоками позволяет реализовать логико-семантический анализ текста: вначале выделить ключевые слова, затем построить смысловые ряды и, наконец, выделив цепь знаний, сформулировать доминанту. Именно так и только так (по О.А. Андрееву) можно увидеть главное, действительно, проникнуть в суть вещей, явлений, излагаемых автором.

Возможны три основных способа чтения.

- Первый способ – артикуляция или проговаривание вслух (или почти вслух) того, что читаешь. Скорость такого чтения невелика.
- Второй способ – чтение про себя, при котором речевой процесс проявлен в форме внутренней речи, то есть без открытой артикуляции. Текст, при этом

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

усваивается более эффективно. Способ в принципе допускает быстрое чтение.

- Третий, наиболее совершенный способ чтения – тоже молча, но в условиях максимального сжатия внутренней речи, при котором она проявляется в виде коротких залпов ключевых слов и смысловых рядов, адекватно отражающих смысл текста.

Итак, артикуляция замедляет процесс чтения и от нее необходимо избавиться. Однако не приведет ли сокращение артикуляции при повышении скорости чтения к снижению качества восприятия и осмысления полученной информации?

Как показали исследования психологов, иногда при чтении слова могут быть заменены наглядными представлениями, пространственными схемами. Целые группы слов – одним словом.

Быстро читающие люди обладают способностью, не проговаривания читаемый текст, сразу улавливать и фиксировать замысел автора, а затем усваивать его на уровне внутренней речи. В этом случае, несмотря на высокую скорость чтения, происходит глубокое понимание и усвоение прочитанного, так как основная идея понятна с самого начала. Задачу научиться такому чтению можно решать в два этапа. Первый предполагает сокращение артикуляции, если она ярко выражена, второй – овладение приемами чтения, при которых текст воспринимается крупными информационными блоками.

Как известно, людей по способу восприятия и переработки информации делят на три типа: зрительный, слуховой и кинестетический. Люди зрительного типа при чтении используют код наглядных образов, тогда как люди слухового типа применяют менее производительный код речедвижений. Наблюдения за людьми, читающими быстро, показывают, что они, как правило, относятся к зрительному типу. Вот пример, как описывает О.Бальзак процесс быстрого чтения: «Впитывание мысли в процессе чтения достигло у него способности феноменальной. Взгляд его охватывал семь – восемь строчек сразу, и разум постигал смысл со скоростью, соответствующей скорости глаз. Часто одно-единственное слово позволяло ему усвоить смысл целой фразы».

Направленным обучением можно практически любого здорового человека научить в процессе чтения использовать код наглядных зрительных образов при соответствующем сокращении артикуляции.

С опорой на работу Л.Г. Одинцова «Как научиться хорошо учиться» (М., 1996) приводим следующие рекомендации по работе с книгой.

- В тексте всегда есть элементы, нахождение и использование которых позволяет извлечь требуемую информацию наиболее быстро. Например, при чтении учебника в первую очередь отыскивается наиболее важная информация данной главы, параграфа, а она часто следует после слов: в итоге, в результате, выводы и т.д.
- Попробуйте в процессе чтения мысленно заглянуть вперед, представить себе, о чем будет идти речь, к какому выводу придет автор, как далее будет строиться изложение и т.д. например, если описывается одна сторона явления, то, очевидно, далее будет описана и другая и т.д. Это позволяет предварительно подготовиться к последующей информации.
- Хорошим упражнением по развитию навыков «предвидения» является остановка чтения в момент, когда, по вашему мнению, заканчивается какая-то часть текста. Попытайтесь предугадать содержание следующей части.
- До начала чтения текста важно собрать о нем как можно больше информации, чтобы точнее представить, что можно получить из данного текста и как лучше работать с ним. Это помогут сделать название, автор, издательство, аннотация, оглавление, предисловие и заключение. Предварительное ознакомление с книгой перед настоящим чтением позволяет сберечь время и труд.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Как правило, предисловие пишется крупным специалистом в данной области, и поэтому излагаемая проблема показывается как бы целиком, в общем плане, без подробностей. А это позволяет лучше сориентироваться, начинать чтение, зная основную цель автора.

- Перед углубленным чтением любого текста (статьи, книги, конспекта, лекции перед экзаменом) сначала бегло просмотрите его целиком. При этом постарайтесь выявить основные стержневые идеи, наиболее крупные части и логику их изложения. Лишь после такого просмотра переходите к более детальному чтению.
- Перед чтением статьи или параграфа учебника попробуйте проделать следующее: прочитайте внимательно первый абзац, потом бегло просмотрите первые или последние фразы следующих абзацев (в них обычно содержится основная информация), обратите внимание на курсивы, разрядки, подзаголовочный текста и, наконец, внимательно прочтите один-два последних абзаца; постарайтесь выявить основное направление текста и его построение.
- Прочитав в тексте интересную идею, полезно остановить свое внимание на ней, прислушаться к тем мыслям, которые она у вас вызвала, подумать о тех последствиях, которые из нее вытекают, попытаться развивать ее дальше.
- Существенно замедляют чтение регрессии – частые возвратные движения глаз, многократное повторное прочитывание материала. Возвратиться к уже прочитанному, но недостаточно хорошо понятому участку лучше всего, когда прочитан законченный смысловой фрагмент текста и сделана хотя бы попытка его осмысления, а не в процессе чтения предложения.
- Любой текст не однороден по своей информационной насыщенности. В некоторых предложениях, абзацах сконцентрировано очень много информации, например, формулируются основные положения, ведущие идеи и т.д., а другие служат лишь иллюстрацией, фоном. Таким образом, текст имеет «смысловой рельеф». Чем точнее читатель умеет определить степень важности каждого отрезка теста и приспособить к «смысловому барьеру» способ своего чтения (то есть замедлить и углубить в более важных местах и ускорять в менее важных), тем продуктивнее чтение. Постарайтесь гибко варьировать способ работы с текстом в соответствии с его «смысловым барьером».
- Чтобы чтение было эффективным, попробуйте по прочитанному всегда отвечать на 6 вопросов: «*Кто делает? Что делает? Когда? Почему? Где? Как?*»

Освойте технику быстрого чтения по специальной методике, например по книге О.А. Андреева, Л.Н. Хромова. Учитесь быстро читать (М., 1991).

Большое значение при чтении учебной и научной литературы имеет умение запоминать прочитанный материал, а для этого необходимо тренировать память. Существуют приемы, позволяющие тренировать память, которыми необходимо овладеть, что позволит повысить эффективность работы с учебной и научной литературой.

Тренировка памяти. В учебной деятельности важно не только, и не столько быстро читать, но и усваивать материал, сохранять в памяти. Память прекрасно тренируема и управляема. Однако прежде чем ее развивать, подумайте, какая именно память вам нужна: на идеи, на логику изложения материала, на схемы и формулы. Это разные виды памяти и развивать их надо по-разному.

Наблюдая за собой, выясните, как вам легче запомнить информацию – если вы ее видите, слышите или записываете. В дальнейшем постарайтесь так организовать работу, чтобы максимально использовать ведущий тип своей памяти.

Если у вас хорошая **зрительная память**, то хорошо запоминаются рисунки, расположение информации на странице, цвет и т.д. помогите себе, выделяя цветными карандашами отдельные места конспекта, обводя рамками, делая значки, пометки на полях, представляя зрительно отдельные аспекты текста.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

При хорошей **слуховой памяти** лучше запоминается звучащая речь. Используйте эту особенность, выделяя интонацией, тембром голоса отдельные места текста, слушая его в записи на магнитофоне, рассуждая в слух и т.д.

В случае **памяти на движение** помогает повторная сокращенная запись запоминаемого материала, например выводов, основных положений текста, рисование таблиц, графиков, схем, а при выполнении лабораторных работ лучше все потрогать и проделать самому.

Наряду с использованием ведущего типа памяти, специально позаботьтесь и о развитии отстающих, так как при многих видах профессиональной деятельности они также могут потребоваться.

Использование приемов логического, осмысленного запоминания в несколько раз повышает продуктивность деятельности. Например, при запоминании лекции, глав учебников особенно действенным является основные аспекты содержания, но и запомнить логику – целесообразную связь отдельных частей материала.

Постарайтесь с первого курса развивать память на то, что непосредственно касается вашей будущей профессии. Это и основной круг идей данной отрасли знаний, и методы, и наиболее интересные факты, и фамилии ведущих специалистов области и т.д. при этом лучше не ждать, что запомнится само, а специально стараться запомнить нужное.

Вообще установка на запоминание, особенно длительное, положительно сказывается на прочности и точности сохранения материала в памяти. Прикажите себе запомнить надолго, а не так как нерадивый студент, спешно «набивающий» себе голову информацией непосредственно перед экзаменом с единственной целью – удержать выученное на один – два дня.

Любая информация запоминается лучше, если в ней намечены какие-то спорные моменты – ориентиры. И как по камушкам переходят реку, так и по этим ориентирам потом легче воспроизвести содержание. При запоминании текста выделяйте «смысловые опорные пункты», которые легко запоминаются, но с которыми тесно связаны целые фрагменты материала. Это может быть крылатая фраза, яркая цитата, пример, идея и т.д.

Материал запоминается непроизвольно, то есть легко и без затраты специальных усилий, если он является целью какой-либо поисковой деятельности. Например, если вы задались вопросом и нашли ответ на то, что долго искали, или нашли подтверждение гипотезы, которую вы сами выдвинули, то это запоминается само собой. Отсюда вывод – организуйте свою деятельность так, чтобы предмет запоминался, являлся целью этой деятельности. Например, ищите, выделяйте в тексте наиболее важные его положения – и они запомнятся, делите текст на части, анализируйте связи между ними – и запомнится логика текста.

При повторении курса лекций, запоминая материал по отдельным темам или даже вопросам, не забывайте повторить связь между ними. Именно тогда в голове укладывается система знаний, которая гораздо эффективнее, чем разрозненные обрывки.

В процессе развития памяти старайтесь не использовать стихийно сложившиеся мнения, механическое зазубривание, а применяйте научно обоснованные методы сознательной и рациональной организации развития памяти и поиск новых приемов.

Предпосылкой хорошей памяти являются осознание человеком своей деятельности и разграничение информации на ту, которая решающим образом помогает скорейшему достижению своих целей, и на менее существенную информацию. Начинайте любое дело с четкой и ясной формулировки его цели; определите, какая информация может оказать решающее воздействие на ее достижение, и сконцентрируйтесь на ней.

Прочному запоминанию способствует многомодельность восприятия, то есть запоминаемый текст читается, проговаривается и прослушивается. Везде, где это возможно, постарайтесь использовать три приема (слух, зрение и чувства) обработки запоминаемой информации сразу несколько органов чувств.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Не очень осмысленную вами информацию, которую, тем не менее, надо запомнить, можно удержать с помощью ассоциативных приемов мнемотехники, суть которых в том, что новое связывается с известным не прямо, а через цепочку дополнительных промежуточных ассоциации (помните цвета спектра – «Каждый охотник желает знать, где сидит фазан»). Везде, где трудно запомнить прямо, найдите дополнительный связующий мостик. Такими «связующими» мостиками являются буквальные «узелки» на память, завязываемые многими на носовом платке. В течение дня человек неизбежно пользуется носовым платком, а там – узелок, «напоминающий», что нужно не забыть сделать определенное дело.

Память будет работать прекрасно, если наряду с имеющимися приемами вы будете придумывать все новые, адекватные различным видам информации. Если такая работа привычна для вас, то с каждым годом память будет становиться все более мощной и продуктивной.

3. Формы ведения записей

Самостоятельная работа с книгой может быть успешной, если текст не только прочитан, но и законспектирован. Существует несколько **форм записей**, но любая форма записи не даст нужного результата, если не будет пробуждать мысли того, кто ее ведет, если отсутствует активная работа ума и формирование своих выводов из прочитанного.

Выбор формы записи зависит от индивидуальных особенностей человека, его образованности и опыта. При этом не меньшую роль играет назначение записей, то есть то, какие задачи ставит перед собой человек (для самообразования, для выступления на семинаре, для использования в будущем).

Введение записей мобилизует наряду со зрительной памятью, также и моторную память. Кроме того, у человека, систематически ведущего записи изучаемой литературы, создается свой фонд материалов для быстрого повторения и мобилизации накопленных знаний.

Все записи должны быть убористыми и компактными. Интервалы между строками должны быть достаточными, чтобы вписывать дополнения. Рекомендуется вести записи ручкой, а карандашом или ручкой другого цвета пользоваться для отметок и выделений при последующей работе. Полезно также датировать записи.

Записи могут носить различный характер: план, выписки, тезисы, аннотирование, конспектирование, реферирование.

1. План - наиболее краткая формой записи.. Это перечень вопросов, рассматриваемых в книге или статье. План обычно раскрывает структуру произведения, логику автора, способствует лучшей ориентации в содержании.

Так, составленным планом можно воспользоваться, чтобы вспомнить прочитанное или быстро отыскать в книге нужное место. Представление об основных пунктах плана дает оглавление книги, поэтому во многих случаях наименования глав и разделов можно использовать в качестве пунктов. Составление плана приучает логически мыслить, вырабатывать умение сжато и последовательно излагать суть вопроса в письменной и устной форме.

Существует два способа составления плана: работа над ним по ходу чтения и составление плана после ознакомления с произведением. При этом план получается более последовательным и стройным.

Трудность составления плана состоит в том, что надо выяснить для себя, прежде всего, построение изучаемого текста, ход мыслей автора и лишь затем изложить содержание работы кратко и ясно. При всей своей краткости план дает представление о содержании прочитанного. Следует учесть, что форма плана не исключает цитирования отдельных мет и обобщений. Различают простой и развернутый план. В отличие от простого плана развернутый план не только содержит перечисление вопросов, но и раскрывает основные

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

идей произведения, может включать выдержки из него, схемы, таблицы. Планом, особенно развернутым, необходимо пользоваться при написании выступления или статьи.

В целом развернутый план дает гораздо большее представление о произведении, его основных идеях, задачах, которые в нем решаются. Он может включать положения, замечания, собственные мысли студента.

Важно знать, что составление планов помогает вырабатывать способность к отвлеченному, абстрактному мышлению, но наибольшую пользу составление плана даст подготовленным лицам, которые бывают достаточно лишь взглянуть на перечень основных вопросов, чтобы воспроизвести содержание прочитанного.

2. Тезисы – более сложная и совершенная форма записи, чем составление плана.

Это сжатое изложение основных мыслей прочитанного произведения или подготовляемого вступления. Особенностью тезисов является их утвердительный характер.

В них сосредотачивается самое главное, только выводы и обобщения, в них меньше доказательств, иллюстрации и пояснений. Тезисы не должны повторять дословно текст, но в ряде мест могут быть близки к нему, воспроизводя некоторые характерные выражения автора, важные для понимания хода его мыслей. Составление тезисов помогает глубже понять основные идеи произведения, выделить главное в нем; приучают сжато, точно и четко сформулировать свои мысли, повышает культуру речи и письма. При составлении тезисов учитывают следующее. Прежде всего, если произведение небольшое, необходимо внимательно изучать его в целом, если большое – изучать по главам и разделам. Затем, когда будут ясны основные идеи, кратко и последовательно излагать их в виде пунктов.

Различают простые и сложные, развернутые тезисы. Если записывают только утверждение чего – либо, такой тезис называют простым, а сложным тезисом будет выражение главной мысли, содержащее, кроме утверждения, еще и краткое ее доказательство.

Часто тезисы формулируются самим автором как выводы и обобщения в заключении книги или разделах книги. Нередко тезисы выделяются в тексте другим шрифтом.

Рекомендуется делать тезисные записи своими словами, причем можно записывать один абзац за другим, учитывая смысловую связь между ними. Но в большинстве случаев следует составлять сводный тезис, сложный по форме. При этом объединяется несколько утверждений, тесно связанных между собой.

Тезисы по содержанию очень близки к **конспекту**, но конспект носит более описательный характер, и его положения не столь категоричны, как в тезисах. Кроме того, конспект представляет собой более полную форму записи.

Следует отметить, что различие между формами записей условно, но в любой форме запись – важнейшая часть самостоятельной работы с книгой.

3. Выписки. Это записи текста из книги: теоретических положений, статистических данных, имеющих по мнению читателя важное значение.

Достоинство выписок состоит в точности воспроизведения текста книги, удобстве пользования записями при последующей работе, в накоплении обобщений и фактического материала. Выписки полезны для повторения, освежения в памяти прочитанного, для быстрой мобилизации своих знаний, когда необходимо в короткий срок вспомнить материал. Выписки выделяют из текста самое главное и тем самым помогают глубже понять его. Без них трудно обойтись при подготовке доклада, реферата, выступления. Выписки следует рассматривать как составную часть тезисов и конспектов.

Выписывать текст можно и по ходу чтения и после его завершения. В последнем случае надо замечать места, которые потом будут выписаны. Необходимо каждую выписку снабжать ссылкой на источник с указанием соответствующей станицы. Это нужно, чтобы в последствии можно было быстро найти в книге соответствующее место. Целесообразно выписывать из текста только такие места, в которых содержится самое главное, суть вопроса. Выписки должны быть ориентированы на изучение произведения в целом, а не отдельных мест, поскольку положения, вырванные из общего контакта, понимаются не-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

редко совсем не так, как этого хотел автор. Иначе говоря, отдельно взятые, лишенные пояснений выдержки могут быть не поняты или поняты неправильно.

Выписки бывают дословные (цитаты) и «свободные», когда мысли автора излагаются своими словами. Следует учесть, что большие отрывки, которые трудно цитировать, целесообразнее в краткой форме переложить своими словами, но «яркие» и важные места лучше выписывать дословно. Каждую цитату следует заключать в кавычки. Если ее берут из середины предложения, то после вводных кавычек ставят три точки. Ставят их и в конце цитаты, если из предложения опущены последние слова.

Следует знать, что какого-либо единого метода выписок, годного для всех случаев, не существует, поскольку у каждого человека свои особенности мышления и восприятия, свои подход к теме. Все это влияет на содержание и характер выписок.

Выписки рекомендуется хранить в картотеке, конвертах или папках, на которых следует обозначить общую тему.

4. Аннотация – еще одна форма записи, являющаяся кратким обобщением содержания книги. Ею удобно пользоваться, если имеется намерение вернуться к изучаемому произведению. Аннотация может быть необходима и для того, чтобы не забыть о нем.

Для составления аннотации надо сначала полностью прочитать и глубоко продумать произведение. При всей своей краткости аннотация может содержать отдельные фрагменты авторского текста, а не только оценку книги или статьи.

5. Резюме очень близко к аннотации. Это запись, являющаяся краткой оценкой прочитанного материала. Различие между ними состоит в том, что аннотация сжато характеризует произведение в целом, а резюме концентрирует внимание на его выводах, главных итогах.

6. Конспект – наиболее совершенная и наиболее сложная форма записи. Слово «конспект» происходит от латинского «conspectus», что означает «обзор, изложение». В правильно составленном конспекте обычно выделено самое основное в изучаемом тексте, сосредоточено внимание на наиболее существенном, в кратких и четких формулировках обобщены важные теоретические положения.

Конспект представляет собой относительно подробное, последовательное изложение содержания прочитанного. На первых порах целесообразно в записях ближе держаться тексту, прибегая зачастую к прямому цитированию автора. В дальнейшем, по мере выработки навыков конспектирования, записи будут носить более свободный и сжатый характер.

Конспект книги обычно ведется в тетради. В самом начале конспекта указывается фамилия автора, полное название произведения, издательство, год и место издания. При цитировании обязательная ссылка на страницу книги. Если цитата взята из собрания сочинений, то необходимо указать соответствующий том. Следует помнить, что четкая ссылка на источник – непременное правило конспектирования. Если конспектируется статья, то указывается, где и когда она была напечатана.

Конспект подразделяется на части в соответсвтии с заранее продуманным планом. Пункты плана записываются в тексте или на полях конспекта. Писать его рекомендуется четко и разборчиво, так как небрежная запись с течением времени становиться малопонятной для ее автора. Существует правило: конспект, составленный для себя, должен быть по возможности написан так, чтобы его легко прочитал и кто-либо другой.

Формы конспекта могут быть разными и зависят от его целевого назначения (изучение материала в целом или под определенным углом зрения, подготовка к докладу, выступлению на занятии и т.д.), а также от характера произведения (монография, статья, документ и т.п.). Если речь идет просто об изложении содержания работы, текст конспекта может быть сплошным, с выделением особо важных положений подчеркиванием или различными значками.

В случае, когда не ограничиваются переложением содержания, а фиксируют в конспекте и свои собственные суждения по данному вопросу или дополняют конспект соот-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ветствующими материалами их других источников, следует отводить место для такого рода записей. Рекомендуется разделить страницы тетради пополам по вертикали и в левой части вести конспект произведения, а в правой свои дополнительные записи, совмещая их по содержанию.

Конспектирование в большей мере, чем другие виды записей, помогает вырабатывать навыки правильного изложения в письменной форме важные теоретических и практических вопросов, умение четко их формулировать и ясно излагать своими словами.

Таким образом, составление конспекта требует вдумчивой работы, затраты времени и труда. Зато во время конспектирования приобретаются знания, создается фонд записей.

Конспект может быть текстуальным или тематическим. В **текстуальном конспекте** сохраняется логика и структура изучаемого произведения, а запись ведется в соответствии с расположением материала в книге. За основу тематического конспекта берется не план произведения, а содержание какой-либо темы или проблемы.

Текстуальный конспект желательно начинать после того, как вся книга прочитана и продумана, но это, к сожалению, не всегда возможно. В первую очередь необходимо составить план произведения письменно или мысленно, поскольку в соответствии с этим планом строится дальнейшая работа. Конспект включает в себя тезисы, которые составляют его основу. Но, в отличие от тезисов, конспект содержит краткую запись не только выводов, но и доказательств, вплоть до фактического материала. Иначе говоря, конспект – это расширенные тезисы, дополненные рассуждениями и доказательствами, мыслями и соображениями составителя записи.

Как правило, конспект включает в себя и выписки, но в него могут войти отдельные места, цитируемые дословно, а также факты, примеры, цифры, таблицы и схемы, взятые из книги. Следует помнить, что работа над конспектом только тогда будет творческой, когда она не ограничена текстом изучаемого произведения. Нужно дополнять конспект данными из другими источников.

В конспекте необходимо выделять отдельные места текста в зависимости от их значимости. Можно пользоваться различными способами: подчеркиваниями, вопросительными и восклицательными знаками, репликами, краткими оценками, писать на полях своих конспектов слова: «важно», «очень важно», «верно», «характерно».

В конспект могут помещаться диаграммы, схемы, таблицы, которые придадут ему наглядность.

Составлению **тематического конспекта** предшествует тщательное изучение всей литературы, подобранной для раскрытия данной темы. Бывает, что какая-либо тема рассматривается в нескольких главах или в разных местах книги. А в конспекте весь материал, относящийся к теме, будет сосредоточен в одном месте. В плане конспекта рекомендуется делать пометки, к каким источникам (вплоть до страницы) придется обратиться для раскрытия вопросов. Тематический конспект составляется обычно для того, чтобы глубже изучить определенный вопрос, подготовиться к докладу, лекции или выступлению на семинарском занятии. Такой конспект по содержанию приближается к реферату, докладу по избранной теме, особенно если включает и собственный вклад в изучение проблемы.

Следующим методом самостоятельной работы с книгой является **реферирование** на определенную тему. Слово реферат употребляется в двух различных значениях:

1. Краткое изложение содержания книги, научной работы;

2.Доклад за заданную тему на основе критического образа литературных источников.

7. Реферат – это один из самых сложных видов самостоятельной работы с книгой, а для этого следует овладеть более простыми приемами работы – разработкой плана, составлением тезисов и конспектов. Подготовка реферата и выступление с его изложением углубляет знания, расширяет кругозор, приучает логически, творчески мыслить, развивать культуру речи.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

При просмотре литературы намечается ориентировочный план реферата, в который включается обычно 3-4 основных вопроса или раздела. Каждом из разделов формулируются подвопросы, помогающие последовательно раскрыть содержание проблемы.

В процессе изучения материала формулировки подвопросов и разделов обычно уточняются. При реферировании следует делать выписки, записывать мысли, возникающие при чтении; следует также точно записывать и определения тех понятий, которые будут использованы в реферате. Из прочитанной литературы нужно заимствовать не буквальный текст, а важнейшие мысли, идеи, теоретические положения; можно цитировать небольшие отрывки, приводить диаграммы, схемы, чертежи, но главное – высказывать собственные соображения по вопросам реферата. Приведенные выше советы следует рассматривать как примерные, предполагающие и другие подходы, поскольку у каждого человека вырабатываются свои приемы и навыки составления рефератов. Большую помощь в работе над рефератом оказывают предисловия к монографиям и сборникам. В них можно найти сведения о цели издания, а также о существующих пробелах в исследовании.

При разработке плана реферата важно учитывать, чтобы каждый его пункт раскрывал одну из сторон избранной темы, а все пункты в совокупности охватывали тему целиком. Различают несколько композиционных решений реферата: во-первых, хронологическое, когда тема раскрывается в исторической последовательности; во-вторых, описательное, при котором тема расчленяется на составные части, в целом раскрывающие определенное явление; в-третьих, аналитическое, когда тема исследуется в ее причинно-следственных связях и взаимозависимых проблемах. Важно следить за тем, чтобы каждый пункт плана был соотнесен с главной темой и не содержал повторения в других пунктах. Важными разделами реферата является вступление и заключение. Во вступлении надо обосновать актуальность темы, обозначить круг составляющих ее проблем, четко и кратко определить задачу своей работы. В заключении делаются краткие выводы, подводятся итоги. В конце реферата должен быть приложен список литературы.

В отличие от тематического конспекта реферат требует большей творческой активности, самостоятельности в обобщении изученной литературы, умения логически стройно изложить материал, оценить различные точки зрения на исследуемую проблему и высказать о ней собственное мнение. В реферате важно связать теоретические положения с практикой.

Итак, реферат – это самостоятельное произведение автора, которое должно свидетельствовать о знании литературы по данной теме, ее основной проблематике, отра-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

жать точку зрения автора реферата на эту проблематику, его умение осмысливать явления жизни на основе теоретических знаний.

При оценке реферата обычно руководствуются следующими критериями:

1. Удалось ли его автору раскрыть сущность данной проблемы;
2. Сумел ли автор показать связь рассматриваемой проблемы с жизнью;
3. Проявил ли автор самостоятельность и творческий подход в изложении реферата;
4. Можно ли считать реферат логически стройным и т.д.

4. Как слушать и конспектировать лекции

Основы знаний закладываются на лекциях, им принадлежит ведущая роль в учебном процессе. На лекциях дается самое важное, основное в изучаемой дисциплине. Основные задачи, стоящие перед лектором: помочь студентам понять основы и усвоить материал на самой лекции, дать указания на то, что требует наибольшего внимания, учить правильному мышлению и создавать ясное представление о методологии изучаемой науки.

Лекции являются эффективным видом занятий для формирования у студентов способности быстро воспринимать новые факты, идеи, обобщать их, а также самостоятельно мыслить.

Лектор излагает теоретический и практический материал, относящийся к основному курсу. Из большого числа монографий, учебников, сборников лектор выбирает самое главное, помогает усвоить логику рассуждений. Интонацией голоса и манерой изложения лектором подчеркивает наиболее существенное, выделяет главное и второстепенное.

Лектор может приводить наблюдения и факты из своего личного опыта, что придает материалу убедительность, повышает интерес к предмету лекции, способствует его усвоению.

Важно помнить, что лекция – это творческий процесс, в котором участвуют одновременно и лектор, и студенты, поэтому она требует атмосферы сотрудничества и уважительного отношения к труду лектора.

Студенту следует научиться понимать и основную идею лекции, а также, следуя за лектором, участвовать в усвоении новых мыслей. Но для этого надо быть подготовленным к восприятию очередной темы. Время, отведенное на лекцию, можно считать использованным полноценно, если студенты понимают роль лектора, задачи лекции, если работают вместе с лектором, а не бездумно ведут конспект.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Подготовленным можно считать такого студента, который, присутствуя на лекции, усвоил ее содержание, а перед лекцией припомнил материал раздела, излагаемого на ней или просмотрел свой конспект, или учебник.

Перед лекцией необходимо прочитывать конспект предыдущей лекции, а после окончания крупного раздела курса рекомендуется проработать его по конспектам и учебникам.

Для наиболее важных дисциплин, вызывающих наибольшие затруднения, рекомендуется перед каждой лекцией просматривать содержание предстоящей лекции по учебнику с тем, чтобы лучше воспринять материал лекции. В этом случае предмет усваивается настолько, что перед экзаменом остается сделать немногое для закрепления знаний.

Важно помнить, что ни одна дисциплина не может быть изучена в необходимом объеме только по конспектам. Для хорошего усвоения курса нужна систематическая работа с учебной и научной литературой, а конспект может лишь облегчить понимание и усвоение материала.

Основная задача при слушании лекции – учиться мыслить, понимать идеи, излагаемые лектором. Большую помощь при этом может оказать конспект. Передача мыслей лектора своими словами помогает сосредоточить внимание, не дает перейти на механическое конспектирование. Механическая запись лекции приносит мало пользы.

Ведение конспекта создает благоприятные условия для запоминания услышанного, т.к. в этом процессе принимают участие слух, зрение и рука. Конспектирование способствует запоминанию только в том случае, если студент понимает излагаемый материал. При механическом ведении конспекта, когда просто записывается слова лектора, присутствие на лекции превращается в бесполезную трату времени.

Некоторые студенты полагают, что при наличии учебных пособий, учебников нет необходимости вести конспект. Такие студенты нередко совершают ошибку, так как не используют конспект как средство, позволяющее активизировать свою работу на лекции или полнее и глубже усвоить ее содержание.

Определенная часть студентов считает, что конспекты лекции могут заменить учебники, поэтому они стремятся к дословной записи лекции и нередко не задумываются над ее содержанием. В результате при разборе учебного материала по механической записи требуется больше труда и времени, чем при понимании и кратком конспектировании лекции.

Конспект ведется в тетради или на отдельных листах. Записи в тетради легче оформить, их удобно брать с собой на лекцию или практические занятия. Рекомендуется в тетради оставлять поля для дополнительных записей, замечаний и пунктов плана. Но

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

конспектирование в тетради имеет и недостаток: в нем мало места для пополнения новыми материалами, выводами и обобщениями. В этом отношении более удобен конспект на отдельных листах (карточках). Из него нетрудно извлечь отдельную необходимую запись, конспект можно быстро пополнить листами, в которых содержатся новые выводы, обобщения, фактические данные. При подготовке выступлений, докладов легко подобрать листки из различных конспектов и свести их вместе. В результате такой работы конспект может стать тематическим.

Но вести конспект на отдельных листках или карточках более трудоемко, чем в тетради. Карточки легко рассыпать и перепутать, приходиться обзаводиться ящичками для хранения карточек, возникает необходимость на каждом листке писать его порядковый номер.

Но затрата труда и времени окупается преимуществами конспектирования на карточках перед конспектом в тетради.

Рекомендуется делать такие карточки, которые помещаются в обычный почтовый конверт. Карточки удобно тасовать, менять при необходимости их последовательность, раскладывать на столе для обзора. Например, учителю истории карточки служили бы долго. Перед уроком можно взять соответствующий конверт и найти в нем материал по узловым вопросам темы, записанный еще в вузе.

При конспектировании допускается сокращение слов, но необходимо соблюдать меру. Каждый студент обычно вырабатывает свои правила сокращения. Но если они не введены в систему, то лучше их не применять, т.к. случайные сокращения ведут к тому, что спустя некоторое время конспект становится непонятным.

Следует знать, что не существует какого-либо единого, годного для всех метода конспектирования. Каждый ведет записи так, как ему представляется наиболее целесообразным и удобным. Собственный метод складывается по мере накопления опыта, но во всех случаях надо стремится к тому, чтобы конспективные записи были краткими и наилучшим образом содействовали глубокому усвоению изучаемого материала. Известный отечественный педагог В.А. Сухомлинский, рекомендовал учиться думать над конспектом уже на лекции и работать над записями ежедневно хотя бы в течение 2 часов. Он советовал также делить конспект на две графы: в первой кратко записывать изложенные лекции, а во второй – то, над чем надо подумать; сюда же следовало заносить узловые, главные вопросы, над которыми надо подумать постоянно, связывая с этим повседневное чтение. Он подчеркивал, что узловые вопросы предмета будут программой, на основе которой припоминается весь материал.

5. Использование компьютера в процессе самостоятельной работы студентов.

Наиболее комплексный ряд заданий, выполняемых студентом в процессе учебы в вузе, развивающих самостоятельность – это написание реферативных, курсовых и дипломных работ, выполнение которых требует применения всего спектра знаний, умений и

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

навыков, приобретенных студентом в процессе обучения. Алгоритм, методика и формы выполнения этих работ практически одинаковы, они различаются содержанием и глубинной проработанности материала. И реферат, и курсовая, и дипломная работы должны выполняться в соответствии с действующими требованиями ГОСТов.

На современном этапе никто уже не представляет себе самостоятельную работу без использования международной информационной сети – **Интернет**. Необходимость использования Интернета возникает не только при подготовке к практическим и семинарским занятиям, но, в большей степени, при написании различных исследовательских и творческих работ. Многие современные монографии, периодические журналы изданы только в электронном виде и с ними можно познакомиться только в Интернете.

Написание работ творческого и исследовательского характера требует знания и умения применять различные компьютерные технологии. Можно предложить следующий алгоритм работы по написанию исследовательских и творческих работ с использованием компьютера.

- Первый этап заключается в наборе материала на компьютере. Для этого необходимо, чтобы на компьютер были установлен текстовый и графический редакторы для набора текста и выполнения различных рисунков, графиков или схем. Если материал неоднородный, т.е. содержит графики, схемы, чертежи, текст, то для этих целей лучше выбрать интегрированный пакет, который позволяет совмещать различного формата файлы (например Word, PageMaker и др.). Цитаты из книг и журналов можно переснимать на сканере – удобно и быстро. Здесь как раз и понадобиться база данных, которая значительно упростит работу с выбранной литературой.
- Второй этап - корректировка ошибок, недочетов. Практика показывает, что чтение с листа более привычно и корректировать удобнее файлы, имея распечатанный образец перед собой.
- Третий этап - печать начисто. Откорректированный и исправленный текст необходимо не забыть проверить на орфографию (по возможности и стилистику) перед тем как распечатать. Чертежи лучше выводить на бумагу на графопостроителе.
- Четвертый этап - рецензия специалистов, работающих в данной области.
- Пятый этап - защита курсовых или дипломных работ на кафедре или в лаборатории. Желательно использовать презентационные компьютерные программы (например, Power point) при ответе – это увеличит наглядность доклада и использовать презентационные средства типа Proxima – проектор, позволяющий выводить на экран содержимое дисплея. Можно также использовать телевизор вместо монитора при наличии специального блока сопряжения.

Почти на всех этапах студент работает самостоятельно. За время выполнения исследования у него развиваются:

1. Навыки и методы работы с литературой: ее анализ, отбор необходимого материала.
2. Навыки и методы работы с персональным компьютером: профессиональный набор текста, выполнение рисунков и чертежей, схем и др.
3. Исследовательские навыки и др.

Поиск в Интернете.

1. **Поиск информации в Интернете** лучше всего начинать с работы в **Интернет-каталоге**.

Один из наиболее полных и хорошо систематизированных каталогов в русскоязычном секторе Интернета находится на сайте www.aport.ru. Есть много других Интернет-каталогов: www.yandex.ru, www.list.ru, www.rambler.ru (русскоязычные), www.altavista.com (англоязычный) и др. Выбор каталога зависит от вкусов пользователя, степени проработанности его тематической структуры, скорости доступа к ресурсам каталога и т.д.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

2. Чтобы попасть на эту страничку, вам надо вписать URL(адрес) данного сайта в адресную строку вашего Интернет-обозревателя (браузера), которая находится в верхней части окна.

3. Перед вами откроется главная страница поисковой системы, например «Апорт».

4. Находим на этой странице ссылку на подкаталог «Наука и образование» и кликаем на ней мышью. Теперь мы попадаем на следующую страницу каталога, где пользователю предлагается выбрать интересующую его рубрику.

5. Ищем на этой странице ссылку на рубрики. Кликаем на нее. Загружается следующая страница, на которой будут ссылки на подрубрики. Под списком рубрик появятся ссылки на конкретные Интернет-ресурсы. Вы выбираете интересующий вас ресурс (при этом можно пользоваться краткой аннотацией, рейтингом популярности сайта, информацией о времени его последнего обновления) и кликаете на его ссылке. Откроется новое окно браузера, в которое будет загружен выбранный вами сайт.

Помимо **тематического поиска** в любом Интернет-каталоге есть **контекстный поиск**.

Часто бывает так, что всю страницу сохранять необязательно, так как интерес вызывают лишь отдельные ее элементы. Текстовая часть страницы без графики и средств мультимедиа сохраняется как файл языка HTML. Часто имеет смысл сохранять только текст, так как любые графические объекты занимают много места на дискетах и жестких дисках компьютера.

Если вам необходимо сохранить только графические элементы страницы (рисунки, фотографии и т.д.), достаточно кликнуть на интересующей вас картинке правой клавишей мыши. Появится диалоговое окно, в котором следует выбрать пункт «Сохранить рисунок как».

Следует помнить, что вы не сможете редактировать составленные в формате HTML Интернет-страницы. В случае если вам по каким-то причинам нужно **внести** в них **правку**, дополнить своими материалами, включить в готовый текстовый документ, то нужно:

1. Выделить мышью в окне браузера необходимый фрагмент текста (весь текст выделяется после нажатия на команды меню «правка» и «выделить все»);
2. Копировать его (команда «копировать» на вкладке «Правка» / «edit»);
3. Вставить в нужный текстовый файл в программе Word (команда «Вставить» / «Paste»).

Вы также можете **распечатать** нужные вам страницы:

1. Одновременно нажмите клавиши ctrl и P или выберите команду «Печать» / «Print» на вкладке «Файл».

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

2. Если вам не нужно распечатывать всю страницу, то вы можете распечатать ее фрагмент. Для этого вы выделяете интересующий участок страницы мышью, а в диалоговом окне печати, указывая диапазон печати, выберите пункт «Выделенный фрагмент».

3. Если вы хотите вернуться на предыдущую страницу, достаточно кликнуть мышкой кнопку «Назад», которая находится в левом верхнем углу окна вашего браузера. Обратный шаг можно сделать, нажав на стрелку «Вперед».

Для того чтобы в следующий раз точно попасть на нужную вам страницу Интернета, совсем не обязательно переписывать ее адрес, часто громоздкий и сложный. Достаточно всего лишь добавить ссылку на страницу в папке «Избранное» (она расположена вверху экрана, на рабочей панели браузера). Если вы хотите запомнить много страниц и к тому же систематизировать их, то направляйтесь на специальный сайт www.zakladki.ru, где вы сможете сохранить гиперссылку на любую Интернет-страницу. В этом случае вы сможете работать не только со ссылками, подобранными вами, но и другими пользователями (при условии, что доступ к ним не закрыт паролем).

6. Подготовка к экзаменам

Экзаменационная сессия – очень тяжелый период работы для студентов и ответственный труд для преподавателей. Главная задача экзаменов – проверка качества усвоения содержания дисциплины.

На основе такой проверки оценивается учебная работа не только студентов, но и преподавателей: по результатам экзаменов можно судить и о качестве всего учебного процесса. При подготовке к экзамену студенты повторяют материал курсов, которые они слушали и изучали в течение семестра, обобщают полученные знания, выделяют главное в предмете, воспроизводят общую картину для того, чтобы яснее понять связь между отдельными элементами дисциплины.

Экзаменам, как правило, предшествует сдача зачетов. К экзаменам допускаются только те студенты, которые сдали зачеты.

При подготовке к экзаменам основное направление дают программы курса и конспект, которые указывают, что в курсе наиболее важно. Основной материал должен прорабатываться по учебнику, поскольку конспекта недостаточно для изучения дисциплины. Учебник должен быть проработан в течение семестра, а перед экзаменом важно сосредоточить внимание на основных, наиболее сложных разделах. Подготовку по каждому разделу следует заканчивать восстановлением в памяти его краткого содержания в логической последовательности.

До экзамена обычно проводится консультация, но она не может возместить отсутствия систематической работы в течение семестра и помочь за несколько часов ос-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

воить материал, требующийся к экзамену. На консультации студент получает лишь ответы на трудные или оставшиеся неясными вопросы. Польза от консультации будет только в том случае, если студент до нее проработает весь материал. Надо учиться задавать вопросы, вырабатывать привычку пользоваться справочниками, энциклопедиями, а не быть на иждивении у преподавателей, который не всегда может тут же, «с ходу» назвать какой-либо факт, имя, событие.

На экзамене нужно показать не только знание предмета, но и умение логически связно построить устный ответ.

Получив билет, надо вдуматься в поставленные вопросы для того, чтобы правильно понять их. Нередко студент отвечает не на тот вопрос, который поставлен, или в простом вопросе ищет скрытого смысла. Не поняв вопроса и не обдумав план ответа, не следует начинать писать. Конспект своего ответа надо рассматривать как план краткого сообщения на данную тему и составлять ответ нужно кратко. При этом необходимо показать умение выражать мысль четко и доходчиво.

Отвечать нужно спокойно, четко, продуманно, без торопливости, придерживаясь записи своего ответа.

На экзаменах студент показывает не только свои знания, но и учится владеть собой. После ответа на билет могут следовать вопросы, которые имеют целью выяснить понимание других разделов курса, не вошедших в билет. Как правило, на них можно ответить кратко, достаточно показать знание сути вопроса. Часто студенты при ответе на дополнительные вопросы проявляют поспешность: не поняв смысла того, что у них спрашивают, начинают отвечать и нередко говорят не по сути.

Студент должен знать, что на экзамене осуществляется не только контроль и выставляется оценка, но это еще и дополнительная возможность, систематизация знаний. Если говорить о сверхзадаче экзаменатора, то она состоит в уяснении не только и не столько того, что студент выучил, сколько того, чему он научился и что останется у него после экзамена, поскольку этот остаток будет характеризовать образовательный уровень студента.

Следует помнить, что необходимым условием правильного режима работы в период экзаменационной сессии является нормальный сон, поэтому подготовка к экзаменам не должна быть в ущерб сну. Установлено, что сильное эмоциональное напряжение во время экзаменов неблагоприятно отражается на нервной системе и многие студенты из-за волнений не спят ночи перед экзаменами. Обычно в сессию студенту не до болезни, так как весь организм озабочен одним - сдать экзамены. Но это еще не значит, что последствия неправильно организованного труда и чрезмерной занятости не скажутся потом. Поэтому каждый студент помнить о важности рационального распорядка рабочего дня и о своевременности снятия или уменьшения умственного напряжения.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Список литературы

Основная

(1) Канцедал С.А. Алгоритмизация и программирование: учебное пособие. – М.: ИД «Форум»: ИНФРА-М, 2008 – 352 с.

(2) Программирование на языке Паскаль: задачник/ под редакцией Усковой О.Ф. – СПб.: Питер, 2005. – 336 с.

(3) Зиборов В. В. Visual Basic 2010 на примерах. – СПб.: БХВ-Петербург, 2010. – 336 с.

(4) Дукин А., Пожидаев А. Самоучитель Visual Basic 2010. . – СПб.: БХВ-Петербург, 2010. –529 с.

(5) Установочный диск с развернутой системой помощи языка программирования Visual Basic, 2010

(6) Культин, Н.Б. Turbo Pascal в задачах и примерах: учебное пособие. – БХВ., 2007 – 256 с.

(7) Павловская, Т.А. Паскаль. Программирование на языке высокого уровня. - СПб: Питер, 2007 – 393 с.

(8) Антонова Г.М., Байков А.Ю. Современные средства ЭВМ и телекоммуникаций: учеб. пособие: Допущено НМС по информатике, 2011. - 144 с., обл.

Дополнительные источники:

(9) В.В. Фаронов Турбо Паскаль. Начальный курс. Учебное пособие-М.: «Нолидж», 1996. – 613 с.

(10) С.А. Абрамов и др. Задачи по программированию. — М.: Наука, 1988. – 210 с.

Интернет источники

(11) http://pascal.proweb.kz/index.php?page=251

(12) http://pascal.proweb.kz/index.php?page=3

(13) http://mif.vspu.ru/books/pascal/

(14) http://msdn.microsoft.com/ru-ru/library/wak0wfyt.aspx

(15) http://msdn.microsoft.com/ru-ru/library/zfce6bw8.aspx

(16) http://msdn.microsoft.com/ru-ru/library/9chk30w7.aspx

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

РАЗДЕЛ 6 УЧЕБНАЯ ПРАКТИКА УП 01.02 «РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ»

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ПРАКТИКИ УП 01.02

Разработка программного модуля

МДК 01.02 Прикладное программирование

ПМ 01 Разработка программных модулей программного обеспечения для компьютерных систем

основной профессиональной образовательной программы (ОПОП)

по специальности СПО 230115 Программирование в компьютерных системах

базовой подготовки

Организация-разработчик: ГАОУ СПО Стерлитамакский колледж строительства, экономики и права

Разработчик: Аришина Вера Федоровна преподаватель первой квалификационной категории

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1. ПАСПОРТ ПРОГРАММЫ УЧЕБНОЙ ПРАКТИКИ УП 01.02

Разработка программного модуля

1.1. Область применения программы

Рабочая программа учебной практики УП 01.02 является частью основной профессиональной образовательной программы в соответствии с ФГОС по специальности **230115 Программирование в компьютерных системах**, входящей в состав укрупненной группы специальностей СПО **230000 Информатика и вычислительная техника** и может быть использована в дополнительном профессиональном образовании в рамках реализации программ переподготовки кадров в учреждениях СПО.

1.2. Место учебной практики в структуре основной профессиональной образовательной программы

Учебная практика проводится в колледже после освоения Раздела 3 «Разработка прикладного программного обеспечения» профессионального модуля ПМ 01 «Разработка программных модулей программного обеспечения для компьютерных систем» на основании приказа Министерства образования и науки Российской Федерации (Минобрнауки России) от 26 ноября 2009г. №673 «Об утверждении Положения об учебной и производственной практике студентов, осваивающих основные профессиональные образовательные программы среднего профессионального образования», зарегистрированного в Минюсте РФ 15 января 2010г.

Настоящее Положение распространяется на все образовательные учреждения, реализующие основные профессиональные образовательные программы среднего профессионального образования (далее - ОПОП СПО) в соответствии с федеральными государственными образовательными стандартами среднего профессионального образования (далее - ФГОС СПО).

Программа учебной практики студентов являются составной частью ОПОП СПО, обеспечивающей реализацию ФГОС СПО.

1.3. Цели и задачи дисциплины – требования к результатам освоения дисциплины:

Целью учебной практики «Разработка программного модуля» является приобретение первоначального практического опыта по по разработке программных модулей.

В результате прохождения учебной практики обучающийся должен **иметь практический опыт**:

- разработки кода программного продукта на основе готовой спецификации на уровне модуля;
- использования инструментальных средств на этапе отладки программного продукта;
- проведения тестирования программно модуля по определенному сценарию.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Освоить ОК:

ОК 1.	Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес.
ОК 2.	Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.
ОК 3.	Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.
ОК 4.	Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.
ОК 5.	Использовать информационно-коммуникационные технологии в профессиональной деятельности.
ОК 6.	Работать в коллективе и в команде, эффективно общаться с коллегами, руководством, потребителями.
ОК 7.	Брать на себя ответственность за работу членов команды (подчиненных), за результат выполнения заданий.
ОК 8.	Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.
ОК 9.	Ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1.4. Рекомендуемое количество часов на освоение программы учебной практики:

максимальной учебной нагрузки обучающегося 54 часов, в том числе:

обязательной аудиторной учебной нагрузки обучающегося 36 часов;

самостоятельной работы обучающегося 18 часов.

2. СТРУКТУРА И ПРИМЕРНОЕ СОДЕРЖАНИЕ УЧЕБНОЙ ПРАКТИКИ

2.1. Объем учебной практики и виды выполняемой работы

Вид выполняемой работы	***Объем часов***
Максимальная учебная нагрузка (всего)	54
Обязательная аудиторная учебная нагрузка (всего)	36
в том числе:	
лабораторные занятия	28
практические занятия	8
Самостоятельная работа обучающегося (всего)	18
в том числе:	
- Поиск дополнительной информации и составление опорного конспекта по работе изучаемой среды программирования	2
- Разработка структуры программы по заданию	4
- Программирование кода в среде программирования.	4
- Подготовка к устным ответам по контрольным вопросам и анализу решенной задачи.	2
- Подбор материала и написание текстовой части отчета по практике	2

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

- Ознакомление с нормативными документами по написанию текста отчета. - Разработка тестового задания к программе	2 2
Итоговая аттестация в форме дифференцированного зачета	

2.2. Примерный тематический план и содержание учебной практики «Разработка программного модуля»

Наименование разделов и тем	Содержание учебного материала, лабораторные и практические работы, самостоятельная работа обучающихся	Объем часов
Тема 1. Составление программного кода модуля		**26**
Практические занятия		4
1.	Постановка цели и инструктаж по ТБ. (ОК 1)	
2.	Постановка групповых задач. (ОК 2)	
Лабораторные занятия		22
1.	Составление структуры кода модуля по заданию.	
2.	Реализация элементов составленной структуры в среде программирования.	
3.	Написание программы по выданной задаче с четким выделением структурных элементов	
4.	Написание программного кода модуля.	
5.	Редактирование и совершенствование элементов кода модуля.	
6.	Объединения процедур и функций пользователя по заданию в единый код.	4
7.	Корректировка данных в объединенных элементах.	
8.	Редактирование и совершенствование кода модуля.	
9.	Работа с полученным кодом.	4
Самостоятельная работа обучающихся		**13**
Поиск дополнительной информации и составление опорного конспекта по работе изучаемой среды программирования (ОК 9) Разработка структуры кода по заданию Программирование кода в среде программирования. Подготовка к устным ответам по контрольным вопросам и анализу решенной задачи.		
Тема 2. Отладка и тестирование программного кода модуля		**10**
Лабораторные занятия		
1.	Выполнение отладки готового программного модуля.	
2.	Тестирование кода по составленному тестовому заданию.	
Практические занятия		6
1.	Составление введение и заключения текста отчета по практике. (ОК 5)	
2.	Дифференцированный зачет (ОК 8.1.2)	2
Самостоятельная работа обучающегося		**5**
Подбор материала и написание текстовой части отчета по практике (ОК 4) Ознакомление с нормативными документами по написанию текста отчета. Разработка тестового задания к программе.		
Всего:		**54**

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

3. УСЛОВИЯ РЕАЛИЗАЦИИ ПРОГРАММЫ ПРАКТИКИ

3.1. Требования к минимальному материально-техническому обеспечению

Реализация программы практики предполагает наличие лаборатории системного и прикладного программирования и полигона вычислительной техники.

Оборудование лаборатории системного и прикладного программирования

- рабочие места по количеству обучающихся с установленным лицензионным программным обеспечением и выходом в глобальную сеть Internet;
- рабочее место преподавателя;
- комплект учебно-методической документации;
- маркерная доска (доска для записей).
- точки электропитания;
- сетевое оборудование, обеспечивающее работу локальной сети;
- мультимедийное оборудование;
- источники бесперебойного питания;
- интерактивная доска.

Комплекс цифровых обязательных ресурсов

- электронные пособия и учебники;
- электронные видеоматериалы.

3.2 Информационное обеспечение обучения

Перечень рекомендуемых учебных изданий, Интернет-ресурсов, дополнительной литературы

Основные источники:

1. ГОСТ 19.701-90 (ИСО 5807-85) схемы алгоритмов, программ, данных и систем.
2. Канцедал С.А. Алгоритмизация и программирование: учебное пособие. – М.: ИД «Форум»: ИНФРА-М, 2008 – 352 с.
3. Программирование на языке Паскаль: задачник/ под редакцией Усковой О.Ф. – СПб.: Питер, 2005. – 336 с.
4. Установочный диск с развернутой системой помощи языка программирования Visual Basic, 2010

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

5. Культин, Н.Б. Turbo Pascal в задачах и примерах: учебное пособие. – БХВ., 2007 – 256 с.

6. Павловская, Т.А. Паскаль. Программирование на языке высокого уровня. - СПб: Питер, 2007 – 393 с.

Дополнительные источники:

1. В.В. Фаронов Турбо Паскаль. Начальный курс. Учебное пособие-М.: «Нолидж», 1996.-613 с.

2. С.А. Абрамов и др. Задачи по программированию. — М.: Наука, 1988. – 210 с.

Интернет-ресурсы

1. Методы программирования
   http://www.tstu.ru/education/elib/pdf/2006/kulakov.pdf
2. Структурное программирование
   http://digital.sibsutis.ru/Progr/StrProgr.htm
3. Модульное программирование
   http://digital.sibsutis.ru/Progr/ManyMod.htm

4. КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ УЧЕБНОЙ ПРАКТИКИ

Контроль и оценка **результатов освоения учебной практики осуществляется преподавателем в процессе проведения практических занятий и лабораторных работ, а также выполнения обучающимися индивидуальных заданий, проектов, исследований.**

Результаты обучения (иметь практический опыт)	**Формы и методы контроля и оценки результатов обучения**
В результате освоения практики обучающийся **должен иметь практический опыт**:	
разработки кода программного продукта на основе готовой спецификации на уровне модуля	Оценка кода программного продукта, написанного в среде программирования, выполненных операций по сохранению и отладке программы по критериям (использование соответствующих правил работы в указанной среде программирования, соответствие спецификации) на дифференцированном зачете
использования инструментальных средств на этапе отладки программного продукта	Оценка продукта учебной деятельности (способы отладки программного кода модуля) по критериям (использование и инструментальных средств для отладки кода) на дифференцированном зачете
проведения тестирования программно модуля по определенному сценарию	Оценка продукта учебной деятельности (результаты тестирования программного кода модуля, составление тестового задания для написанного программного кода) по критериям (соответствие предложенным результатам, соответствие заданию) на дифференцированном зачете

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Формы и методы контроля и оценки результатов обучения должны позволять проверять у обучающихся не только полученный практический опыт, но и развитие общих компетенций.

Результаты обучения (освоенные общие компетенции)	**Формы и методы контроля и оценки результатов обучения**
ОК 1 Понимать сущность и социальную значимость своей профессии, проявлять к ней устойчивый интерес	Оценка результатов формализованного наблюдения за деятельностью обучающихся на учебной практике
ОК 2 Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.	Оценка продукта деятельности обучающегося (решение практико-ориентированного задания) по критериям на дифференцированном зачете
ОК 3 Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.	Оценка результатов формализованного наблюдения за деятельностью обучающегося по критериям на учебной практике
ОК 4 Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.	Оценка результатов формализованного наблюдения за учебной / профессиональной деятельностью обучающегося по критериям на учебной практике
ОК 5 Использовать информационно - коммуникационные технологии в профессиональной деятельности.	Оценка продукта учебной деятельности обучающегося по критериям на учебной практике
ОК 8 Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.	Оценка результатов выполненного задания на дифференцированном зачете
ОК 9 Ориентироваться в условиях частой смены технологий в профессиональной деятельности.	Оценка результатов выполненного задания на дифференцированном зачете

Разработчик:
ГАОУ СПО СКСЭиП преподаватель В.Ф.Аришина

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

№	Наименование разделов, тем	Кол час.	Дата проведен	№ заня-тия	Вид за-нятия	Оборудование заня-тия	*Самостоятельная работа студентов*	Дом. задание	Прим
	Тема 1. Составление программного кода модуля								
1.	Постановка цели и инструктаж по ТБ.	2		1.	Пр. 1	Доска, локальная сеть	Составление конспекта	Конспект	
2.	Постановка групповых задач.	2		2.	Пр. 2	Дидактический матер, локальная сеть	Составление конспекта	Конспект	
3.	Составление структуры кода модуля по заданию.	2		3.	Лаб. 1	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
4.	Изучение и применение дополнительных возможностей среды программирования.	2		4.	Лаб. 2	Дидактические материалы, компьютер, локальная сеть	Составление конспекта Выполнение задания	Конспект	
5.	Реализация элементов составленной структуры в среде программирования.	2		5.	Лаб. 3	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
6.	Написание программы по выданной задаче с четким выделением структурных элементов	2		6.	Лаб. 4	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
7.	Написание программного кода модуля.	2		7.	Лаб. 5	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
8.	Редактирование и совершенствование элементов кода модуля.			8.	Лаб. 6	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
9.	Объединение процедур и функций пользователя по заданию в единый код.	2		9.	Лаб. 7	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
10.	Редактирование процедур и функций			10.	Лаб. 8	Дидактические	Выполнение зада-	Конспект	

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

№	Наименование разделов, тем	Кол час.	Дата проведен	№ занятия	Вид занятия	Оборудование занятия	*Самостоятельная работа студентов*	Дом. задание	Прим
	пользователя по заданию.					материалы, компьютер, локальная сеть	ния		
11.	Корректировка данных в объединенных элементах.	2		11.	Лаб. 9	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
12.	Редактирование и совершенствование кода модуля.	2		12.	Лаб. 10	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
13.	Работа с полученным кодом.	2		13.	Лаб. 11	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
14.	Работа с полученным кодом.			14.	Лаб. 12	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Конспект	
	Тема 2. Отладка и тестирование программного кода модуля								
15.	Выполнение отладки готового программного модуля.			15.	Лаб. 13	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Составление тестового задания	
16.	Тестирование кода по составленному тестовому заданию.			16.	Лаб. 14	Дидактические материалы, компьютер, локальная сеть	Выполнение задания	Подготовка отчета по практике	
17.	Составление введение и заключения текста отчета по практике.	2		17.	Пр. 3	Дидактические материалы, компьютер, локальная сеть	Оформление отчета	Подготовка к диф. зач	
18.	Дифференцированный зачет	2		18.	Пр. 4	Экран, проектор компьютер, локальная сеть			
	Итого:	**36ч.**							

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Задание на учебную практику

«Разработка программного модуля в среде программирования»

ЗАДАНИЕ: Разработать программный код в среде программирования по заданию. Составить для него тестовое задание с последующим тестированием программы. Выполнить отладку программы по выявленным ошибкам. По окончании программирования необходимо написать отчет по проделанной работе.

Примерные темы:

1. Разработать программу вычисления определенного интеграла двумя методами приближенного вычисления.
2. Разработать программу выполнения операций умножения и возведения в степень комплексных чисел, предварительно представив их в тригонометрической форме.
3. Разработать программу выполнения сложения и извлечения корня 3 степени из комплексных чисел, предварительно представив их в показательной форме.
4. Разработать программу вычисления определителя 3 порядка.
5. Разработать программу преобразования натуральных чисел, записанных в римской нумерации, в десятичную систему счисления.
6. Разработать программу, осуществляющую решение дифференциальных уравнений методом Ньютона.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ГАОУ СПО Стерлитамакский колледж строительства, экономики и права

УТВЕРЖДАЮ
Заместитель директора
по учебной работе
_____ В.А. Роганова
подпись

«___»_____20__ г.

Комплект
контрольно-измерительных материалов
по учебной практике УП 01.02

Разработка программного модуля в среде программирования

МДК 01.02 Прикладное программирование

ПМ 01 Разработка программных модулей программного обеспечения для компьютерных систем

основной профессиональной образовательной программы (ОПОП)

по специальности СПО 230115 Программирование в компьютерных системах

базовый уровень подготовки

ГАОУ СПО СКСЭиП преподаватель В.Ф. Аришина

Стерлитамак, 2012

Разработчик:

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

1. Паспорт комплекта контрольно-измерительных материалов

1.1. Область применения

Комплект контрольно-измерительных материалов предназначен для проверки результатов освоения учебной практики (далее УП), относящейся к Разделу 3 «**Разработка прикладного программного обеспечения**» ПМ 01 «**Разработка программных модулей программного обеспечения для компьютерных систем**» основной профессиональной образовательной программы (далее ОПОП) по специальности СПО **230115 Программирование в компьютерных системах базовой подготовки**

Комплект контрольно-измерительных материалов позволяет оценивать освоение практического опыта, общих компетенций:

Освоенный практический опыт и ПК	**Показатели оценки результата**	**Вид задания для проверки**
В результате изучения МДК обучающийся должен **иметь практический опыт**:		
Разработки кода программного продукта на основе готовой спецификации на уровне модуля	Код программного продукта, построен на основе готовой спецификации, в соответствии с правилами программирования на уровне модуля	Составить программный код, на основе готовой спецификации
Использования инструментальных средств на этапе отладки программного продукта	Отладка программного кода выполнена с использованием инструментальных средств	Выполнить отладку полученного кода с использования инструментальных средств
Проведения тестирования программно модуля по определенному сценарию	Тестовое задание составлено в соответствии с заданием, результаты тестирования соответствуют выданному тестовому заданию	Составить тестовое задание и выполнить тестирование программного кода
Освоенные ОК	**Показатели оценки результата**	**Вид задания для проверки**
1	**2**	**3**
ОК 1 Понимать сущность и социальную значимость своей профессии, проявлять к ней устойчивый интерес	Приведены произвольные примеры социальной значимости своей профессии Дано объяснение сущности профессии	Написать введение и выводы в отчете по практике
ОК 2 Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.	Поставленная цель разбита на задачи. Набор ресурсов определен в соответствии с условиями предложенного задания Выбранный метод и способ решения профессиональной задачи соответствует типовому (известному) алгоритму решения Оценка эффективности и качества метода и способа решения задачи соответствует заданной методике оценивания	Решить поставленную в группе часть задачи
ОК 3 Принимать решения в стандартных и нестан-	Анализ проблемы проведен в соответствии с условиями заданной стандартной / нестандартной ситуации	Провести грамотное обсуждение и разделение единой

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

дартных ситуациях и нести за них ответственность.	Принятое решение соответствует результатам проведенного анализа проблемы и позволяет ее решить	задачи на составляющие
ОК 4 Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.	Обращается к информационным системам, базам и банкам компьютерных данных по интересующему вопросу в ходе решения поставленной профессиональной задачи Общается со специалистами по интересующему вопросу	Осуществлять поиск дополнительной информации по программированию поставленной задачи.
ОК 5 Использовать информационно - коммуникационные технологии в профессиональной деятельности.	Использование информационно - коммуникационных технологий в профессиональной деятельности соответствует условиям поставленной профессиональной задачи и характеристикам имеющегося аппаратного обеспечения.	Подготовить теоретическую часть отчета.
ОК 8 Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.	Задачи профессионального развития определены в соответствии с условиями практико-ориентированного задания Запланированное повышение квалификации соответствует запросам потенциальных работодателей, предъявляемым к специалисту «техник-программист»	Самостоятельное выполнение задачи и составление отчета
ОК 9 Ориентироваться в условиях частой смены технологий в профессиональной деятельности.	Перечисляет современные программные и аппаратные ресурсы, соответствующие условиям поставленной профессиональной задачи Формулирует основные характеристики имеющихся программных и аппаратных ресурсов Осваивает предложенную технологию по имеющейся сопроводительной документации	Подготовить теоретическую часть отчета по практике.

1.2. Система контроля и оценки освоения программы УП

Система контроля и оценки освоения программы УП соответствует положению об итоговой и промежуточной аттестации в ГАОУ СПО СКСЭиП.

1.2.1. Организация контроля и оценки освоения программы УП

Контроль и оценка освоения программы УП осуществляется в форме:

- текущего контроля:

при выполнении лабораторных и практических работ;

- промежуточного контроля:

ежемесячная аттестация по текущим оценкам,

- итогового контроля:

дифференцированный зачет в форме отчета по практике.

Оценка освоенного практического опыта, ОК осуществляется с помощью отчета по проделанной работе.

Условием положительной аттестации по УП 01.02 является положительная оценка за отчет по практике.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

2. Комплект материалов для оценки освоения практического опыта и ОК

2.1. Инструкция для обучающегося

Задание состоит в написании программного кода в среде программирования для решения поставленной задачи, составления для него тестового задания с последующим тестированием программы, выполнение отладки программы по выявленным ошибкам. Программный код должен быть разработан на уровне модуля для реализации функциональности программы. По окончании программирования необходимо написать отчет по проделанной работе. На итоговом лабораторном занятии – дифференцированном зачете обучающийся защищает результаты работы по составленному отчету.

Проверяемые результаты обучения:

- опыт разработки кода программного продукта на основе готовой спецификации на уровне модуля,
- опыт использования инструментальных средств на этапе отладки программного продукта,
- опыт проведения тестирования программно модуля по определенному сценарию,
- понимание сущности и социальной значимости своей профессии, проявление к ней устойчивого интереса,
- организация собственной деятельности, выбор методов и способов выполнения профессиональных задач, оценивание их эффективности и качества,
- умение принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность,
- осуществление поиска и использования информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития,
- использование информационно - коммуникационные технологии в профессиональной деятельности,
- самостоятельность определения задач профессионального и личностного развития, занимания самообразованием, осознанное планирование повышения квалификации,
- умение ориентироваться в условиях частой смены технологий в профессиональной деятельности.

Максимальное время для защиты отчета 5-6 мин.
Защита выполненной работы выполняется на двух последних лабораторных занятиях – дифференцированном зачете, **180 мин**.

Основные требования к работе
Требования к структуре и оформлению отчета по практике.

1. Титульный лист. (Приложение А)
2. Текст задания.
3. Введение - постановка целей и задач практики (в соответствии с заданием практики).
4. Теоретические основы решения задачи (включает структуру программы).
5. Описание выполненной обучающимся практической части задания (включая шаги по оптимизации, тестовое задание).
6. Выводы пор практике.
7. Приложения (код программы, результаты тестирования).

Требования к защите работы.

1. Выполняя защиту, обучающийся должен четко знать цель и задачи выполненной работы.
2. Четко пояснять методы и средства, использованные для программирования кода.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

3. Обучающийся должен быть готов к дополнительным вопросам по проделанной работе, направленным на понимание проекта.

Показатели оценки работы

Наименования проверяемых результатов обучения	**Показатели оценки результата**
1. Программный код	Программный код с заданной функциональностью
2. Тестовые задания	Самостоятельно составлено тестовое задание для выявления ошибок
3. Элементы самостоятельного изучения	Наличие в работе элементов, изученных самостоятельно.
4. Отчет по практике	Соответствие отчета формальным и содержательным требованиям.
5. Владение приемами отладки и стандартами кодирования	Формулирует приемы отладки и стандарты кодирования..

Условия выполнения заданий

Выполнение задания проходит в индивидуальной форме. При выполнении задания необходимо воспользоваться компьютером.

Место проведения: лаборатория системного и прикладного программироания.

Используемое оборудование компьютер**.**

Информационно-справочные материалы не требуются.

3. Пакет для эксперта

3.1 Инструкция для эксперта

Количество вариантов заданий для обучающихся, сдающих дифференцированный зачет: 6.

Задание состоит в написании программного кода в среде программирования для решения поставленной задачи, составления для него тестового задания с последующим тестированием программы, выполнение отладки программы по выявленным ошибкам. Программный код должен быть разработан на уровне модуля для реализации функциональности программы. По окончании программирования необходимо написать отчет по проделанной работе. На итоговом лабораторном занятии – дифференцированном зачете обучающийся защищает результаты работы по составленному отчету.

Проверяемые результаты обучения:

- опыт разработки кода программного продукта на основе готовой спецификации на уровне модуля,
- опыт использования инструментальных средств на этапе отладки программного продукта,
- опыт проведения тестирования программно модуля по определенному сценарию,
- понимание сущности и социальной значимости своей профессии, проявление к ней устойчивого интереса,
- организация собственной деятельности, выбор методов и способов выполнения профессиональных задач, оценивание их эффективности и качества,
- умение принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность,

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

- осуществление поиска и использования информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития,
- использование информационно - коммуникационные технологии в профессиональной деятельности,
- самостоятельность определения задач профессионального и личностного развития, заниматься самообразованием, осознанное планирование повышения квалификации,
- умение ориентироваться в условиях частой смены технологий в профессиональной деятельности.

Максимальное время для защиты отчета 5-6 мин.

Защита выполненной работы выполняется на двух последних лабораторных занятиях – дифференцированном зачете, **180 мин**.

Условия выполнения заданий

Выполнение задания проходит в индивидуальной форме. При выполнении задания необходимо воспользоваться компьютером.

Место проведения: лаборатория системного и прикладного программироания.

Используемое оборудование компьютер**.**

Информационно-справочные материалы не требуются.

Рекомендации по проведению оценки**:**

1. Ознакомьтесь с отчетом обучающийсяа, сдающего дифференцированный зачет, оцениваемыми результатами обучения и показателями оценки

2.Результаты работы на учебной практике предусматривают проверку конечных шагов решения задачи и оцениваются по набранным баллам в соответствии с модельным ответом.

Наименования проверяемых результатов обучения	Показатели оценки результата	Оценка
Программный код	Составлен программный код с заданной функциональностью	
Тестовые задания	Самостоятельно составлено тестовое задание для выявления ошибок	
Элементы самостоятельного изучения	Наличие в работе элементов, изученных самостоятельно.	
Отчет по практике	Соответствие отчета формальным и содержательным требованиям.	
Владение приемами отладки и стандартами кодирования	Верный ответ на поставленный вопрос.	

2 балла – результат присутствует в полном объеме,

1 балл – результат присутствует частично,

0 баллов – результат отсутствует.

4. Суммируйте баллы, полученные обучающимся за верно выполненные задания.

5. Поставьте оценку, руководствуясь следующей шкалой:

Сумма баллов	% выполнения заданий	Оценка
10 баллов	91-100%	Отлично
8-9 баллов	76-90%	Хорошо
6-7 баллов	51-75%	удовлетворительно
менее 5 баллов	50%	неудовлетворительно

6. Перенесите № вариантов, набранные баллы обучающимся и выставленные им оценки в оценочную ведомость.

Показатели оценки результатов освоения программы УП

Вид задания для проверки	Освоенный практический опыт и ПК	Показатели оценки результата
	В результате изучения МДК	

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

	обучающийся должен **иметь практический опыт**:	
Составить программный код, на основе готовой спецификации	Разработки кода программного продукта на основе готовой спецификации на уровне модуля	Код программного продукта, построен на основе готовой спецификации, в соответствии с правилами программирования на уровне модуля
Выполнить отладку полученного кода с использования инструментальных средств	Использования инструментальных средств на этапе отладки программного продукта	Отладка программного кода выполнена с использованием инструментальных средств
Составить тестовое задание и выполнить тестирование программного кода	Проведения тестирования программно модуля по определенному сценарию	Тестовое задание составлено в соответствии с заданием, результаты тестирования соответствуют выданному тестовому заданию
Вид задания для проверки	**Освоенные ОК**	**Показатели оценки результата**
Написать введение и выводы в отчете по практике	ОК 1 Понимать сущность и социальную значимость своей профессии, проявлять к ней устойчивый интерес	Приведены произвольные примеры социальной значимости своей профессии Дано объяснение сущности профессии
Решить поставленную в группе часть задачи	ОК 2 Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.	Поставленная цель разбита на задачи. Набор ресурсов определен в соответствии с условиями предложенного задания Выбранный метод и способ решения профессиональной задачи соответствует типовому (известному) алгоритму решения Оценка эффективности и качества метода и способа решения задачи соответствует заданной методике оценивания
Провести грамотное обсуждение и разделение единой задачи на составляющие	ОК 3 Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.	Анализ проблемы проведен в соответствии с условиями заданной стандартной / нестандартной ситуации Принятое решение соответствует результатам проведенного анализа проблемы и позволяет ее решить
Осуществлять поиск дополнительной информации по программированию поставленной задачи.	ОК 4 Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.	Обращается к информационным системам, базам и банкам компьютерных данных по интересующему вопросу в ходе решения поставленной профессиональной задачи Общается со специалистами по интересующему вопросу
Подготовить теоретическую часть отчета.	ОК 5 Использовать информационно - коммуникационные технологии в профессиональной деятельности.	Использование информационно - коммуникационных технологий в профессиональной деятельности соответствует условиям поставленной профессиональной задачи и характеристикам имеющегося аппаратного обеспечения.
Самостоятельное выполнение задачи и составление отчета	ОК 8 Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.	Задачи профессионального развития определены в соответствии с условиями практико-ориентированного задания Запланированное повышение квалификации соответствует запросам потенциальных работодателей, предъявляемым к специалисту «техник-программист»
Подготовить теоретическую часть отчета по практике.	ОК 9 Ориентироваться в условиях частой смены технологий в профессиональной деятельности.	Перечисляет современные программные и аппаратные ресурсы, соответствующие условиям поставленной профессиональной задачи Формулирует основные характеристики имеющихся программных и аппаратных ресурсов Осваивает предложенную технологию по имеющейся сопроводительной документации

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

3.2. Оценочная ведомость

Министерство образования РБ

ГАОУ СПО Стерлитамакский колледж строительства, экономики и права

Оценочная ведомость

для итогового контроля в форме дифференцированного зачета
по УП 01.02 **«Разработка программного модуля в среде программирования»**
обучающихся в группе _____ курса _____ по специальности 230115 «Программирование в компьютерных сетях» базового уровня подготовки

№ п/п	№ вари-анта	Фамилия, имя, отчество обучающегося	Количество набранных баллов	Оценка
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				
29				
30				

Дата проведения дифференцированного зачета «____»_____20 __года
Время, отведённое на проведение дифференцированного зачета, 180 мин.
Эксперт_____(_____) Преподаватель _____ (_____)
Подпись *ФИО* *Подпись* *ФИО*

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

ПРИЛОЖЕНИЕ А. Образец оформления титульного листа отчета

Министерство образования Республики Башкортостан
ГАОУ СПО Стерлитамакский колледж строительства, экономики и права

Специальность 230115
Программирование в компьютерных системах

Отчёт
по учебной практике

Подготовил отчёт
обучающийся группы _____ _____

Руководитель практики _____

Оценка _____

201_

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)