

Государственное бюджетное профессиональное образовательное учреждение  
«Южно-Уральский государственный колледж»

**РАССМОТРЕНО**

Председатель ПЦК

«Информационных технологий»

\_\_\_\_\_/ Назарова Н.А.

«10» мая 2023 г.

**Комплект контрольно-измерительных материалов**

**по профессиональному модулю**

**ПМ.01. РАЗРАБОТКА МОДУЛЕЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ  
КОМПЬЮТЕРНЫХ СИСТЕМ**

**Образовательной программы по специальности СПО**

**09.02.07 Информационные системы и программирование**  
**Квалификация: программист**

Челябинск, 2023

Разработчики:

ГБПОУ «ЮУГК»

(место работы)

преподаватель

(занимаемая должность)

В.В. Исакова

(инициалы, фамилия)

Эксперты:

ЗАО ЮУИК «Трейд-Альянс»

(место работы)

Руководитель отдела А.Ю. Скворцов

информационных  
технологий

(занимаемая должность)

(инициалы, фамилия)

## СОДЕРЖАНИЕ

1. Общие положения	4
2. Комплект КИМ для текущего контроля	11
3. Комплект КИМ для промежуточной аттестации	93
Приложение 1	..
Приложение 2	..
Приложение 3	..

## 1. Общие положения

**Комплект контрольно-измерительных материалов (КИМ) по ПРОФЕССИОНАЛЬНОМУ МОДУЛЮ ПМ.01. РАЗРАБОТКА МОДУЛЕЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ КОМПЬЮТЕРНЫХ СИСТЕМ**

Образовательной программы по специальности СПО

09.02.07 Информационные системы и программирование

содержит КИМ для текущего контроля и КИМ для промежуточной аттестации, которые позволяют оценивать сформированность общих и профессиональных компетенций в соответствии с установленными показателями (спецификация).

**Спецификация сформированности общих компетенций**, освоение которых подтверждается действиями обучающегося при текущем контроле и на промежуточной аттестации:

Таблица 1

ОК	Дескрипторы (показатели сформированности)	Код	Умения	Код	Знания	Код
ОК.01	1. правильно распознает задачу в профессиональном контексте 2. точно перечисляет методы работы в сфере ИТ	ОД.01-1	распознавать задачу и/или проблему в профессиональном и/или социальном контексте;	ОУ.01-1	методы работы в профессиональной и смежных сферах;	ОЗ.01-1
	1. правильно выполняет этапы по решению задачи 2. точно называет структуру плана для решения задачи	ОД.01-2	анализировать задачу и/или проблему и выделять её составные части; определять этапы решения задачи;	ОУ.01-2	структуру плана для решения задач;	ОЗ.01-2
	1. правильно осуществляет поиск информации 2. точно называет порядок оценки результатов решения задачи	ОД.01-3	выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы;	ОУ.01-3	порядок оценки результатов решения задач профессиональной деятельности	ОЗ.01-3
	1. правильно составляет план действий	ОД.01-4	составить план действия;	ОУ.01-4		
	1. правильно определяет ресурсы для решения задачи	ОД.01-5	определить необходимые ресурсы;	ОУ.01-5		
	1. правильно	ОД.01-6	владеть	ОУ.01-6		

ОК	Дескрипторы (показатели сформирован- ности)	Код	Умения	Код	Знания	Код
	применяет методы работы в сфере ИТ		актуальными методами работы в профессиональной и смежных сферах;			
	1.точно и правильно может реализовать составленный план по решению задачи	ОД.01-7	реализовать составленный план;	ОУ.01-7		
	1.объективно оценивает результат своих действий	ОД.01-8	оценивать результат и последствия своих действий (самостоятельно или с помощью наставника)	ОУ.01-8		
ОК.02	1.правильно определяет задачи и ищет информацию средствами ИТ 2.точно и правильно перечисляет номенклатуру информационны х источников	ОД.02-1	определять задачи для поиска информации; определять необходимые источники информации;	ОУ.02-1	номенклатура информационных источников, применяемых в профессионально й деятельности;	ОЗ.02-1
	1.правильно перечисляет приемы структурирован ия информации 2.точно и правильно планирует процесс поиска информации и ее структурирован ие средствами ИТ	ОД.02-2	планировать процесс поиска; структурировать получаемую информацию;	ОУ.02-2	приемы структурирования информации;	ОЗ.02-2
	1.правильно определяет формат оформления	ОД.02-3	выделять наиболее значимое в перечне информации; оценивать	ОУ.02-3	формат оформления результатов поиска	ОЗ.02-3

ОК	Дескрипторы (показатели сформирован- ности)	Код	Умения	Код	Знания	Код
	поиска результатов		практическую значимость результатов поиска; оформлять результаты поиска		информации	
ОК.03	1.точно и правильно определяет актуальность нормативно- правовой документации средствами ИТ	ОД.03-1	определять актуальность нормативно- правовой документации в профессиональной деятельности	ОУ.03-1	содержание актуальной нормативно- правовой документации	ОЗ.03-1
	1.правильно применяет современную научную и профессиональн ую терминологию	ОД.03-2			современная научная и профессиональная терминология	ОЗ.03-2
	1.правильно называет возможные траектории профессиональн ого развития и самообразовани я в сфере ИТ	ОД.03-3			возможные траектории профессиональног о развития и самообразования	ОЗ.03-3
ОК.04	1.правильно организовывает работу коллектива	ОД.04-1	организовывать работу коллектива и команды;	ОУ.04-1	психологические основы деятельности коллектива, психологические особенности личности; основы проектной деятельности	ОЗ.04-1
	1.правильно взаимодействуе т с коллегами в ходе работы на занятиях	ОД.04-2	взаимодействовать с коллегами, руководством, клиентами в ходе профессиональной деятельности	ОУ.04-2		
ОК.05	1.правильно оформляет документы с использованием ИТ	ОД.05-1	грамотно излагать свои мысли и оформлять документы по профессиональной	ОУ.05-1	особенности социального и культурного контекста;	ОЗ.05-1

ОК	Дескрипторы (показатели сформирован- ности)	Код	Умения	Код	Знания	Код
			тематике на государственном языке, проявлять толерантность в рабочем коллективе			
	1.точно называет правила оформления документов средствами ИТ	ОД.05-2			правила оформления документов и построения устных сообщений	ОЗ.05-2
ОК.09	1.правильно применяет средства информационны х технологий для решения профессиональн ых задач 2.правильно определяет современные средства и устройства информатизаци и	ОД.09-1	применять средства информационных технологий для решения профессиональных задач;	ОУ.09-1	современные средства и устройства информатизации	ОЗ.09-1
	1.правильно и точно использует современное программное обеспечение 2.точно называет порядок применения ПО в сфере ИТ	ОД.09-2	использовать современное программное обеспечение	ОУ.09-2	порядок их применения и программное обеспечение в профессионально й деятельности	ОЗ.09-2
ОК.10	1.правильно понимает тексты на темы, связанные со сферой ИТ	ОД.10-1	высказываний на известные темы (профессиональны е и бытовые), понимать тексты на базовые профессиональные темы;	ОУ.10-1	профессиональн ые темы;	ОЗ.10-1
	1.правильно применяет диалоги на	ОД.10-2	участвовать в диалогах на знакомые общие и	ОУ.10-2	основные общеупотребител ьные глаголы	ОЗ.10-2

ОК	Дескрипторы (показатели сформирован- ности)	Код	Умения	Код	Знания	Код
	темы, связанные со сферой ИТ		профессиональные темы;		(бытовая и профессиональная лексика);	
	1.точно и правильно строит простые высказывания о себе и о сфере ИТ	ОД.10-3	строить простые высказывания о себе и о своей профессиональной деятельности;	ОУ.10-3	лексический минимум, относящийся к описанию предметов, средств и процессов профессиональной деятельности;	ОЗ.10-3
	1.правильно объясняет свои действия	ОД.10-4	кратко обосновывать и объяснить свои действия (текущие и планируемые);	ОУ.10-4	особенности произношения;	ОЗ.10-4
	5.правильно пишет и читает тексты ИТ-направленности	ОД.10-5	писать простые связные сообщения на знакомые или интересующие профессиональные темы	ОУ.10-5	правила чтения текстов профессиональной направленности	ОЗ.10-5

**Спецификация профессиональных компетенций**, освоение которых подтверждается действиями обучающегося при текущем контроле и на промежуточной аттестации:

Таблица 2

Формируемые компетенции	Действия	Код	Умения	Код	Знания	Код
ПК.1.1.	1.правильно анализирует техническое задание 2.разрабатывает алгоритм, соответствующий техническому заданию и оформленный в соответствии со стандартами	ПД1.1-1	Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.	ПУ1.1-1	Основные этапы разработки программного обеспечения. Основные принципы технологии структурного и объектно-ориентированного программирования.	ПЗ1.1-1
	1. поясняет основные структуры	ПД1.1-2	Оформлять документацию на программные	ПУ1.1-2	Актуальная нормативно-правовая база в области до-	ПЗ1.1-2



Формируемые компетенции	Действия	Код	Умения	Код	Знания	Код
	технического задания, 2. указывает использованные стандарты в области документирования;		средства.		документирования алгоритмов.	
	1. выполняет оценку сложности алгоритма	ПД1.1-3	Оценка сложности алгоритма.	ПУ1.1-3	Методы оценки сложности алгоритмов	ПЗ1.1-3
ПК.1.2.	1. Разрабатывает программный модуль, полностью соответствующий техническому заданию, на указанном языке программирования методами объектно-ориентированного/структурного программирования.	ПД1.2-1	Создавать программу по разработанному алгоритму как отдельный модуль.	ПУ1.2-1	Основные принципы технологии структурного и объектно-ориентированного программирования. Знание API современных мобильных операционных систем.	ПЗ1.2-1
	соблюдает и поясняет основные этапы разработки	ПД1.2-2	Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.	ПУ1.2-2	Основные этапы разработки программного обеспечения.	ПЗ1.2-2
	документация на модуль оформлена и соответствует	ПД1.2-3	Оформлять документацию на программные средства.	ПУ1.2-3	Документация на программные средства	ПЗ1.2-3

Формируемые компетенции	Действия	Код	Умения	Код	Знания	Код
	стандартам					
ПК.1.3.	Выполняет отладку модуля с использованием инструментария среды проектирования, с пояснением особенностей отладочных классов	ПД1.3-1	Выполнять отладку и тестирование программы на уровне модуля. Применять инструментальные средства отладки программного обеспечения.	ПУ1.3-1	Основные принципы отладки и тестирования программных продуктов.	ПЗ1.3-1
	сохранены и представлены результаты отладки	ПД1.3-2	Оформлять документацию на программные средства.	ПУ1.3-2	Инструментарий отладки программных продуктов.	ПЗ1.3-2
ПК.1.4.	выполняет тестирование модуля, в том числе с помощью инструментальных средств	ПД1.4-1	Выполнять отладку и тестирование программы на уровне модуля.	ПУ1.4-1	Основные виды и принципы тестирования программных продуктов.	ПЗ1.4-1
	оформляет результаты тестирования в соответствии со стандартами	ПД1.4-2	Оформлять документацию на программные средства.	ПУ1.4-2	Документация на программные средства.	ПЗ1.4-2
ПК.1.5.	определяет качественные характеристики программного кода с помощью инструментальных средств;	ПД1.5-1	Определять необходимость оптимизации и рефакторинга программного кода	ПУ1.5-1	Инструментальные средства анализа алгоритма. Способы оптимизации и приемы рефакторинга.	ПЗ1.5-1
	выявляет фрагменты некачественного кода; выполняет рефакторинг на	ПД1.5-2	Выполнять рефакторинг программного кода.	ПУ1.5-2	Методы организации рефакторинга и оптимизации кода.	ПЗ1.5-2

Формируемые компетенции	Действия	Код	Умения	Код	Знания	Код
	уровнях переменных, функций, классов, алгоритмических структур;					
	проводит оптимизацию и подтверждает повышение качества программного кода	ПД1.5-3	Выполнять оптимизацию программного кода. Работать с системой контроля версий.	ПУ1.5-3	Принципы работы с системой контроля версий.	ПЗ1.5-3
ПК.1.6.	1.правильно называет этапы разработки ПО 2.точно и правильно осуществляет разработку кода программного модуля	ПД1.6-1	Осуществлять разработку кода программного модуля на современных языках программирования.	ПУ1.6-1	Основные этапы разработки программного обеспечения.	ПЗ1.6-1
	1.правильно называет принципы технологии структурного и объектно-ориентированного программирования 2.правильно и точно оформляет документацию на программные средства	ПД1.6-2	Оформлять документацию на программные средства.	ПУ1.6-2	Основные принципы технологии структурного и объектно-ориентированного программирования	ПЗ1.6-2

**Перечень учебных изданий,  
дополнительной литературы, Интернет-ресурсов**

**Основные источники:**

**Печатные издания**

1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н Федорова. – М.: Академия, 2016. – 336 с.
2. Подбельский В. Язык С#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с.

**Электронные издания (электронные ресурсы)**

1. Лаврищева Е. М. Программная инженерия и технологии программирования сложных систем: учебник для вузов / Е. М. Лаврищева. — 2-е изд., испр. и доп. (Бакалавр. Академический курс). Текст: электронный Москва: Издательство Юрайт. <https://biblio-online.ru>. 2019 г., 432 с.
2. Зараменских Е. П. Информационные системы: управление жизненным циклом: учебник и практикум для среднего профессионального образования / Е. П. Зараменских. (Профессиональное образование. Текст: электронный Москва: Издательство Юрайт. <https://biblio-online.ru>. 2019 г., 431 с.
3. Учебники по программированию <http://programm.ws/index.php>

**Дополнительные источники:**

1. Герберт Шилдт. С# 4.0: Полное руководство. – М.: ООО "И.Д. Вильямс". 2011 г., 592 с.
2. Галисеев Г.В. Программирование на языке С#. М.: Издательский дом «Вильямс». 2017 г., 368 с.

## **1. Комплект КИМ для текущего контроля**

Текущий контроль освоения студентами материала дисциплины (или междисциплинарного курса) состоит из следующих видов: *оперативный и рубежный контроль*.

При проведении текущего контроля используются следующие формы:

- 1) *компьютерное тестирование*
- 2) *практическое задание*
- 3) *устный опрос в аудитории*

При проведении текущего контроля при проведении компьютерного тестирования используется оболочка Moodle; при выполнении практического задания выдается методическая разработка для студентов; при проведении письменного опроса выдается задание для студентов в виде списка вопросов.

**КИМ № 1**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ УСТНОГО ОПРОСА В АУДИТОРИИ**

<b>Раздел 1. Разработка программных модулей</b>		<b>МДК. 01.01 Разработка программных модулей</b>
<b>Тема 1.1.1 Жизненный цикл ПО</b>		1. Понятие ЖЦ ПО. Этапы ЖЦ ПО. Стандарты жизненного цикла ПО. Процессы жизненного цикла ПО: процессы соглашения, процессы организационного обеспечения проекта, процессы проекта, технические процессы, процессы реализации программных средств, процессы поддержки программных средств, процессы повторного применения программных средств. Модели жизненного цикла ПО.
<b>Форма контроля</b>		<i>устный опрос в аудитории</i>
<b>Вид контроля</b>		Индивидуальная работа Пользуясь презентациями, ответить на вопросы.
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Устный опрос выполняется в аудитории, время проведения работы 15 минут
<b>Инструкция для студентов</b>		Устно ответить на поставленный вопрос
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО;
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с. Электронные издания (электронные ресурсы)

	<p>1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a></p> <p>Дополнительные источники:</p> <p>1. Подбельский В. Язык C#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342</p>
<b>Вариант</b>	<p>Вопросы с открытым ответом:</p> <ol style="list-style-type: none"> <li>1. Что такое жизненный цикл информационной системы?</li> <li>2. Каковы этапы предпроектной стадии жизненного цикла информационной системы?</li> <li>3. Каковы этапы стадии проектирования в жизненном цикле информационной системы?</li> <li>4. Каковы этапы стадии внедрения в жизненном цикле информационной системы?</li> <li>5. Какие процессы могут возникать на стадии функционирования в жизненном цикле информационной системы?</li> <li>6. Что такое модель жизненного цикла информационной системы?</li> <li>7. Каков алгоритм «модели кодирования и устранения ошибок»?</li> <li>8. Что такое «каскадная стратегия»?</li> <li>9. Каковы достоинства и недостатки каскадной модели?</li> <li>10. Что такое инкрементная стратегия?</li> <li>11. Что такое спиральная стратегия?</li> <li>12. Каковы достоинства спиральной модели?</li> <li>13. Каковы недостатки спиральной модели?</li> <li>14. Провести сравнительный анализ моделей ЖЦ ПО.</li> <li>15. Что такое RAD-разработка?</li> <li>16. Что такое методология XP?</li> <li>17. Каковы особенности V-модели ЖЦ ПО?</li> <li>18. Что такое модель Хаоса?</li> <li>19. Что такое компонентно-ориентированная модель?</li> </ol>
<b>Пакет преподавателя</b>	<p><i>Ответы:</i></p> <p>1. Жизненный цикл информационных систем – это период их создания и использования, охватывающий различные состояния, начиная с момента возникновения необходимости в такой системе и заканчивая моментом ее полного выхода из употребления у пользователей.</p> <p>2. На предпроектной стадии можно выделить следующие этапы:</p> <ol style="list-style-type: none"> <li>1) Сбор материалов для проектирования – предусматривает разработку и выбор варианта концепции системы, выявление всех характеристик объекта и управленческой деятельности, потоков внутренних и внешних информационных связей, состава задач и специалистов, которые будут работать в новых технологических условиях, уровень их подготовки, как будущих пользователей системы.</li> <li>2) Анализ материалов и формирование документации – составление задания на проектирование, утверждение технико-экономического обоснования.</li> </ol> <p>3. Стадия проектирования делится на:</p> <ol style="list-style-type: none"> <li>1) Этап технического проектирования – формируются проектные решения по обеспечивающей и функциональной частям информационной системы, моделированию производственных, хозяйственных, финансовых ситуаций, осуществляется постановка задачи, блок-схемы и их решение.</li> <li>2) Этап рабочего проектирования – осуществляется разработка и доводка системы, корректировка структуры, создание различной документации: на поставку, на установку технических средств, инструкции по эксплуатации, должностные инструкции.</li> </ol> <p>4. Стадия внедрения информационной системы предполагает:</p> <ol style="list-style-type: none"> <li>1) Подготовку к вводу в эксплуатацию – на этом этапе производится установка технических средств, настройка системы, обучение персонала, пробное использование.</li> <li>2) Проведение опытных испытаний всех компонентов системы перед запуском.</li> <li>3) Сдача в промышленную эксплуатацию, которая оформляется актом сдачи-приемки работ.</li> </ol> <p>5. На этапе функционирования информационной системы в рабочем режиме не исключается корректировка функций и управляющих параметров. Также осуществляется оперативное обслуживание и администрирование.</p> <p>6. Модель жизненного цикла информационной системы — это структура, содержащая процессы, действия и задачи, которые осуществляются в ходе разработки, использования и сопровождения программного продукта.</p>

7. Эта модель имеет следующий алгоритм:

- Постановка задачи
- Выполнение
- Проверка результата
- При необходимости переход к первому или второму пункту.

8. Каскадная стратегия (однократный проход, водопадная или классическая модель, автор Уинстон Ройс, 1970г.) – подразумевает линейную последовательность выполнения стадий создания информационной системы. Переход с одной стадии на следующую происходит только после того, как будет полностью завершена работа на текущей.

9. Достоинства каскадной модели:

- на каждом этапе формируется законченный набор проектной документации, отвечающий критериям полноты и согласованности;
- выполняемые в логичной последовательности этапы работ позволяют планировать сроки завершения всех работ и соответствующие затраты (финансовые, материальные и людские).

Недостатки каскадной модели:

- Основным недостатком каскадного подхода является существенное запаздывание с получением результатов. Согласование результатов с пользователями производится только в точках, планируемых после завершения каждого этапа работ, требования к ИС "заморожены" в виде технического задания на все время ее создания.
- Модели (как функциональные, так и информационные) автоматизируемого объекта могут устареть одновременно с их утверждением.

10. Инкрементная стратегия (англ. increment – увеличение, приращение) подразумевает разработку информационной системы с линейной последовательностью стадий, но в несколько инкрементов (версий), т. е. с запланированным улучшением продукта.

11. Спиральная стратегия (эволюционная или итерационная модель, автор Барри Бозм, 1988 г.) – подразумевает разработку в виде последовательности версий, но в начале проекта определены не все требования. Требования уточняются в результате разработки версий.

12. Достоинства спиральной модели:

- позволяет быстрее показать пользователям системы работоспособный продукт, тем самым активизируя процесс уточнения и дополнения требований;
- допускает изменение требований при разработке информационной системы, что характерно для большинства разработок, в том числе и типовых;
- обеспечивает большую гибкость в управлении проектом;
- позволяет получить более надежную и устойчивую систему – по мере развития системы ошибки и слабые места обнаруживаются и исправляются на каждой итерации;
- позволяет совершенствовать процесс разработки – анализ, проводимый в каждой итерации, позволяет проводить оценку того, что должно быть изменено в организации разработки, и улучшить ее на следующей итерации;
- уменьшаются риски заказчика. Заказчик может с минимальными для себя финансовыми потерями завершить развитие неперспективного проекта.

13. Недостатки спиральной модели:

- увеличивается неопределенность у разработчика в перспективах развития проекта. Этот недостаток вытекает из предыдущего достоинства модели;
- затруднены операции временного и ресурсного планирования всего проекта в целом.

Для решения этой проблемы необходимо ввести временные ограничения на каждую из стадий жизненного цикла. Переход осуществляется в соответствии с планом, даже если не вся запланированная работа выполнена. План составляется на основе статистических данных, полученных в предыдущих проектах и личного опыта разработчиков.

14.

Характеристика проекта	Модель (стратегия)		
	Каскадная	Инкрементная	Спиральная
Новизна разработки и обеспеченность ресурсами	Типовой. Хорошо проработаны технология и методы решения задачи		
	Ресурсов заказчика и разработчика хватает для реализации проекта в сжатые сроки	Ресурсов заказчика или разработчика не хватает для реализации проекта в сжатые сроки	Нетиповой (новаторский). Нетрадиционный для разработчика



Характеристика проекта	Модель (стратегия)		
	Каскадная	Инкрементная	Спиральная
Масштаб проекта	Малые и средние проекты	Средние и крупные проекты	Любые проекты
Сроки выполнения проекта	До года	До нескольких лет. Разработка одной версии может занимать срок от нескольких недель до года	
Заключение отдельных договоров на отдельные версии	Заключается один договор. Версия и есть итоговый результат проекта	На отдельную версию или несколько последовательных версий обычно заключается отдельный договор	
Характеристика проекта	Модель (стратегия)		
	Каскадная	Инкрементная	Спиральная
Определение основных требований в начале проекта	Да	Да	Нет
Изменение требований по мере развития проекта	Нет	Незначительное	Да
Разработка итерациями (версиями)	Нет	Да	Да
Распространение промежуточного ПО	Нет	Может быть	Да

15. Под RAD-разработкой обычно понимается процесс разработки, содержащий 3 элемента:

- небольшую команду программистов (до 10 человек);
- короткий, но тщательно проработанный производственный график (от 2 до 6 месяцев);
- повторяющийся цикл, при котором разработчики по мере того, как приложение начинает обретать форму, реализуют в продукте требования, полученные через взаимодействие с заказчиком.

Помимо особенностей, характерных для спиральной модели жизненного цикла, методология RAD подразумевает использование на каждой итерации:

- CASE-средств (Computer Aided Software Engineering (автоматизированная разработка ПО) для формирования и анализа требований, проектирования системы, автоматической генерации кода программ и структуры БД, а также автоматического тестирования программного обеспечения;

инструментальных средств, обеспечивающих визуальную разработку (программирование) системы.

- инструментальных средств, поддерживающих объектно-ориентированный подход. Эти средства позволяют создать описание предметной области в виде совокупности объектов – сущностей реального мира, характеризующихся свойствами (атрибутами) и поведением (методами);

- инструментальных средств, обеспечивающих событийное программирование. Каждый объект, входящий в состав приложения, может генерировать события и реагировать на события, генерируемые другими объектами;

шаблонов и библиотек готовых решений как собственной разработки, так и сторонних производителей.

17. Данный подход ориентирован на разработку информационных систем группами малого и среднего размера в условиях неопределенных или быстро изменяющихся требований.

Отличительными особенностями XP-разработки являются:

- частая смена версий и модификаций (длительность итераций вплоть до часов и минут, чаще – 2 недели);

непрерывная связь с заказчиком – в группе все время находится квалифицированный представитель заказчика.

- простое проектирование – при разработке всегда выбирается наиболее простое решение;

- простой дизайн – система должна быть спроектирована настолько просто, насколько это возможно на каждый момент времени. Чем интерфейс проще, тем быстрее и качественнее идет освоение системы пользователями;

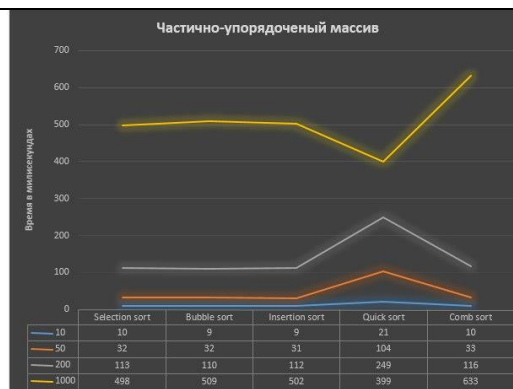
- коллективное владение кодом – любой, кто видит возможность улучшить какую-то часть кода, может сделать это в любой момент времени. Это подразумевает применение одинаковых стандартов и правил оформления кода с исчерпывающими комментариями, а также и ведение общедоступной истории развития системы;

	<ul style="list-style-type: none"> <li>• программирование в парах – на пару программистов приходится один компьютер. Пока один из них непосредственно программирует, другой обдумывает вопросы реализации требований (функций, БД и т.п.);</li> <li>• непрерывное и пересекающееся проектирование, разработка, интеграция и тестирование системы.</li> </ul> <p>17. V-модель – направлена на упрощение понимания сложностей, связанных с разработкой систем. Она используется для определения единой процедуры разработки программных продуктов, аппаратного обеспечения и человеко-машинных интерфейсов.</p> <p>18. Модель хаоса – главное правило этой модели: всегда решать наиболее важную задачу первой.</p> <p>Задача — это незавершенная частная задача программирования.</p> <p>Наиболее важная задача — это комбинация большого размера, срочности и устойчивости.</p> <ul style="list-style-type: none"> <li>• Задачи большого размера ценны для пользователей настолько, насколько они функциональны.</li> <li>• Срочные задачи своевременны настолько, насколько должны быть, иначе задерживается остальная работа.</li> <li>• Устойчивые задачи проверены и испытаны. Разработчики могут благополучно сфокусироваться на другом.</li> </ul> <p>Решить означает привести в состояние стабильности.</p> <p>19. Компонентно-ориентированная модель – является развитием спиральной модели и тоже основывается на эволюционной стратегии конструирования.</p> <p>Программные компоненты, созданные в реализованных программных проектах, хранятся в библиотеке. В новом программном проекте, исходя из требований заказчика, выявляются кандидаты в компоненты. Далее проверяется наличие этих кандидатов в библиотеке. Если они найдены, то компоненты извлекаются из библиотеки и используются повторно. В противном случае создаются новые компоненты, они применяются в проекте и включаются в библиотеку.</p>	
Критерии оценки	Отлично	Ответ полный и правильный, показывающий прочные знания в области профессиональной деятельности. В ответе может быть допущена 1 ошибка
	Хорошо	ответ полный и правильный, показывающий прочные знания в области профессиональной деятельности, при этом допущены две-три ошибки в ответе, исправленные самостоятельно по требованию преподавателя
	Удовлетворительно	ответ полный, но при этом допущены 4-5 ошибок в ответе
	Неудовлетворительно	при ответе обнаружено непонимание обучающимся основного содержания

**КИМ № 2**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Тема 1.1.2</b> <b>Структурное программирование</b>		<p><b>1. Технология структурного программирования.</b> Парадигмы программирования. Сравнительная характеристика структурного программирования. Управляющие структуры: последовательность, ветвление, цикл. Принципы структурного программирования. Подпрограмма.</p> <p><b>2. Инструментальные средства оформления и документирования алгоритмов программ.</b> Единая система программной документации (ЕСПД). Текст программы. Описание программы. Комментарии, отступы. Правила именования. Венгерская нотация.</p> <p><b>3. Оценка сложности алгоритма: классификация, классы алгоритмов, неразрешимые задачи.</b> Оценка порядка. Определение сложности. Сложность рекурсивных алгоритмов: простая рекурсия, многократная рекурсия. Объемная сложность рекурсивных алгоритмов. Общие функции оценки сложности.</p>
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники:

	<p>Печатные издания</p> <p>1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н Федорова. – М.: Академия, 2016. – 336 с.</p> <p>Электронные издания (электронные ресурсы)</p> <p>1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a></p> <p>Дополнительные источники:</p> <p>1. Подбельский В. Язык C#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342</p>																																																																																																																																																										
Вариант	Вариант 1. Выполнить сравнение алгоритмов сортировки. Результат представить в графиках. Сделать вывод.																																																																																																																																																										
Пакет преподавателя	<p>Проверяется правильность выполнения задания, согласно критериям</p> <p>Пример решения:</p> <p>Полностью неотсортированный массив:</p> <table><thead><tr><th rowspan="2">Название сортировки</th><th colspan="2">10 элементов</th><th colspan="2">50 элементов</th><th colspan="2">200 элементов</th><th colspan="2">1000 элементов</th></tr><tr><th>Время</th><th>Память</th><th>Время</th><th>Память</th><th>Время</th><th>Память</th><th>Время</th><th>Память</th></tr></thead><tbody><tr><td>Selection sort</td><td>13 ms</td><td>510 К</td><td>37 ms</td><td>637 К</td><td>118 ms</td><td>854 К</td><td>550 ms</td><td>936 К</td></tr><tr><td>Bubble sort</td><td>11 ms</td><td>524 К</td><td>37 ms</td><td>629 К</td><td>116 ms</td><td>863 К</td><td>564 ms</td><td>932 К</td></tr><tr><td>Insertion sort</td><td>12 ms</td><td>512 К</td><td>38 ms</td><td>641 К</td><td>116 ms</td><td>849 К</td><td>556 ms</td><td>928 К</td></tr><tr><td>Quick sort</td><td>22 ms</td><td>389 К</td><td>102 ms</td><td>454 К</td><td>412 ms</td><td>612 К</td><td>732 ms</td><td>730 К</td></tr><tr><td>Comb sort</td><td>12 ms</td><td>505 К</td><td>37 ms</td><td>632 К</td><td>117 ms</td><td>854 К</td><td>560 ms</td><td>936 К</td></tr></tbody></table> <p>Частично отсортированный массив (половина элементов упорядочена):</p> <table><thead><tr><th rowspan="2">Название Сортировки</th><th colspan="2">10 элементов</th><th colspan="2">50 элементов</th><th colspan="2">200 элементов</th><th colspan="2">1000 элементов</th></tr><tr><th>Время</th><th>Память</th><th>Время</th><th>Память</th><th>Время</th><th>Память</th><th>Время</th><th>Память</th></tr></thead><tbody><tr><td>Selection sort</td><td>10 ms</td><td>501 К</td><td>32 ms</td><td>612 К</td><td>113 ms</td><td>823 К</td><td>498 ms</td><td>942 К</td></tr><tr><td>Bubble sort</td><td>9 ms</td><td>498 К</td><td>32 ms</td><td>601 К</td><td>110 ms</td><td>812 К</td><td>509 ms</td><td>939 К</td></tr><tr><td>Insertion sort</td><td>9 ms</td><td>482 К</td><td>31 ms</td><td>597 К</td><td>112 ms</td><td>802 К</td><td>502 ms</td><td>920 К</td></tr><tr><td>Quick sort</td><td>21 ms</td><td>321 К</td><td>104 ms</td><td>409 К</td><td>249 ms</td><td>610 К</td><td>399 ms</td><td>874 К</td></tr><tr><td>Comb sort</td><td>10 ms</td><td>498 К</td><td>33 ms</td><td>563 К</td><td>116 ms</td><td>601 К</td><td>505 ms</td><td>907 К</td></tr></tbody></table> <p>Графики:</p> <table><thead><tr><th></th><th>Selection sort</th><th>Bubble sort</th><th>Insertion sort</th><th>Quick sort</th><th>Comb sort</th></tr></thead><tbody><tr><td>10 элементов</td><td>13</td><td>11</td><td>12</td><td>22</td><td>12</td></tr><tr><td>50 элементов</td><td>37</td><td>37</td><td>38</td><td>102</td><td>37</td></tr><tr><td>200 элементов</td><td>118</td><td>116</td><td>116</td><td>412</td><td>117</td></tr><tr><td>1000 элементов</td><td>550</td><td>564</td><td>556</td><td>732</td><td>560</td></tr></tbody></table>	Название сортировки	10 элементов		50 элементов		200 элементов		1000 элементов		Время	Память	Время	Память	Время	Память	Время	Память	Selection sort	13 ms	510 К	37 ms	637 К	118 ms	854 К	550 ms	936 К	Bubble sort	11 ms	524 К	37 ms	629 К	116 ms	863 К	564 ms	932 К	Insertion sort	12 ms	512 К	38 ms	641 К	116 ms	849 К	556 ms	928 К	Quick sort	22 ms	389 К	102 ms	454 К	412 ms	612 К	732 ms	730 К	Comb sort	12 ms	505 К	37 ms	632 К	117 ms	854 К	560 ms	936 К	Название Сортировки	10 элементов		50 элементов		200 элементов		1000 элементов		Время	Память	Время	Память	Время	Память	Время	Память	Selection sort	10 ms	501 К	32 ms	612 К	113 ms	823 К	498 ms	942 К	Bubble sort	9 ms	498 К	32 ms	601 К	110 ms	812 К	509 ms	939 К	Insertion sort	9 ms	482 К	31 ms	597 К	112 ms	802 К	502 ms	920 К	Quick sort	21 ms	321 К	104 ms	409 К	249 ms	610 К	399 ms	874 К	Comb sort	10 ms	498 К	33 ms	563 К	116 ms	601 К	505 ms	907 К		Selection sort	Bubble sort	Insertion sort	Quick sort	Comb sort	10 элементов	13	11	12	22	12	50 элементов	37	37	38	102	37	200 элементов	118	116	116	412	117	1000 элементов	550	564	556	732	560
Название сортировки	10 элементов		50 элементов		200 элементов		1000 элементов																																																																																																																																																				
	Время	Память	Время	Память	Время	Память	Время	Память																																																																																																																																																			
Selection sort	13 ms	510 К	37 ms	637 К	118 ms	854 К	550 ms	936 К																																																																																																																																																			
Bubble sort	11 ms	524 К	37 ms	629 К	116 ms	863 К	564 ms	932 К																																																																																																																																																			
Insertion sort	12 ms	512 К	38 ms	641 К	116 ms	849 К	556 ms	928 К																																																																																																																																																			
Quick sort	22 ms	389 К	102 ms	454 К	412 ms	612 К	732 ms	730 К																																																																																																																																																			
Comb sort	12 ms	505 К	37 ms	632 К	117 ms	854 К	560 ms	936 К																																																																																																																																																			
Название Сортировки	10 элементов		50 элементов		200 элементов		1000 элементов																																																																																																																																																				
	Время	Память	Время	Память	Время	Память	Время	Память																																																																																																																																																			
Selection sort	10 ms	501 К	32 ms	612 К	113 ms	823 К	498 ms	942 К																																																																																																																																																			
Bubble sort	9 ms	498 К	32 ms	601 К	110 ms	812 К	509 ms	939 К																																																																																																																																																			
Insertion sort	9 ms	482 К	31 ms	597 К	112 ms	802 К	502 ms	920 К																																																																																																																																																			
Quick sort	21 ms	321 К	104 ms	409 К	249 ms	610 К	399 ms	874 К																																																																																																																																																			
Comb sort	10 ms	498 К	33 ms	563 К	116 ms	601 К	505 ms	907 К																																																																																																																																																			
	Selection sort	Bubble sort	Insertion sort	Quick sort	Comb sort																																																																																																																																																						
10 элементов	13	11	12	22	12																																																																																																																																																						
50 элементов	37	37	38	102	37																																																																																																																																																						
200 элементов	118	116	116	412	117																																																																																																																																																						
1000 элементов	550	564	556	732	560																																																																																																																																																						



#### Вывод:

Для сортировки неотсортированного массива наиболее оптимальным является быстрая сортировка. Несмотря на более длительное время выполнения алгоритм потребляет меньше памяти, что может быть важным в крупных проектах. Однако такие алгоритмы как сортировка выбором, обменом и вставками могут лучше подойти для научных целей, например, в обучении, где не нужно обрабатывать огромное количество данных. При частично отсортированном массиве результаты не сильно отличаются, все алгоритмы сортировки показывают время примерно на 2-3 миллисекунды меньше. Однако при сортировке частично отсортированного массива быстрая сортировка срабатывает намного быстрее и потребляет меньшее количество памяти.

#### Критерии оценки

Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 3**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Тема 1.1.3. Объектно-ориентированное программирование</b>		<p><b>1. Основные принципы объектно-ориентированного программирования. Классы: основные понятия.</b> Понятие класса. Члены класса. Понятие объекта. Инкапсуляция. Наследование. Полиморфизм.</p> <p><b>2. Перегрузка методов.</b> Сигнатура метода. Понятие конструктора.</p> <p><b>3. Операции класса.</b> Сигнатура операции. Список параметров.</p> <p><b>4. Иерархия классов.</b> Наследование классов. Понятие абстрактного класса.</p> <p><b>5. Синтаксис интерфейсов. Интерфейсы и наследование.</b> Понятие интерфейса. Множественное наследование.</p> <p><b>6. Структуры.</b> Понятие структуры. Использование структур.</p> <p><b>7. Делегаты.</b> Понятие делегата. Общие сведения о делегатах. Использование делегатов.</p> <p><b>8. Регулярные выражения.</b> Синтаксис регулярных выражений. Класс Regex. Пространство имен System.Text.RegularExpressions.</p> <p><b>9. Коллекции. Параметризованные классы.</b> Создание и применение коллекций. Универсальные классы. Универсальные шаблоны.</p> <p><b>10. Указатели.</b> Хранение переменных. Понятие адреса. Выделение памяти. Понятие указателя. Динамическая память. Размер кучи, память кучи. Стек. Операции присваивания и разыменования. Инструментальные средства для работы с динамической памятью. Соответствие типов указателей. Передача указателей функциям. Область видимости. Операции с указателями. Массивы указателей.</p> <p><b>11. Операции со списками.</b> Понятие списка. Хранение списка в памяти. Создание пустого списка. Создание, изменение, удаление списков и работа с его элементами. Добавление в список нового элемента. Поиск элемента в списке. Удаление элемента из списка. Добавление элемента в список после заданного элемента. Сортировка списков. Слияние списков.</p>
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2

	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с. Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a> Дополнительные источники: 1. Подбельский В. Язык C#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342
<b>Вариант</b>		1. Создать класс Tiles (кафель), который будет содержать поля с открытым доступом: brand, size_h, size_w, price и метод класса getData(). В главной функции объявить пару объектов класса и внести данные в поля. Затем отобразить их, вызвав метод getData(). 2. Для решения задачи описать класс Traveler, создать объект класса. Вычислить по формуле значение t для данного объекта. Задача 2. После того, как британскими учеными было доказано, что Земля полая, отважный путешественник решил узнать, что находится в центре Земли. Для этого он проделал сквозное отверстие в Земле и,снарядившись видеокамерой, прыгнул навстречу приключениям. Известно, что радиус Земли равен 6 371 км. Ускорение свободного падения $g = 9,8 \text{ м/с}^2$ . Программа также должна спрашивать и выводить имя путешественника, название его страны, города. Рассчитать, через сколько дней и часов путешественник достигнет своей цели. Определить, успеет ли путешественник вернуться до Нового года, если до него осталась 1 неделя. $t=S/g$ , где t – время полётопадения, S – расстояние в метрах, g – ускорение свободного падения. 3. Определить класс Children, который содержит такие поля (члены класса): закрытые — имя ребенка, фамилию и возраст, публичные — методы ввода данных и отображения их на экран. Объявить два объекта класса, внести данные и показать их. 4. Преобразовать строки двумерного массива в столбцы. Класс должен содержать два метода: один заполнит массив значениями, второй произведет замену значений строк на значения столбцов.
<b>Пакет преподавателя</b>		Проверяется правильность выполнения задания, согласно критериям 1.



```

1 #include <iostream>
2 using namespace std;
3
4 class Tiles
5 {
6 public:
7     char brand[32];
8     int size_h; // размер в высоту
9     int size_w; // размер в ширину
10    double price;
11    void getData();
12 };
13
14 int main()
15 {
16     setlocale(LC_ALL, "rus");
17
18     Tiles SaloniCeramica;
19     strcpy_s(SaloniCeramica.brand, "SaloniCeramica");
20     SaloniCeramica.size_h = 10;
21     SaloniCeramica.size_w = 10;
22     SaloniCeramica.price = 30;
23
24     Tiles PorcellanaDiRocca;
25     strcpy_s(PorcellanaDiRocca.brand, "PorcellanaDiRocca");
26     PorcellanaDiRocca.size_h = 20;
27     PorcellanaDiRocca.size_w = 30;
28     PorcellanaDiRocca.price = 25;
29
30     SaloniCeramica.getData();
31     PorcellanaDiRocca.getData();
32
33     return 0;
34 }
35
36 void Tiles::getData()
37 {
38     cout << brand << ": " << size_h << 'x' << size_w << " - " << price;
39 };

```

```

public class Traveller
{
    public string sName;
    public string sCountry;
    public string sCity;
}

private void button1_Click(object sender, EventArgs e)
{
    Traveller traveller = new Traveller();
    traveller.sName = textBox1.Text;
    traveller.sCity = textBox2.Text;
    traveller.sCountry = textBox3.Text;

    double dTime;
    double S = 6371000;
    double G = 9.8;
    dTime = ((S / G) / 60) / 60;

    label4.Text = "Путешественник по имени " + traveller.sName + "\n из города " + traveller.sCity
        + " страны " + traveller.sCountry + "\n с момента начала полёта прошло \n " + dTime + " часов или " + dTime / 24 + " дней";
    if (168 <= dTime) label5.Text = "К сожалению " + traveller.sName + " не успел";
    label4.Visible = true;
    label5.Visible = true;
}

```

2.



```

1  #include <iostream>
2  using namespace std;
3
4  class Children
5  {
6      char name[32]; // поля закрытые по умолчанию
7      char surname[32];
8      int age;
9  public: // открытые поля
10     void fillData();
11     void showData();
12 };
13
14 int main()
15 {
16     setlocale(LC_ALL, "rus");
17
18     Children FirstChild;
19     Children SecondChild;
20
21     cout << "Внесите данные!\n";
22     FirstChild.fillData();
23     SecondChild.fillData();
24
25     FirstChild.showData();
26     SecondChild.showData();
27
28     return 0;
29 }
30 // определение методов класса
31 void Children::fillData()
32 {
33     cout << "Имя: ";
34     cin.getline(name, 32);
35     cout << "Фамилия: ";
36     cin.getline(surname, 32);
37     cout << "Возраст: ";
38     cin >> age;
39     cin.get();
40 }
41 //=====
42 void Children::showData()
43 {
44     cout << name << " " << surname << " " << age << " лет;\n";
45 }

```

3.

4.

```

#include <iostream>
using namespace std;
class Matrix
{
    int ** matrixInClass;
public:
    void setMatrix(int rowAmount, int colAmount);
    void changeRowAndColumn(int rowAmount, int
colAmount);
};
void Matrix::setMatrix(int rowAmount, int colAmount) //
заполнение массива данными
{
    matrixInClass = new int*[rowAmount]; // выделяем память
для матрицы
    for (int i = 0; i < rowAmount; i++)
    {
        matrixInClass[i] = new int[colAmount];
    }
    for (int i = 0; i < rowAmount; i++) // записываем значения в
массив
    {
        cout << " | ";
        for (int j = 0; j < colAmount; j++)
        {
            matrixInClass[i][j] = i + j;
            cout << matrixInClass[i][j] << " ";
        }
    }
}

```

```

        cout << " | " << endl;
    }
}
void Matrix::changeRowAndColumn(int rowAmount, int
colAmount)
{
    int** tempMatrix = new int*[colAmount]; // выделяем
    память для временной матрицы
    for (int i = 0; i < colAmount; i++)
    {
        tempMatrix[i] = new int[rowAmount];
    }
    for (int i = 0; i < colAmount; i++) // копируем столбцы в
    строки, а строки в столбцы
    {
        for (int j = 0; j < rowAmount; j++)
        {
            tempMatrix[i][j] = matrixInClass[j][i];
        }
        cout << endl;
    }
    for (int i = 0; i < rowAmount; i++) // Освобождаем память
    перед выделением новой
    {
        delete[] matrixInClass[i];
    }
    delete[] matrixInClass;
    matrixInClass = new int*[colAmount]; // выделяем новую
    память
    for (int i = 0; i < colAmount; i++)
    {
        matrixInClass[i] = new int[rowAmount];
    }
    for (int i = 0; i < colAmount; i++) // копируем из временной
    матрицы
    {
        cout << "|";
        for (int j = 0; j < rowAmount; j++)
        {
            matrixInClass[i][j] = tempMatrix[i][j];
            cout << matrixInClass[i][j] << " ";
        }
        cout << "|" << endl;
    }
    for (int i = 0; i < colAmount; i++) // Освобождаем память
    временной матрицы
    {
        delete[] tempMatrix[i];
    }
    delete[] tempMatrix;
}
int main()
{
    setlocale(LC_ALL, "rus");
    int rowAmount;
    int colAmount;
    cout << "Введите количество строк двумерного массива:

```

	<pre> ";     cin &gt;&gt; rowAmount;     cout &lt;&lt; "Введите количество столбцов двумерного массива: ";     cin &gt;&gt; colAmount;     Matrix Object;     Object.setMatrix(rowAmount, colAmount);     cout &lt;&lt; "\nЗамена значений строк на значения столбцов:";     Object.changeRowAndColumn(rowAmount, colAmount);     return 0; } </pre>	
Критерии оценки	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 4**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Тема 1.1.4. Паттерны проектирования</b>		<p><b>1. Назначение и виды паттернов.</b> Понятие паттерна проектирования. Порождающие паттерны. Структурные паттерны. Паттерны поведения.</p> <p><b>2. Основные шаблоны:</b> Шаблон делегирования, шаблон функционального дизайна, неизменяемый интерфейс, интерфейс, интерфейс-маркер, контейнер свойств, канал событий.</p> <p><b>3. Порождающие шаблоны.</b> Назначение порождающих шаблонов. Фабрика объектов. Виртуальный конструктор.</p> <p><b>4. Структурные шаблоны.</b> Назначение структурных шаблонов. Примеры структурных шаблонов.</p> <p><b>5. Поведенческие шаблоны.</b> Назначение поведенческих шаблонов. Особенности паттернов поведения.</p>
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания

	<p>1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с.</p> <p>Электронные издания (электронные ресурсы)</p> <p>1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a></p> <p>Дополнительные источники:</p> <p>1. Подбельский В. Язык С#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342</p>	
<b>Вариант</b>	Используя Fabric Method (Порождающий шаблон проектирования), распределить обязанности менеджера по найму дочерним классам.	
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям</p> <p>Пример решения:</p> <p>Изначально у нас есть интерфейс Interviewer и несколько реализаций для него:</p> <pre>interface Interviewer {     public function askQuestions(); } class Developer implements Interviewer {     public function askQuestions()     {         echo 'Задаёт вопрос';     } } class CommunityExecutive implements Interviewer {     public function askQuestions()     {         echo 'Спрашивает о работе с сообществом';     } }</pre> <p>Создание HiringManager</p> <pre>abstract class HiringManager {     // Фабричный метод     abstract public function makeInterviewer(): Interviewer;     public function takeInterview()     {         \$interviewer = \$this-&gt;makeInterviewer();         \$interviewer-&gt;askQuestions();     } }</pre> <p>И теперь любой дочерний класс может расширять его и предоставлять необходимого интервьюера:</p> <pre>class DevelopmentManager extends HiringManager {     public function makeInterviewer(): Interviewer     {         return new Developer();     } } class MarketingManager extends HiringManager {     public function makeInterviewer(): Interviewer     {         return new CommunityExecutive();     } }</pre> <p>Пример использования:</p> <pre>\$devManager = new DevelopmentManager(); \$devManager-&gt;takeInterview(); // \$marketingManager = new MarketingManager(); \$marketingManager-&gt;takeInterview(); // Вывод: Спрашивает о работе с сообществом</pre>	
<b>Критерии оценки</b>	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.

	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 5**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Тема 1.1.5. Событийно-управляемое программирование</b>		<p><b>1. Событийно-управляемое программирование.</b> Сфера применения. Применение в серверных приложениях: мультиплексирование. Применение в настольных приложениях. Языки событийно-управляемого программирования. Инструменты и библиотеки для событийно-ориентированного программирования.</p> <p><b>2. Элементы управления. Диалоговые окна. Обработчики событий.</b> Создание экземпляра элемента управления. Изменение внешнего вида элемента управления. Создание стиля для элемента управления. Создание шаблона ControlTemplate. Работа с диалоговыми окнами. Обработка и вызов событий. Данные событий. Обработчики статических и динамических событий. Создание нескольких событий одним классом.</p> <p><b>3. Введение в графику.</b> Построение пользовательских интерфейсов.</p>
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование:

	– ПК, ПО
<b>Источники</b>	<p>Основные источники:</p> <p>Печатные издания</p> <p>1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с.</p> <p>Электронные издания (электронные ресурсы)</p> <p>1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a></p> <p>Дополнительные источники:</p> <p>1. Подбельский В. Язык C#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342</p>
<b>Вариант</b>	Разработать полиформное игровое приложение с анимацией.
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям</p> <p>Пример описание класса</p> <pre>using System.Collections; using System.Collections.Generic; using UnityEngine; using UnityEngine.SceneManagement; using UnityEngine.UI; using UnityEngine.Audio;</pre> <pre>public class GameController : MonoBehaviour {      public TouchToPause touchToPauseImage;     public GameObject playerObject;     public AudioClip[] bg_loops;     private AudioSource audioSource;     public AudioManager audioMixer;     public float waitBeforeFirstWave;     public int gameScore;     public Text gameScoreText;     public Image bossWarningImage;     public bool gameOver;     public Wave[] waves;     private int waveIndex;     private int bossWaveIndex;     private bool firstWaveSpawned;     private bool firstBossWaveSpawned;     public bool canControl;     public bool canFire;     public PlayerWaypoint[] startLVLWaypoints;     public PlayerWaypoint[] bossAlarmWaypoints;     public Wave[] bossWaves;      public Image youWonImg;     public Image youLostImg;      void Start () {         gameOver = false;         canControl = false;         waveIndex = 0;     } }</pre>



```

    bossWaveIndex = 0;
    audioSource = GetComponent<AudioSource>();
    audioSource.clip = bg_loops[Random.Range(0, bg_loops.Length)];
    audioSource.Play();
    StartCoroutine(StartLevelCutScene());
}
public void GameOver(bool player = false)
{
    if (!player)
    {
        int remainingEnemiesTotal = 0;
        foreach(Wave wave in waves)
        {
            remainingEnemiesTotal += wave.EnemiesAlive();
        }
        if (remainingEnemiesTotal > 0 || waves.Length > waveIndex) { return; }
        int remainingBossEnemiesTotal = 0;
        foreach(Wave wave in bossWaves)
        {
            remainingBossEnemiesTotal += wave.EnemiesAlive();
        }
        if (!firstBossWaveSpawned)
        {
            if (bossWaves.Length != 0)
            {
                touchToPauseImage.UnPause();
                StartCoroutine(BossAlarm());
                StartCoroutine(BossFight());
            } else {
                touchToPauseImage.UnPause();
                StartCoroutine(YouWon());
            }
        } else {
            if (remainingBossEnemiesTotal > 0 || bossWaves.Length > bossWaveIndex) {
                return; }
            Debug.Log("GAME OVER!");
        }

        } else {
        StartCoroutine(YouLost());
    }
}
public void UpdateGameScore(int score)
{
    gameScore += score;
    gameScoreText.text = gameScore.ToString("D6");
}
void SpawnFirstWave()
{
    waves[waveIndex].SpawnWave();
    waveIndex += 1;
    firstWaveSpawned = true;
}
public void SpawnNextWave()
{
    if (firstWaveSpawned)
    {
        if(waveIndex < waves.Length)

```

```

        {
            waves[waveIndex].SpawnWave();
            waveIndex += 1;
        }
        else if (firstBossWaveSpawned && bossWaveIndex <
bossWaves.Length)
        {
            bossWaves[bossWaveIndex].SpawnWave();
            bossWaveIndex += 1;
        }
    }
}
public void Pause()
{
    Time.timeScale = 0.05f;
    Time.fixedDeltaTime = Time.timeScale * 0.035f;
}
public void UnPause()
{
    Time.timeScale = 1f;
    Time.fixedDeltaTime = Time.timeScale * 0.035f;
}
IEnumerator StartLevelCutScene()
{
    yield return StartCoroutine(CutScene(1f, startLVLWaypoints));
    canControl = true;
    canFire = true;
    Invoke("SpawnFirstWave", waitBeforeFirstWave);
}
public IEnumerator CutScene(float waitBeforeCutScene, PlayerWaypoint[]
waypoints, int startWaypoint = 0)
{
    yield return new WaitForSeconds(waitBeforeCutScene);
    int waypointIndex = startWaypoint;
    while (waypointIndex < waypoints.Length)
    {
        Vector3 directionToWaypoint = playerObject.transform.position -
waypoints[waypointIndex].transform.position;
        while(directionToWaypoint.sqrMagnitude > 2)
        {
            playerObject.GetComponent<PlayerController>().SetVelocity(way
points[waypointIndex].speed, directionToWaypoint);
            yield return new WaitForSeconds(0.2f);
            directionToWaypoint = playerObject.transform.position -
waypoints[waypointIndex].transform.position;
        }
        waypointIndex += 1;
        yield return new WaitForSeconds(0.2f);
    }
    playerObject.GetComponent<PlayerController>().SetVelocity(0,
Vector3.zero);
}
IEnumerator BossAlarm()
{
    canFire = false;
    bossWarningImage.gameObject.SetActive(true);
    yield return StartCoroutine(CutScene(0f, bossAlarmWaypoints));
}

```

	<pre> yield return new WaitForSeconds(3f);     bossWarningImage.GetComponent&lt;Animator&gt;().Play("BossDisassemble Canvas");     yield return new WaitForSeconds(1f);     bossWarningImage.gameObject.SetActive(false);     canFire = true; } IEnumerator BossFight() {     yield return new WaitForSeconds(1f);     bossWaves[bossWaveIndex].SpawnWave();     bossWaveIndex += 1;     firstBossWaveSpawned = true; } IEnumerator YouWon() {     int countdown = 3;     gameOver = true;     canControl = false;     canFire = false;     Pause();     youWonImg.SetActive(true);     yield return new WaitForSeconds(0.05f);     while (countdown &gt; -1)     {         Debug.Log("You won! New game will start in " + countdown + " seconds");         countdown -= 1;         yield return new WaitForSeconds(0.05f);     }     UnPause();     Scene loadedLevel = SceneManager.GetActiveScene ();     SceneManager.LoadScene (loadedLevel.buildIndex); } IEnumerator YouLost() {     int countdown = 3;     gameOver = true;     canControl = false;     canFire = false;     Pause();     yield return new WaitForSeconds(0.05f);     while (countdown &gt; -1)     {         Debug.Log("You lost. New game will start in " + countdown + " seconds");         countdown -= 1;         yield return new WaitForSeconds (0.05f);     }     UnPause();     Scene loadedLevel = SceneManager.GetActiveScene ();     SceneManager.LoadScene (loadedLevel.buildIndex); } } </pre>		
Критерии оценки	<table border="1"> <tr> <td data-bbox="531 2000 874 2080">Отлично</td><td data-bbox="874 2000 1482 2080">Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или</td></tr> </table>	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или
Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или		

		образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 6**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Тема 1.1.6.</b> <b>Оптимизация и рефакторинг кода</b>		<p><b>1. Методы оптимизации программного кода.</b> Мемоизация и кэширование. Распараллеливание. Распределение нагрузки. Ленивые вычисления. Отложенные расчеты. Аппроксимация.</p> <p><b>2. Цели и методы рефакторинга.</b> Цель рефакторинга. Соотношение рефакторинга с оптимизации производительности и реинжинирингом. Причины применения рефакторинга. Проблемы в коде, требующие рефакторинга. Методы рефакторинга: изменение сигнатуры метода, инкапсуляция поля, выделение класса, выделение интерфейса, выделение локальной переменной.</p>
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее

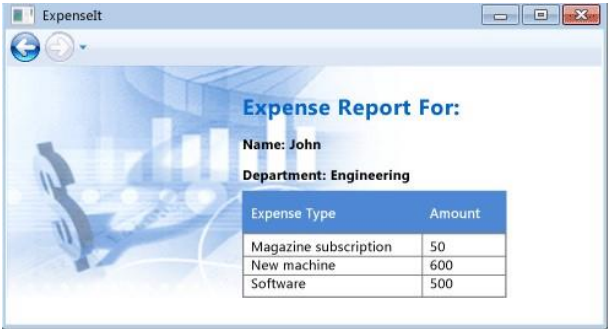
	<p>профессиональное образование, профессиональная подготовка / Г.Н Федорова. – М.: Академия, 2016. – 336 с.</p> <p>Электронные издания (электронные ресурсы)</p> <p>1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a></p> <p>Дополнительные источники:</p> <p>1. Подбельский В. Язык C#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342</p>
<b>Вариант</b>	<p>1. Выполнить оптимизацию кода следующей программы.</p> <p>Структура <i>vec</i> является вектором элементов типа <i>float</i>.</p> <p>Функция <i>combine0</i> вычисляет результат перемножения всех элементов вектора. Эту функцию нужно оптимизировать. Размер массива = 5000, он инициализирован случайными числами.</p> <pre>typedef struct { long len; float *data; } vec; long vec_len(vec *v) { return v-&gt;len; } void combine0(vec *v, float *dest) { long i; *dest = 1; for (i = 0; i &lt; vec_len(v); i++) { *dest *= v-&gt;data[i]; } }</pre> <pre>#define SIZE 5000 float a[SIZE]; vec v = {SIZE, a}; int main() { float res; for (i = 0; i &lt; SIZE; i++) // инициализация вектора случайными числами a[i] = rand(); combine0(&amp;v, &amp;res); }</pre>
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям</p> <p>1. Избавление от неэффективности цикла</p> <pre>void combine1(vec *v, float *dest) { long i, len = vec_len(v); *dest = 1; for (i = 0; i &lt; len; i++) { *dest *= v-&gt;data[i]; } }</pre> <pre>void lower(char *s) { for (long i = 0; i &lt; strlen(s); i++) if (s[i] &gt;= 'A' &amp;&amp; s[i] &lt;= 'Z') s[i] -= ('A' - 'a'); }</pre> <p>Уменьшение количества обращений к памяти</p> <pre>void combine2(vec *v, float *dest) { long i, len = vec_len(v); float acc = 1;</pre>

	<pre> for (i = 0; i &lt; len; i++) { acc *= v-&gt;data[i]; } *dest = acc; }  Раскрутка цикла с несколькими аккумуляторами: float combine3(float a[], long size) { long i, limit = size-1; float acc0 = 1; float acc1 = 1; for (i = 0; i &lt; limit; i+=2) { acc0 *= a[i]; acc1 *= a[i+1]; } while (i &lt; size) acc0 *= a[i++]; return acc0 * acc1; }  Реассоциация: for (long i = 0; i &lt; limit; i+=3) { float x = a[i], y = a[i+1], z = a[i+2]; acc = acc * x * y * z; } </pre>	
Критерии оценки	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 7**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Тема 1.1.7.</b> <b>Разработка</b> <b>пользовательского</b> <b>интерфейса.</b>		<b>1. Правила разработки интерфейсов пользователя.</b> Минимализм: логическая структура, дизайн элементов, колористика, анимация. Интуитивность: иерархичность, привычная структура, использование аналогий. Адаптивность: адаптивность верстки, адаптивность контента. Анализ пользовательского графического интерфейса на примере продуктов семейства Windows.
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с. Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a>



	<p>Дополнительные источники:</p> <p>1. Подбельский В. Язык C#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342</p>
<b>Вариант</b>	<p>В IDE Visual Studio на языке C# разработать следующий интерфейс пользователя.</p> 
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям</p> <p>После того, как компоненты будут размещены на форме, необходимо добавить XAML-файл для указания пользовательского интерфейса, автоматически отображаемого при запуске приложения.</p> <pre> &lt;Application x:Class="ExpenseIt.App"     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"     StartupUri="MainWindow.xaml"&gt;     &lt;Application.Resources&gt;          &lt;/Application.Resources&gt;     &lt;/Application&gt; </pre> <p>Изменить свойства:</p> <pre> &lt;NavigationWindow x:Class="ExpenseIt.MainWindow"     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"     Title="ExpenseIt" Height="350" Width="500"&gt; &lt;/NavigationWindow&gt; </pre> <p>Код обработки события:</p> <pre> using System; using System.Collections.Generic; using System.Linq; using System.Text; using System.Windows; using System.Windows.Controls; using System.Windows.Data; using System.Windows.Documents; using System.Windows.Input; using System.Windows.Media; using System.Windows.Media.Imaging; using System.Windows.Navigation; using System.Windows.Shapes;  namespace ExpenseIt {     /// &lt;summary&gt;     /// Interaction logic for MainWindow.xaml     /// &lt;/summary&gt;     public partial class MainWindow : NavigationWindow     { </pre>

```

public MainWindow()
{
    InitializeComponent();
}
}

```

Добавление файлов в приложение:

```

<Page x:Class="ExpenseIt.ExpenseItHome"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="300"
    Title="ExpenseIt - Home">

```

```
<Grid>
```

```
</Grid>
```

```
</Page>
```

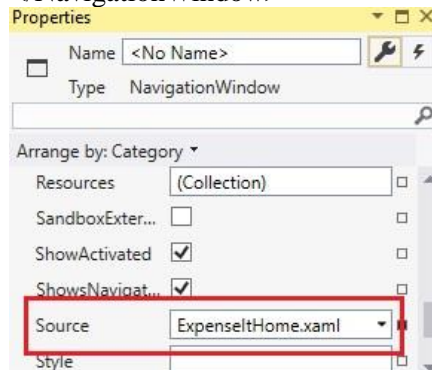
Задать первую страницу:

```

<NavigationWindow x:Class="ExpenseIt.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="ExpenseIt" Height="350" Width="500"
    Source="ExpenseItHome.xaml">

```

```
</NavigationWindow>
```



```

<Page x:Class="ExpenseIt.ExpenseReportPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="300"
    Title="ExpenseIt - View Expense">

```

```
<Grid>
```

```
</Grid>
```

```
</Page>
```

Следующий код программной части обрабатывает логику, реагирующую на действия пользователя:

```
using System;
```

```

using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

```

```

namespace ExpenseIt
{
    /// <summary>
    /// Interaction logic for ExpenseItHome.xaml
    /// </summary>
    public partial class ExpenseItHome : Page
    {
        public ExpenseItHome()
        {
            InitializeComponent();
        }
    }
}

```

И следующим образом для ExpenseReportPage:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

```

```

namespace ExpenseIt
{
    /// <summary>
    /// Interaction logic for ExpenseReportPage.xaml
    /// </summary>
    public partial class ExpenseReportPage : Page
    {
        public ExpenseReportPage()
        {
            InitializeComponent();
        }
    }
}

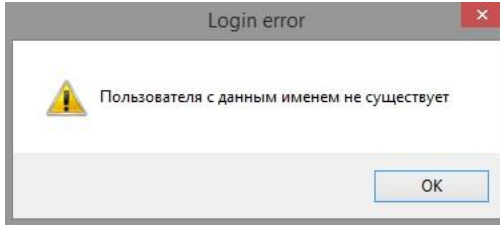
```

Добавить существующий элемент. Нажать F5.

Критерии оценки	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 8**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Тема 1.1.8.</b> <b>Основы ADO.Net</b>		<b>1. Работа с базами данных.</b> Введение в ADO.Net. Создание базы данных. Строка подключения. Создание подключения. <b>2. Доступ к данным.</b> Доступ к данным с помощью ADO.Net. Компоненты ADO.Net и объектная модель. <b>3. Создание таблицы, работа с записями.</b> Работа с данными в таблицах ADO.Net. Свод событий ADO.Net.
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с. Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a> Дополнительные источники: 1. Подбельский В. Язык C#. Базовый курс. Издание второе,

	переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342
<b>Вариант</b>	<p>Импортировать файлы *.csv в SQL Server. Создать приложение «Форма авторизации». Связать форму с базой данных. Обеспечить проверку пользовательского ввода в поля Логин и пароль, а также вывод соответствующих сообщений, например:</p> 
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям После необходимых действий с базой данных в SQL Server, необходимо выполнить оформление формы и запрограммировать событие нажатия на кнопку.</p> <pre> using System; using System.Collections.Generic; using System.ComponentModel; using System.Data; using System.Drawing; using System.Linq; using System.Text; using System.Threading.Tasks; using System.Windows.Forms; using System.Data.SqlClient; namespace Esoft {     public partial class Deals : Form     {         SqlConnection con = Resources.ConnectSql.GetSringsSql();          public Deals()         {             InitializeComponent();              dataGridView1.Columns[1].Width = 120;         }         private void Deals_Load(object sender, EventArgs e)         {             con.Open();             this.agentsTableAdapter.Fill(this.homeConnect.agents);             this.dealsTableAdapter.Fill(this.homeConnect.deals);             this.suppliesTableAdapter.Fill(this.homeConnect.supplies);             cb_supplies.SelectedIndex = -1;             cb_type_client.SelectedIndex = -1;             SqlDataAdapter sda = new             SqlDataAdapter(CommandsSql.select_id_demands(), con);             DataTable dt = new DataTable();             sda.Fill(dt);             cb_demands.DataSource = dt;             cb_demands.DisplayMember = "Id";             cb_demands.ValueMember = "Id";             cb_demands.SelectedIndex = -1;              con.Close();         }         private void refresh_grid_Click(object sender, EventArgs e)         { </pre>

```

        this.dealsTableAdapter.Fill(this.homeConnect.deals);
    }
    private void btn_show_deal_Click(object sender, EventArgs e)
    {
        panel_deal_totalSum.Visible = false;
        panelDeduction.Visible = false;
        if (panelDeal.Visible)
        {
            panelDeal.Visible = false;
        }
        else { panelDeal.Visible = true; }
        if (pc_logo.Visible) { pc_logo.Visible = false; } else { pc_logo.Visible = true;
    }
    }
    private void btn_deduction_Click(object sender, EventArgs e)
    {
        if (pc_logo.Visible) { pc_logo.Visible = false; } else { pc_logo.Visible = true;
    }
    }
    panel_deal_totalSum.Visible = false;
    panelDeal.Visible = false;
    if (panelDeduction.Visible)
    {
        panelDeduction.Visible = false;
    }
    else { panelDeduction.Visible = true; }
    label9.Visible = false;
    lblSum.Visible = false;
    label8.Visible = false;
    cb_type_object_com.Visible = false;
    button1.Visible = false;
    cb_type_client.SelectedIndex = -1;
    }
    private void btnMin_Click(object sender, EventArgs e)
    {
        this.WindowState = FormWindowState.Minimized;
    }
    private void btnExit_Click(object sender, EventArgs e)
    {
        Application.Exit();
    }
    private void btnBack_Click(object sender, EventArgs e)
    {
        Form1 f1 = new Form1();
        f1.Show();
        this.Close();
    }
    private void panel1_MouseDown(object sender, MouseEventArgs e)
    {
        panel1.Capture = false;
        var m2 = Message.Create(Handle, 0xa1, new IntPtr(2), IntPtr.Zero);
        WndProc(ref m2);
    }

    private void panel4_MouseDown(object sender, MouseEventArgs e)
    {
        panel4.Capture = false;
        var m2 = Message.Create(Handle, 0xa1, new IntPtr(2), IntPtr.Zero);
        WndProc(ref m2);
    }
    private void btn_insert_Click(object sender, EventArgs e)//INSERT
    {
        con.Open();
        SqlDataAdapter sda1 = new SqlDataAdapter("SELECT Supply_Id FROM

```

```

deals Where Id = '"+cb_supplies.Text+"', con);
    DataTable dt1 = new DataTable();
    sda1.Fill(dt1);
    SqlDataAdapter sda2 = new SqlDataAdapter("SELECT Demand_Id FROM
deals Where Id = '" + cb_demands.Text + "'", con);
    DataTable dt2 = new DataTable();
    sda2.Fill(dt2);
    try
    {
        if (dt1.Rows.Count == 0 && dt2.Rows.Count == 0)
        {
            SqlCommand com = new SqlCommand("INSERT INTO [deals] (Id,
Demand_Id, Supply_Id) VALUES(@Id,@Demand_Id, @Supply_Id)", con);
            com.Parameters.AddWithValue("Id", tb_id.Text);
            com.Parameters.AddWithValue("Demand_Id", cb_demands.Text);
            com.Parameters.AddWithValue("Supply_Id", cb_supplies.Text);
            com.ExecuteNonQuery();
        }
        else
        {
            MessageBox.Show("Предложение или потребность, которые уже
участвуют в сделке, не могут быть выбраны повторно",
            "Error insert", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
    catch
    {
        MessageBox.Show("Ошибка добавления", "Error insert",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    con.Close();
}
private void btn_upd_Click(object sender, EventArgs e)//UPDATE
{
    con.Open();
    if(!string.IsNullOrEmpty(tb_id.Text))
    {
        SqlDataAdapter sda = new SqlDataAdapter("SELECT Id FROM deals
WHERE Id= '" + tb_id.Text + "'", con);
        DataTable dt = new DataTable();
        sda.Fill(dt);
        if (dt.Rows.Count != 0)
        {
            SqlCommand com = new SqlCommand("UPDATE deals SET Id=@id,
Demand_Id=@Demand_Id, Supply_Id=@Supply_Id WHERE Id=@Id", con);
            com.Parameters.AddWithValue("Id", tb_id.Text);
            com.Parameters.AddWithValue("Demand_Id", cb_demands.Text);
            com.Parameters.AddWithValue("Supply_Id", cb_supplies.Text);

            com.ExecuteNonQuery();
        }
        else
        {
            MessageBox.Show("Сделка с идентификатором " + tb_id.Text + " не
найден ", "Error update", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
    else
    {
        MessageBox.Show("Введите Id сделки, которую хотите изменить",
        "Error update", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    con.Close();
}

```



```

    }
    private void btn_del_Click(object sender, EventArgs e)//DELETE
    {
        con.Open();
        SqlDataAdapter sda = new SqlDataAdapter("SELECT Id FROM deals
WHERE Id= " + tb_id.Text + "", con);
        DataTable dt = new DataTable();
        sda.Fill(dt);
        if(dt.Rows.Count != 0)
        {
            SqlCommand com = new SqlCommand("DELETE FROM deals WHERE
Id= " + tb_id.Text + "", con);
            com.ExecuteNonQuery();
        }
        else
        {
            MessageBox.Show("Сделка с идентификатором "+tb_id.Text+" не
найден ", "Error insert", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        con.Close();
    }
    private void button1_Click(object sender, EventArgs e)//Summa
    {
        con.Open();

        if(!string.IsNullOrEmpty(tb_id_com.Text))
        {
            if (!string.IsNullOrEmpty(cb_type_client.Text))
            {
                if (cb_type_client.Text.Equals("продавец"))
                {
                    SqlDataAdapter sda = new
                    SqlDataAdapter(CommandsSql.check_id_demands(tb_id_com.Text), con);
                    DataTable dt = new DataTable();
                    sda.Fill(dt);
                    if(dt.Rows.Count !=0)
                    {
                        if (cb_type_object_com.Text.Equals("Квартира"))
                        {
                            try
                            {
                                SqlCommand command = new SqlCommand("SELECT Price
* 0.01 + 36000 as Comissya, ClientId FROM supplies WHERE
ClientId=@ClientId", con);
                                command.Parameters.AddWithValue("ClientId",
tb_id_com.Text);

                                SqlDataReader s = command.ExecuteReader();
                                s.Read();
                                lblSum.Text = s[0].ToString();
                            }
                            catch
                            {
                                MessageBox.Show("Ошибка поиска", "Error find",
                                MessageBoxButtons.OK, MessageBoxIcon.Error);
                            }
                        }
                        else if (cb_type_object_com.Text.Equals("Дом"))
                        {
                            try
                            {
                                SqlCommand command = new SqlCommand("SELECT Price
* 0.02 + 30000 as Comissya, ClientId FROM supplies WHERE
ClientId=@ClientId", con);

```

```

        command.Parameters.AddWithValue("ClientId",
tb_id_com.Text);

        SqlDataReader s = command.ExecuteReader();
        s.Read();
        lblSum.Text = s[0].ToString();
    }
    catch
    {
        MessageBox.Show("Ошибка поиска", "Error find",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
else if (cb_type_object_com.Text.Equals("Земля"))
{
    try
    {
        SqlCommand command = new SqlCommand("SELECT Price
* 0.01 + 30000 as Comissya, ClientId FROM supplies WHERE
ClientId=@ClientId", con);
        command.Parameters.AddWithValue("ClientId",
tb_id_com.Text);

        SqlDataReader s = command.ExecuteReader();
        s.Read();
        lblSum.Text = s[0].ToString();
    }
    catch
    {
        MessageBox.Show("Ошибка поиска", "Error find",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
else
{
    MessageBox.Show("Ошибка поиска", "Error find",
    MessageBoxButtons.OK, MessageBoxIcon.Error);
}
}
else
{
    MessageBox.Show("Продавца с идентификатором " +
tb_id_com.Text + " не найдено", "Error find", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}
else if (cb_type_client.Text.Equals("покупатель"))
{
    SqlDataAdapter sda = new SqlDataAdapter("Select ClientId FROM
supplies Where ClientId='" + tb_id_com.Text + "'", con);
    DataTable dt = new DataTable();
    sda.Fill(dt);
    if (dt.Rows.Count != 0)
    {
        try
        {
            SqlCommand command = new SqlCommand("SELECT Price *
0.03 as Comissya, ClientId FROM supplies WHERE ClientId=@ClientId", con);
            command.Parameters.AddWithValue("ClientId",
tb_id_com.Text);

            SqlDataReader s = command.ExecuteReader();
            s.Read();
            lblSum.Text = s[0].ToString();
        }
        catch
    }
}

```

```

        {
            MessageBox.Show("Ошибка поиска", "Error find",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
    else
    {
        MessageBox.Show("Клиент с идентификатором " +
        tb_id_com.Text + " не является покупателем", "Error find",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
}
else
{
    MessageBox.Show("Укажите, кем является клиент", "Error insert",
    MessageBoxButtons.OK, MessageBoxIcon.Error);
}
}
else
{
    MessageBox.Show("Укажите Id", "Error insert", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
con.Close();
}
private void bnt_type_object_Click(object sender, EventArgs e)
{
    if (cb_type_client.Text.Equals("продавец"))
    {
        label9.Visible = true;
        lblSum.Visible = true;
        label8.Visible = true;
        cb_type_object_com.Visible = true;
        button1.Visible = true;
    }
    else if (cb_type_client.Text.Equals("покупатель"))
    {
        label9.Visible = false;
        lblSum.Visible = true;
        label8.Visible = true;
        cb_type_object_com.Visible = false;
        button1.Visible = true;
    }
    }
    else
    {
        label9.Visible = false;
        lblSum.Visible = false;
        label8.Visible = false;
        cb_type_object_com.Visible = false;
        button1.Visible = false;
    }
}
private void btn_deal_totalSum_Click(object sender, EventArgs e)
{
    if (pc_logo.Visible) { pc_logo.Visible = false; } else { pc_logo.Visible = true;
}

panelDeduction.Visible = false;
panelDeal.Visible = false;

if (panel_deal_totalSum.Visible)
{
    panel_deal_totalSum.Visible = false;
}

```

```

    }
    else
    {
        panel_deal_totalSum.Visible = true;
    }
}
private void btn_sumTotal_Click(object sender, EventArgs e)
{
    con.Open();

    if(!string.IsNullOrEmpty(tb_idDeal_total.Text))
    {
        SqlDataAdapter sda = new SqlDataAdapter("SELECT Id FROM deals
WHERE Id='"+tb_idDeal_total.Text+"'",con);
        DataTable dt = new DataTable();
        sda.Fill(dt);
        if(dt.Rows.Count != 0)
        {
            SqlCommand com1 = new SqlCommand("SELECT Price * 0.03,
ClientId FROM supplies WHERE Id='"+ tb_idDeal_total.Text + "'", con);
            SqlDataReader r = com1.ExecuteReader();
            r.Read();
            lbl_clUp_sum.Text = r[0].ToString();
            r.Close();

            SqlDataAdapter sda1 = new SqlDataAdapter("select DealShare from
agents", con);//if realtor have %
            DataTable dt1 = new DataTable();
            sda1.Fill(dt1);
            if (dt1.Rows.Count != 0)
            {
                //for raltor of customer
                SqlCommand comR = new
SqlCommand(CommandsSql.cRb(tb_idDeal_total.Text) , con);
                SqlDataReader rR = comR.ExecuteReader();
                rR.Read();
                lbl_r_ClUp.Text = rR[0].ToString();
                rR.Close();
                SqlCommand comRd = new SqlCommand("SELECT Price * 0.03,
ClientId FROM supplies WHERE Id='"+ tb_idDeal_total.Text + "'", con);
                SqlDataReader rRd = comRd.ExecuteReader();
                rRd.Read();
                rRd.Close();
                SqlCommand company = new
SqlCommand(CommandsSql.company(tb_idDeal_total.Text), con);
                SqlDataReader rCompany = company.ExecuteReader();
                rCompany.Read();
                lbl_company_sum.Text = rCompany[1].ToString();
                rCompany.Close();
            }
        }
        else
        {
            SqlDataAdapter checksda = new
SqlDataAdapter(CommandsSql.check_id(tb_idDeal_total.Text), con);
            DataTable dtcheck = new DataTable();
            checksda.Fill(dtcheck);
            if(dtcheck.Rows.Count > 0 )
            {
                SqlDataAdapter checkAp = new SqlDataAdapter("SELECT Id
FROM apartment_demands WHERE Id = '" + tb_idDeal_total.Text + "'", con);
                SqlDataAdapter checkHo = new SqlDataAdapter("SELECT Id
FROM house_demands WHERE Id = '" + tb_idDeal_total.Text + "'", con);
                SqlDataAdapter checkLa = new SqlDataAdapter("SELECT Id

```

```

FROM land_demands WHERE Id = "" + tb_idDeal_total.Text + "", con);
        DataTable dtAp = new DataTable();
        DataTable dtHo = new DataTable();
        DataTable dtLa = new DataTable();
        checkAp.Fill(dtAp);
        checkAp.Fill(dtHo);
        checkAp.Fill(dtLa);
        if (dtAp.Rows.Count > 0)
        {
            SqlCommand comAp = new
SqlCommand(CommandSql.cmsNullbuyer_ap(tb_idDeal_total.Text), con);
            SqlDataReader rAp = comAp.ExecuteReader();
            rAp.Read();
            lbl_clDown_sum.Text = rAp[0].ToString();
            lbl_r_ClDown.Text = rAp[1].ToString();
            SqlCommand comR = new
SqlCommand(CommandSql.cmsNull1(tb_idDeal_total.Text), con);
            SqlDataReader rR = comR.ExecuteReader();
            rR.Read();
            lbl_r_ClUp.Text = rR[0].ToString();
            rR.Close();

        }
        else if (dtHo.Rows.Count > 0)
        {
            SqlCommand comHo = new
SqlCommand(CommandSql.cmsNullbuyer_ho(tb_idDeal_total.Text), con);
            SqlDataReader rHo = comHo.ExecuteReader();
            rHo.Read();
            lbl_clDown_sum.Text = rHo[0].ToString();
            lbl_r_ClDown.Text = rHo[1].ToString();
            SqlCommand comR = new
SqlCommand(CommandSql.cmsNull2(tb_idDeal_total.Text), con);
            SqlDataReader rR = comR.ExecuteReader();
            rR.Read();
            lbl_r_ClUp.Text = rR[0].ToString();
            rHo.Close();

        }
        else if (dtLa.Rows.Count > 0)
        {
            SqlCommand comLa = new
SqlCommand(CommandSql.cmsNullbuyer_la(tb_idDeal_total.Text), con);
            SqlDataReader rLa = comLa.ExecuteReader();
            rLa.Read();
            lbl_clDown_sum.Text = rLa[0].ToString();
            lbl_r_ClDown.Text = rLa[1].ToString();
            SqlCommand comR = new
SqlCommand(CommandSql.cmsNull3(tb_idDeal_total.Text), con);
            SqlDataReader rR = comR.ExecuteReader();
            rR.Read();
            lbl_r_ClUp.Text = rR[0].ToString();
            rLa.Close();

        }
        else
        {
            lbl_clDown_sum.Text = "Недвижимости не существует.";
            lbl_r_ClDown.Text = "Недвижимости не существует.";
        }
    }
    else
    {
        lbl_clDown_sum.Text = "Недвижимость не участвует в
сделке.";
    }
}

```

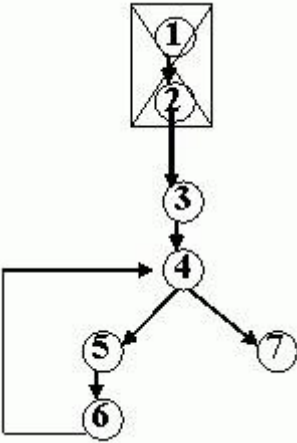
	<pre> lbl_r_CIDown.Text = "Недвижимость не участвует в сделке."; } SqlCommand company = new SqlCommand(CommandsSql.company_null_rieltor(tb_idDeal_total.Text), con); SqlDataReader rCompany = company.ExecuteReader(); rCompany.Read(); lbl_company_sum.Text = rCompany[1].ToString(); rCompany.Close(); } } else {     MessageBox.Show("Сделки с идентификатором "+tb_idDeal_total.Text+" не существует", "Error find", MessageBoxButtons.OK, MessageBoxIcon.Error); } } else {     MessageBox.Show("Введите Id сделки", "Error find", MessageBoxButtons.OK, MessageBoxIcon.Error); }  con.Close(); } } } </pre>	
Критерии оценки	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 9**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Раздел 1.2</b> <b>Поддержка и тестирование программных модулей</b>		<b>МДК.01.02 Поддержка и тестирование программных модулей</b>
<b>Тема 1.2.1 Отладка и тестирование программного обеспечения</b>		1. Тестирование как часть процесса верификации программного обеспечения. 2. Виды ошибок. Методы отладки. 3. Методы тестирования. 4. Классификация тестирования по уровням. 5. Тестирование производительности 6. Регрессионное тестирование.
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с.

	<p>Электронные издания (электронные ресурсы)</p> <p>1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a></p> <p>Дополнительные источники:</p> <p>1. Подбельский В. Язык С#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342</p>
<b>Вариант</b>	<p>1. Дана функция, которая вычисляет неотрицательную степень <math>n</math> числа <math>x</math>.</p> <pre> 1 double Power(double x, int n){ 2 double z=1; int i; 3 for (i=1; 4 n&gt;=i; 5 i++) 6 {z = z*x;} /* Возврат в п.4 */ 7 return z;} </pre> <p>Схематично нарисовать управляющий граф программы, отображающий поток управления программы. Определить пути и ветви.</p> <p>2. Определить для следующей функции реализуемые и нереализуемые пути.</p> <pre> float H(float x,float y) { float H; 1 if (x*x+y*y+2&lt;=0) 2 H = 17; 3 else H = 64; 4 return H*H+x*x; } </pre> <p>3. Написать основной фрагмент схемы программы управления схватом робота, где интервал между моментами срабатывания схвата не определен.</p> <p>4. Вставить оператор протоколирования в исходный текст метода Power:</p> <pre> double Power(double x, int n) { double z=1; int i; for (i=1;n&gt;=i;i++) { z = z*x; } return z; } </pre> <p>5. Выполнить тестирование всех значений переменных после выполнения каждого оператора:</p> <pre> double PowerNonNeg(double x, int n) { double z=1; int i; if (n&gt;0) { for (i=1;n&gt;=i;i++) { z = z*x; x,z,n,i); } } } </pre>



	<pre>     }     }     else printf(         "Ошибка! Степень числа n должна быть больше 0.\n");     return z;     } </pre>
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям 1.</p>  <p>Примеры путей: (3,4,7), (3,4,5,6,4,5,6), (3,4), (3,4,5,6).  Примеры ветвей: (3,4), (4,5,6,4), (4,7).  2. путь (1,3,4) реализуем, путь (1,2,4) нереализуем в условиях нормальной работы. Но при сбоях даже нереализуемый путь может реализоваться.  3. а)  // Прочитать значения датчика  static public bool ReadSensor(bool Sensor)  {      //...чтение значения датчика      Console.WriteLine("...reading sensor value");      return Sensor;  }    // Открыть схват  static public void OpenHand()  {      //...открываем схват      Console.WriteLine("...opening hand");  }    // Закрыть схват  static public void CloseHand()  {      //...закрываем схват      Console.WriteLine("...closing hand");  }    [STAThread]  static void Main(string[] args)  {      while (true)      {          Console.WriteLine("Enter Sensor value (true/false)"); </p>

```

        if (ReadSensor(Convert.ToBoolean(Console.ReadLine())))
        {
            OpenHand();
            CloseHand();
        }
    }
}
3.б)
#include <stdio.h>

/* Прочитать значения датчика */
int ReadSensor(int Sensor)
{
    /* ...чтение значения датчика */
    printf("...reading sensor value\n");
    return Sensor;
}

/* Открыть схват */
void OpenHand()
{
    /* ...открываем схват */
    printf("...opening hand\n");
}

/* Закрыть схват */
void CloseHand()
{
    /* ...закрываем схват */
    printf("...closing hand\n");
}

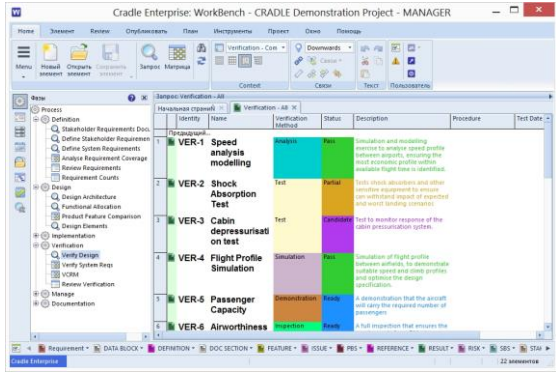
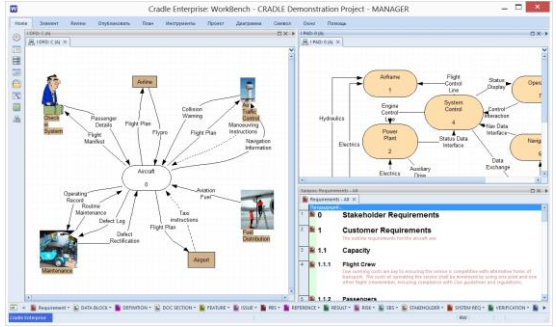

void main(void)
{
    int s;
    while (1)
    {
        printf("Enter Sensor value (0/1)");
        scanf("%d",&s);
        if (ReadSensor(s))
        {
            OpenHand();
            CloseHand();
        }
    }
}
4.
double Power(double x, int n)
{
    double z=1;
    int i;
    for (i=1;n>=i;i++)
    {
        z = z*x;
        printf("i = %d z = %f\n",i,z);
    }
    return z;
}

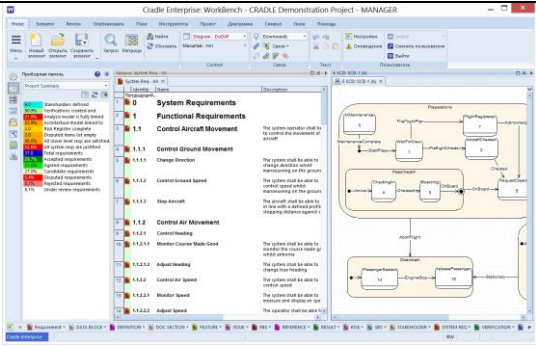
```

	<pre> } 5. double PowerNonNeg(double x, int n) {     double z=1;     int i;     printf("x=%f z=%f n=%d\n",x,z,n);     if (n&gt;0)     {         printf("x=%f z=%f n=%d\n",x,z,n);         for (i=1;n&gt;=i;i++)         {             z = z*x;             printf("x=%f z=%f n=%d i=%d\n",                 x,z,n,i);         }     }     else printf(         "Ошибка! Степень числа n должна быть больше 0.\n");     return z; } </pre>	
Критерии оценки	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 10**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**


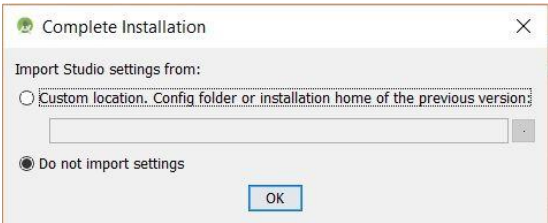
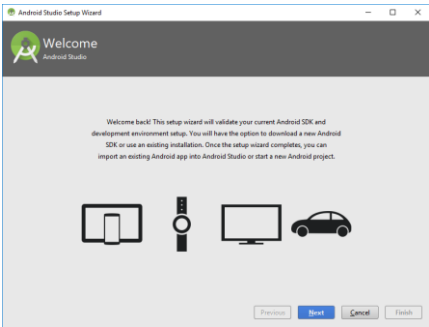
<b>Тема 1.2.2. Документирование</b>		1. Средства разработки технической документации. Технологии разработки документов. 2. Документирование программного обеспечения в соответствии с Единой системой программной документации. 3. Автоматизация разработки технической документации Автоматизированные средства оформления документации
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с. Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a> Дополнительные источники: 1. Подбельский В. Язык С#. Базовый курс. Издание второе,

	переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342
<b>Вариант</b>	Оформить документацию на программный продукт с использованием инструментальных средств
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям</p> <ol style="list-style-type: none"> <li>1. Загрузить исходные требования Заказчика из документов Word.</li> <li>2. Разработать требования в свободной текстовой форме или с помощью заданных форм (полей), например, в форме User Story, Use Cases.</li> </ol>  <ol style="list-style-type: none"> <li>3. Разработать модель UML.</li> <li>4. Структурировать требования с помощью различных типов связей.</li> </ol>  <ol style="list-style-type: none"> <li>5. Распределить права доступа и ограничить видимость или возможность редактирования проектных данных.</li> </ol>  <ol style="list-style-type: none"> <li>6. Сгенерировать готовые документы по шаблону, включающему требования, модели, организованные в разделы технического задания.</li> </ol>

		
Критерии оценки	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

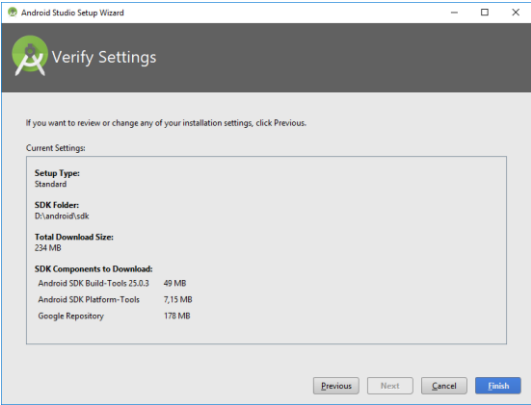
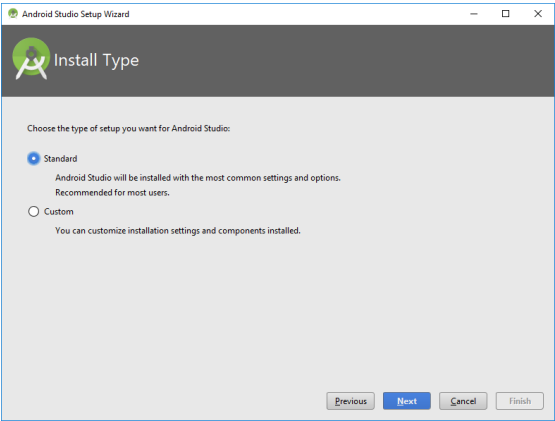
**КИМ № 11**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Раздел 1.3</b> <b>Разработка</b> <b>мобильных</b> <b>приложений</b>		<b>МДК.01.03 Разработка мобильных приложений</b>
<b>Тема 1.3.1.</b> Основные платформы и языки разработки мобильных приложений		1. Основные платформы мобильных приложений, сравнительная характеристика 2. Нативные приложения, веб-приложения, гибридные и кроссплатформенные приложения, их области применения 3. Основные языки для разработки мобильных приложений (Java, Objective-C и др.) 4. Инструменты разработки мобильных приложений (JDK/ AndroidStudio/ WebView/ Phonegap и др.)
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с.

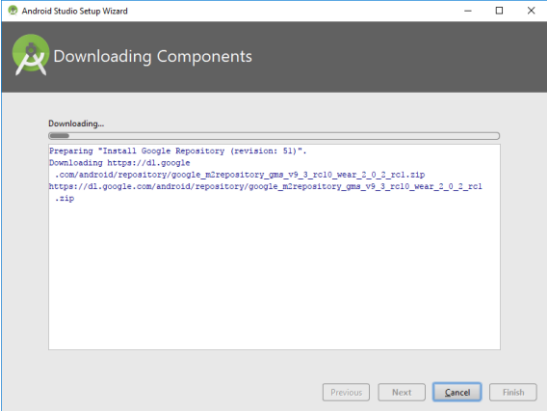
	<p>Электронные издания (электронные ресурсы)</p> <p>1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a></p> <p>Дополнительные источники:</p> <p>1. Подбельский В. Язык C#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342</p>
<b>Вариант</b>	<p>1. Настроить аппаратное обеспечение для отладки приложений на устройстве.</p> <p>2. Настроить операционную систему для отладки на реальном устройстве.</p>
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям</p> <p>1. Включить на устройстве Android возможность выполнения отладки приложений через подключение к порту USB. Чтобы включить возможность отладки через подключение к порту USB, находясь на домашнем экране устройства Android, нужно нажать кнопку Menu, после чего запустить приложение Настройки (Settings), выбрать в списке элемент Приложения (Applications), затем – элемент Разработка (Development), и установить флажок Отладка по USB (USB Debugging). Чтобы включить возможность отладки через подключение к USB на планшете Archos 5 Internet Tablet, необходимо выбрать элемент списка Device Storage &amp; USB connection (Память устройства и подключение по USB), затем – элемент списка USB Connection Mode (Режим подключения по USB) и, наконец, элемент списка Debug Bridge (ADB) (Отладочный мост (ADB)).</p> <p>2. Установить USB-драйверы Android. Установить соответствующее SDK (JDK). Установить Android Studio:</p>   



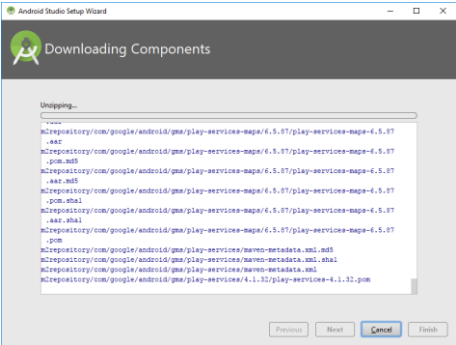
Выбрать Standart:

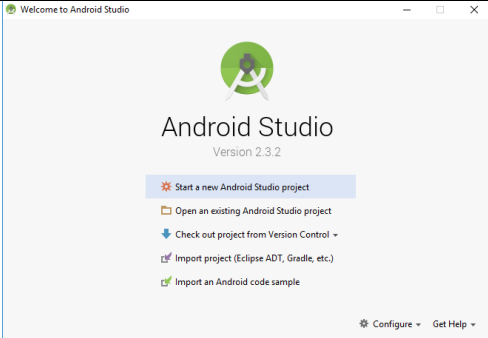


Процесс загрузки:



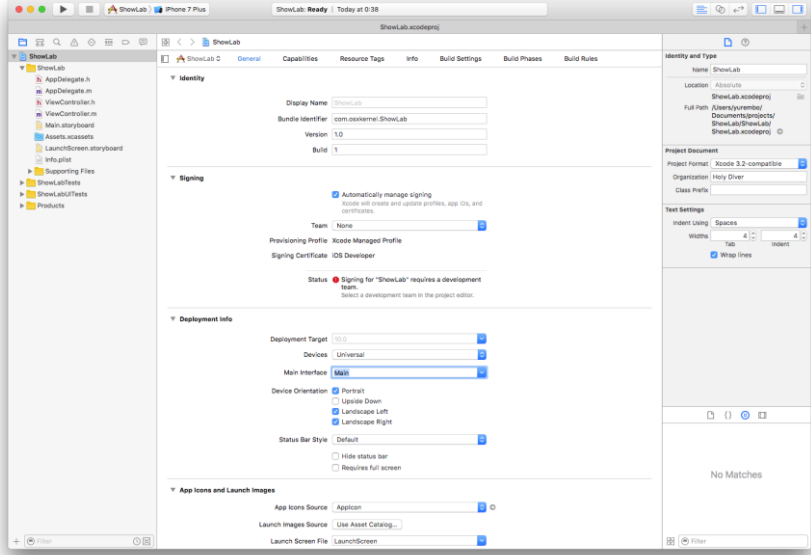
Процесс распаковки:



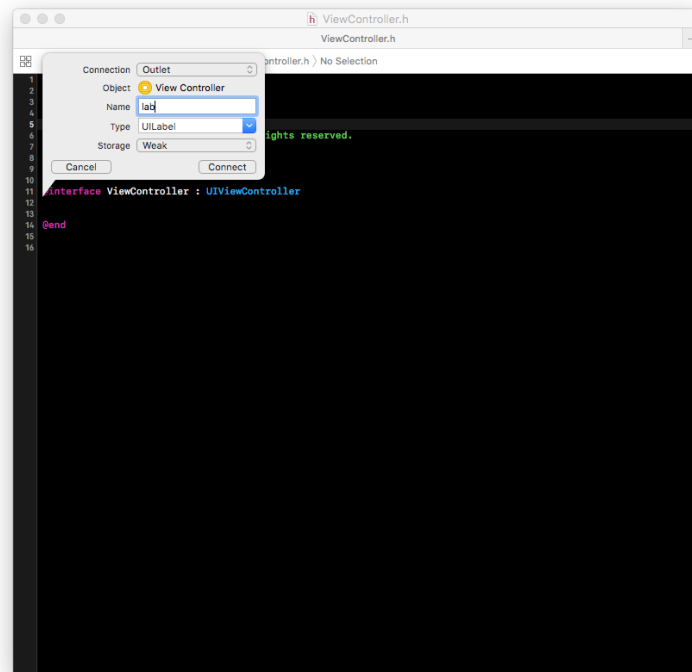
		
Критерии оценки	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 12**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<b>Тема 1.3.2</b> Создание и тестирование модулей для мобильных приложений		1. Инструментарий среды разработки мобильных приложений 2. Структура типичного мобильного приложения 3. Элементы управления и контейнеры 4. Работа со списками 5. Способы хранения данных
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с. Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a> Дополнительные источники: 1. Подбельский В. Язык С#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика,

	2013. – 408 с. - ISBN: 9785279035342
<b>Вариант</b>	<p>1. Разработать простое мобильное приложение для мобильной операционной системы iOS. Построить пользовательский интерфейс с помощью Interface Builder. Связать графические элементы с кодом приложения. Создать обработчики событий. Протестировать готовое приложение на симуляторе iPhone 7.</p> <p>2. Создать анимацию View-компонентов: изменение прозрачности, изменение размера, перемещение, поворот.</p>
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям</p> <p>1. В качестве заготовки для приложения выбрать Single View Application. Нажать Next. На следующей странице мастера ввести имя проекта, например, ShowLab. Далее в ниспадающем списке Devices выбрать Universal. Снять флажки Include Unit Tests и Include UI Tests. Нажать Next. Выбрать папку для сохранения проекта. Нажать кнопку Create. В результате откроется окно со списком всех примененных к проекту параметров. В этом окне можно изменить установленные ранее в мастере параметры: ориентацию, целевое устройство и прочее.</p>  <p>Создание интерфейса приложения. Для этого одним кликом выбрать файл Main.storyboard в списке слева (если этот файл не видно, развернуть содержимое папки ShowLab). Правее списка все окно займет Interface Builder. В центре отобразится макет устройства. В правом нижнем углу окна находится панель компонентов. Перетащить оттуда на макет компоненты Label и Button. Выше панели компонентов находится список свойств. Если он отсутствует, нажать кнопку Show the Attributes Inspector, находящуюся под заголовком окна в правой части интерфейса. Выделить в макете компонент Label и настроить его свойство Text: в ниспадающем списке оставить выбор Plain, в строку ниже ввести нужную надпись, например, «Hello, World». Если текст не помещается в границы надписи, изменить их, перетаскивая маркеры на краях компонента. Чтобы централизовать его по горизонтали, нужно перейти на страницу Size Inspector, нажав на кнопку Show the Size Inspector (справа от Show the Attributes Inspector). На этой странице из ниспадающего списка Arrange выбрать пункт Center Horizontally in Container. Теперь выбрать компонент Button, изменить его свойство Text на желаемую метку — Switch. Отцентрировать так же, как описано выше.</p> <p>Создать связь между графическими элементами и кодом следующим</p>

образом. В Visual Studio (или Delphi) объект в коде создается автоматически в тот момент, когда визуальный компонент помещается на форму. В Xcode этого не происходит. Нужно открыть содержимое заголовочного файла ViewController.h в отдельном окне, дважды щелкнув на нем. В этом файле находится объявление расширения класса UIViewController, помечается ключевым словом @interface. Затем переместить курсор мыши на компонент — текстовую метку, нажать клавишу Ctrl и левую кнопку мыши. Переместить курсор в окно с кодом (файл ViewController.h), за курсором потянется синяя линия. Отпустить мышь и клавишу внутри описания интерфейса ViewController. Появится окно создания Outlet'a.



Это свойство объекта, которое ссылается на другой объект (в данном случае визуальный компонент). Нужно ввести имя объекта Outlet, по нему будет необходимо обращаться к визуальному компоненту, например, lab. Далее выбрать тип объекта, он выбран правильно: UILabel. Еще ниже в списке Storage выбрать тип ссылки на объект: weak или strong. Если выбрать strong, то объект, на который указывает свойство, будет существовать до тех пор, пока свойство указывает на него, в таком случае он не сможет автоматически удалиться, когда перестанет использоваться. С другой стороны, когда действует слабая ссылка (weak), объект может самоуничтожиться. Итак, выбрать тип ссылки weak и нажать кнопку Connect. В итоге в код добавится следующая строка:

```
@property (weak, nonatomic) IBOutlet UILabel *lab;
```

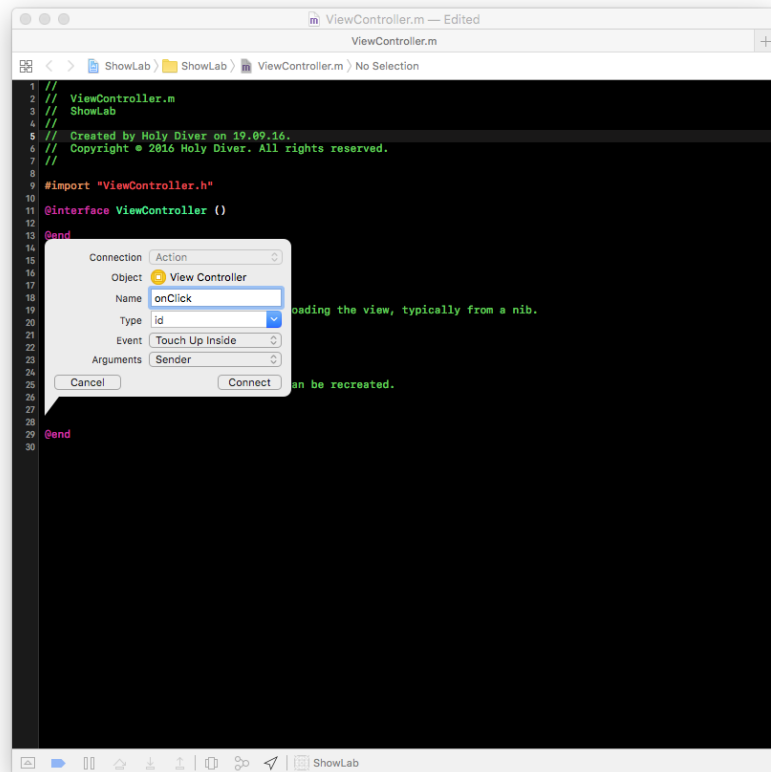
Создать Outlet для кнопки. Алгоритм остался прежним. Только для свойства Name ввести другое имя, например, but. В код будет добавлена строка:

```
@property (weak, nonatomic) IBOutlet UIButton *but;
```

В результате получатся 2 указателя на визуальные компоненты: lab и but — соответственно, надпись и кнопку. Теперь, используя указатели, можно манипулировать компонентами в коде.

Затем надо создать обработчик события нажатия кнопки. Для этого в отдельном окне открыть файл реализации ViewController.m. Точно таким же образом, как перетаскивалась линия в заголовочный файл для создания аутлета, от кнопки нужно перетащить линию в файл

реализации и отпустить до закрывающей командной скобки — @end. Появится окно для создания события, подобное окну создания аутлета.



Заполнить поле Name: его значение представляет имя свойства — метода. Например, onClick. Значение поля Type оставить по умолчанию — id. В списке Arguments оставить значение по умолчанию: Sender — это объект, отправивший данный сигнал, т.е. кнопка. Нажать кнопку Connect. В итоге будет добавлен следующий код:

```
-(IBAction)onClick:(id)sender {
}
```

Между командными скобками написать выполняемый при нажатии кнопки код:

```
_lab.hidden = !_lab.hidden;
```

Скомпилировать приложение и запустить на симуляторе для iPhone 7.

## 2. Создать проект:

Project name: P0201\_SimpleAnimation

Build Target: Android 2.3.3

Application name: SimpleAnimation

Package name: ru.startandroid.develop.p0201simpleanimation

Create Activity: MainActivity

В папке res создать папку anim: правой кнопкой на res и в меню выбрать New -> Folder. В папке anim создать: правой кнопкой на anim и в меню выбрать New -> File. В этих файлах нужно конфигурировать анимацию.

Создать следующие файлы в папке res/anim:

Имя файла: myalpha.xml

Содержимое:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<alpha
```

```
xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    android:fromAlpha="0.0"
    android:toAlpha="1.0"
    android:duration="3000">
</alpha>
```

Описание трансформации: меняется прозрачность с 0 до 1 в течение трех секунд.

Имя файла: myscale.xml

Содержимое:

```
<?xml version="1.0" encoding="utf-8"?>
<scale
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromXScale="0.1"
    android:toXScale="1.0"
    android:fromYScale="0.1"
    android:toYScale="1.0"
    android:pivotX="50%"
    android:pivotY="50%"
    android:duration="3000">
</scale>
```

Описание трансформации: изменение размера с 0.1 от оригинальной ширины и высоты до 1. Точка, относительно которой будет производиться масштабирование, лежит ровно посередине объекта (pivotX, pivotY). Продолжительность – 3 сек.

Имя файла: mytrans.xml

Содержимое:

```
<?xml version="1.0" encoding="utf-8"?>
<translate
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromXDelta="-150"
    android:toXDelta="0"
    android:fromYDelta="-200"
    android:toYDelta="0"
    android:duration="3000">
</translate>
```

Описание трансформации: перемещение с -150 относительно текущей позиции по оси X и -200 по оси Y в текущую позицию (0,0). Продолжительность – 3 сек.

Имя файла: myrotate.xml

Содержимое:

```
<?xml version="1.0" encoding="utf-8"?>
<rotate
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromDegrees="0"
    android:toDegrees="360"
    android:duration="3000">
</rotate>
```

Описание трансформации: поворот относительно левого верхнего угла (т.к. не указаны pivotX, pivotY) на 360 градусов в течение трех секунд

Имя файла: mycombo.xml

Содержимое:

```
<?xml version="1.0" encoding="utf-8"?>
<set
  xmlns:android="http://schemas.android.com/apk/res/android">
  <rotate
    android:fromDegrees="0"
    android:toDegrees="360"
    android:duration="3000"
    android:pivotX="50%"
    android:pivotY="50%">
  </rotate>
  <scale
    android:fromXScale="0.1"
    android:toXScale="1.0"
    android:fromYScale="0.1"
    android:toYScale="1.0"
    android:pivotX="50%"
    android:pivotY="50%"
    android:duration="3000">
  </scale>
</set>
```

Описание трансформации: одновременно увеличение размера и вращение в течение трех секунд. Для комбинации трансформ использовать тег <set>.

Открыть main.xml и создать экран:

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:id="@+id/frameLayout1"
  android:layout_height="match_parent">
  <TextView
    android:text="TextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_vertical|center_horizontal"
    android:id="@+id/tv"
    android:textSize="38sp">
  </TextView>
</FrameLayout>
```

По центру экрана находится TextView, к нему и будут применяться трансформации. Для этого создать контекстное меню для TextView, добавить пункты меню, соответствующие наборам.

```
package ru.startandroid.develop.p0201simpleanimation;
```

```
import android.app.Activity;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.ContextMenu.ContextMenuInfo;
import android.view.MenuItem;
import android.view.View;
import android.view.animation.Animation;
```



```

import android.view.animation.AnimationUtils;
import android.widget.TextView;

public class MainActivity extends Activity {

    // константы для ID пунктов меню
    final int MENU_ALPHA_ID = 1;
    final int MENU_SCALE_ID = 2;
    final int MENU_TRANSLATE_ID = 3;
    final int MENU_ROTATE_ID = 4;
    final int MENU_COMBO_ID = 5;

    TextView tv;

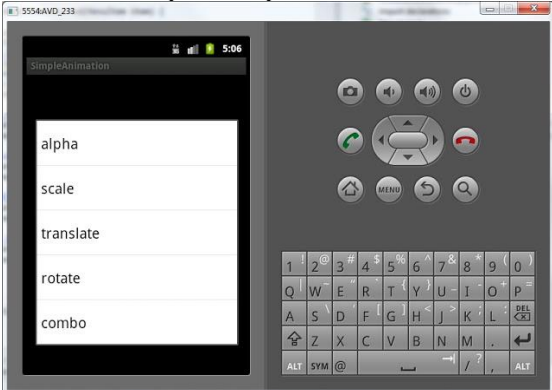
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        tv = (TextView) findViewById(R.id.tv);
        // регистрируем контекстное меню для компонента tv
        registerForContextMenu(tv);
    }

    @Override
    public void onCreateContextMenu(ContextMenu menu, View v,
        ContextMenuInfo menuInfo) {
        // TODO Auto-generated method stub
        switch (v.getId()) {
            case R.id.tv:
                // добавляем пункты
                menu.add(0, MENU_ALPHA_ID, 0, "alpha");
                menu.add(0, MENU_SCALE_ID, 0, "scale");
                menu.add(0, MENU_TRANSLATE_ID, 0, "translate");
                menu.add(0, MENU_ROTATE_ID, 0, "rotate");
                menu.add(0, MENU_COMBO_ID, 0, "combo");
                break;
        }
        super.onCreateContextMenu(menu, v, menuInfo);
    }

    @Override
    public boolean onContextItemSelected(MenuItem item) {
        Animation anim = null;
        // определяем какой пункт был нажат
        switch (item.getItemId()) {
            case MENU_ALPHA_ID:
                // создаем объект анимации из файла anim/myalpha
                anim = AnimationUtils.loadAnimation(this, R.anim.myalpha);
                break;
            case MENU_SCALE_ID:
                anim = AnimationUtils.loadAnimation(this, R.anim.myscale);
                break;
            case MENU_TRANSLATE_ID:
                anim = AnimationUtils.loadAnimation(this, R.anim.mytrans);
                break;
        }
    }
}

```

	<pre> case MENU_ROTATE_ID:     anim = AnimationUtils.loadAnimation(this, R.anim.myrotate);     break; case MENU_COMBO_ID:     anim = AnimationUtils.loadAnimation(this, R.anim.mycombo);     break; } // запускаем анимацию для компонента tv tv.startAnimation(anim); return super.onContextItemSelected(item); } } </pre> <p>Все сохранить и запустить приложение. Вызвать контекстное меню для TextView, и протестировать анимации:</p> 	
Критерии оценки	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

**КИМ № 13**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ КОМПЬЮТЕРНОГО ТЕСТИРОВАНИЯ**

<i>Раздел модуля 4. Системное программирование</i>	<i>МДК.01.04 Системное программирование</i>
<i>Тема 1.4.1. Программирование на языке низкого уровня</i>	<ol style="list-style-type: none"> <li>1. <b>Подсистемы управления ресурсами.</b></li> <li>2. <b>Управление процессами.</b> Определение процесса. Создание процессов. Завершение процессов.</li> <li>3. <b>Управление потоками.</b> Определение потока. Контекст потока. Состояния потока.</li> <li>4. <b>Параллельная обработка потоков.</b> Диспетчеризация и планирование потоков. Приостановка и возобновление потоков. Динамическое изменение приоритетов потоков.</li> <li>5. <b>Создание процессов и потоков.</b> Создание потоков. Завершение потоков. Обслуживание потоков. Синхронизация потоков и процессов.</li> <li>6. <b>Обмен данными между процессами. Передача сообщений.</b> Наследование дескрипторов. Дублирование дескрипторов. Псевдодескрипторы процессов.</li> <li>7. <b>Анонимные и именованные каналы.</b> Работа с анонимными каналами в Windows. Анонимные каналы. Создание анонимных каналов. Создание клиентов с анонимным каналом. Обмен данными по анонимному каналу. Перенаправление стандартного ввода-вывода. Именованные каналы. Создание именованных каналов. Соединение сервера с клиентом. Соединение клиентов с именованным каналом. Обмен данными по именованному каналу. Копирование данных из именованного канала. Передача транзакций по именованному каналу. Определение и изменение состояния именованного канала. Получение информации об именованном канале.</li> <li>8. <b>Сетевое программирование сокетов.</b> Понятие сокета. Программирование сетевых задач. Межпроцессное взаимодействие. Сетевое программирование с сокетами и каналами. Идентификация машины. Серверы и клиенты. Тестирование программ без сети. Дейтаграммы. Чтение файла с сервера.</li> <li>9. <b>Динамически подключаемые библиотеки DLL.</b> Концепция механизма отображения файлов в память. Создание и открытие объекта, отображающего файл. Обмен данными между процессами через отображаемый в память файл. Сброс вида в файл. Концепция динамически подключаемых библиотек. Создание DLL. Динамическая загрузка и отключение DLL. Использование DLL. Использование файла определений. Статическая загрузка DLL. Динамическая локальная память потока. Распределение и освобождение локальной памяти потока. Запись и чтение из локальной памяти потока. Статическая локальная память потока.</li> <li>10. <b>Сервисы.</b> Концепция сервиса. Структура сервиса. Организация функции main. Организация функции ServiceMain. Организация обработчика управляющих команд. Открытия доступа к базе данных сервисов. Установка сервиса. Открытие доступа к сервису. Запуск сервиса. Определение и изменение состояния сервиса. Определение и изменение конфигурации сервиса. Определение имени сервиса. Управление сервисом. Блокирование базы данных сервисов.</li> <li>11. <b>Виртуальная память. Выделение памяти процессам.</b> Концепция виртуальной памяти. Организация виртуальной памяти. Алгоритмы замещения страниц. Рабочее множество процесса. Организация виртуальной памяти в Windows. Состояния виртуальной памяти процесса. Резервирование, распределение и освобождение виртуальной</li> </ol>

		<p>памяти. Блокирование виртуальных страниц в реальной памяти. Изменение атрибутов доступа к виртуальной странице. Работа с кучей в Windows.</p> <p><b>12. Работа с буфером экрана.</b> Буфер экрана. Создание и активация буфера экрана. Определение и установка параметров буфера экрана. Функции для работы с курсором. Чтение и установка атрибутов консоли. Прокрутка буфера экрана.</p>
<b>Форма контроля</b>		<i>компьютерное тестирование</i>
<b>Вид контроля</b>		Индивидуальная работа Выполнить тест по теме.
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Компьютерное тестирование выполняется в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Запустить тест №1
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК
<b>Источники</b>		<p><i>Основные источники:</i></p> <ol style="list-style-type: none"> <li>Мезенцев К.Н. Автоматизированные информационные системы, 2013 г.</li> <li>Селищев Н. 1с: Бухгалтерия 8.2 для бухгалтера, 2014 г.</li> <li>Алексеев А., Дерут О. 1С: Предприятие 8. Описание встроенного языка. Москва, фирма 1С. – 2012 г. 350 с.</li> </ol> <p><i>Дополнительные источники:</i></p> <ol style="list-style-type: none"> <li>Учебник по 1С [Электронный ресурс] // Электронные данные. – Режим доступа: <a href="http://www.mista.ru/tutor_1c/">http://www.mista.ru/tutor_1c/</a></li> </ol>
<b>Вариант</b>		<ol style="list-style-type: none"> <li>Выберите из предложенного списка, что может являться критерием эффективности вычислительной системы: <ol style="list-style-type: none"> <li>пропускная способность;</li> <li>занятость оперативной памяти;</li> <li>загруженность центрального процессора;</li> </ol> </li> </ol>

	<p>2. Системы пакетной обработки предназначены для решения задач:</p> <ol style="list-style-type: none"> <li>1. вычислительного характера</li> <li>2. требующих постоянного диалога с пользователем</li> <li>3. требующих решения конкретной задачи за определенный промежуток времени</li> </ol> <p>3. В каких системах гарантируется выполнение задания за определенный промежуток времени:</p> <ol style="list-style-type: none"> <li>1. пакетной обработки</li> <li>2. разделения времени</li> <li>3. системах реального времени</li> </ol> <p>4. В системах пакетной обработки суммарное время выполнения смеси задач:</p> <ol style="list-style-type: none"> <li>1. равно сумме времен выполнения всех задач смеси</li> <li>2. меньше или равно суммы времен выполнения всех задач смеси</li> <li>3. больше или равно суммы времен выполнения всех задач смеси</li> </ol> <p>5. В системах реального времени</p> <ol style="list-style-type: none"> <li>1. набор задач неизвестен заранее</li> <li>2. набор задач известен заранее</li> <li>3. известен или нет набор задач зависит от характера системы</li> </ol> <p>6. Самое неэффективное использование ресурсов вычислительной системы:</p> <ol style="list-style-type: none"> <li>1. в системах пакетной обработки</li> <li>2. в системах разделения времени</li> <li>3. в системах реального времени</li> </ol> <p>7. В многопоточных системах поток есть –</p> <ol style="list-style-type: none"> <li>1. заявка на ресурсы</li> <li>2. заявка на ресурс ЦП</li> <li>3. заявка на ресурс ОП</li> </ol> <p>8. Потоки создаются с целью:</p> <ol style="list-style-type: none"> <li>1. ускорения работы процесса</li> <li>2. защиты областей памяти</li> <li>3. улучшения межпроцессного взаимодействия</li> </ol> <p>9. Как с точки зрения экономии ресурсов лучше распараллелить работу:</p> <ol style="list-style-type: none"> <li>1. создать несколько процессов</li> <li>2. создать несколько потоков</li> <li>3. случаи а) и б) равнозначны, можно выбирать любой из них</li> </ol> <p>10. Планирование потоков игнорирует:</p> <ol style="list-style-type: none"> <li>1. приоритет потока</li> <li>2. время ожидания в очереди</li> <li>3. принадлежность некоторому процессу</li> </ol> <p>11. В каких системах тип планирования статический</p> <ol style="list-style-type: none"> <li>1. реального времени</li> <li>2. разделения времени</li> <li>3. пакетной обработки</li> </ol> <p>12. Состояние, которое не определено для потока в системе:</p> <ol style="list-style-type: none"> <li>1. выполнение</li> <li>2. синхронизация</li> <li>3. ожидание</li> <li>4. готовность</li> </ol> <p>13. Каких смен состояний не существует в системе:</p> <ol style="list-style-type: none"> <li>1. выполнение → готовность</li> <li>2. ожидание → выполнение</li> <li>3. ожидание → готовность</li> <li>4. готовность → ожидание</li> </ol> <p>14. Какой из алгоритмов планирования является централизованным:</p> <ol style="list-style-type: none"> <li>1. вытесняющий</li> <li>2. невытесняющий</li> </ol> <p>15. При каком кванте времени в системах, использующих алгоритм квантования, время ожидания потока в очереди не зависит от длительности ее выполнения:</p> <ol style="list-style-type: none"> <li>1. при маленьком кванте времени</li> <li>2. при длительном кванте времени</li> <li>3. при любом кванте времени</li> </ol> <p>16. Приоритет процесса не зависит от:</p> <ol style="list-style-type: none"> <li>1. того, является ли процесс системным или прикладным</li> </ol>
--	---

	<ul style="list-style-type: none"> <li>2. статуса пользователя</li> <li>3. требуемых процессом ресурсов</li> </ul> <p>17. В каких пределах может изменяться приоритет потока в системе Windows NT:</p> <ul style="list-style-type: none"> <li>1. от базового приоритета процесса до нижней границы диапазона приоритета потоков реального времени</li> <li>2. от нуля до базового приоритета процесса</li> <li>3. базовый приоритет процесса <math>\pm 2</math></li> </ul> <p>18. Каких классов прерываний нет?</p> <ul style="list-style-type: none"> <li>1. аппаратных</li> <li>2. асинхронных</li> <li>3. внутренних</li> <li>4. программных</li> </ul> <p>19. Какие из прерываний можно считать синхронными?</p> <ul style="list-style-type: none"> <li>1. внешние</li> <li>2. внутренние</li> <li>3. программные</li> <li>4. динамические</li> </ul> <p>20. Память с самой высокой стоимостью единицы хранения:</p> <ul style="list-style-type: none"> <li>1. дисковая память</li> <li>2. оперативная память</li> <li>3. регистры процессора</li> </ul> <p>21. Какая функция ОС по управления оперативной памятью характерна только для мультизадачных ОС:</p> <ul style="list-style-type: none"> <li>1. выделение памяти по запросу</li> <li>2. освобождение памяти по завершению процесса</li> <li>3. защита памяти</li> </ul> <p>22. Какая стратегия управления памятью определяет, какие конкретно данные необходимо загружать в память:</p> <ul style="list-style-type: none"> <li>1. выборки</li> <li>2. размещения</li> <li>3. замещения</li> <li>4. загрузки</li> </ul> <p>23. Виртуальные адреса являются результатом работы:</p> <ul style="list-style-type: none"> <li>1. пользователя</li> <li>2. транслятора</li> <li>3. компоновщика</li> <li>4. ассемблера</li> </ul> <p>24. Какого типа адреса могут быть одинаковыми в разных процессах:</p> <ul style="list-style-type: none"> <li>1. виртуальные</li> <li>2. физические</li> <li>3. реальные</li> <li>4. сегментные</li> </ul> <p>25. Недостатки распределения памяти фиксированными разделами:</p> <ul style="list-style-type: none"> <li>1. сложность реализации</li> <li>2. сложность защиты</li> <li>3. ограничение на число одновременно выполняющихся процессов</li> <li>4. фрагментация памяти</li> </ul> <p>26. Какой процесс обязательно должен выполняться в системе памяти с перемещаемыми разделами:</p> <ul style="list-style-type: none"> <li>1. сжатие</li> <li>2. перемещение</li> <li>3. свопинг</li> </ul> <p>27. Что из ниже перечисленного верно для свопинга:</p> <ul style="list-style-type: none"> <li>1. на диск выгружается неиспользуемая в настоящий момент часть процесса</li> <li>2. на диск выгружаются неиспользуемые процессом данные</li> <li>3. на диск выгружается не активный процесс</li> </ul> <p>28. Таблица страниц используется для:</p> <ul style="list-style-type: none"> <li>1. преобразования виртуального адреса в физический</li> <li>2. для ускорения работы процесса</li> <li>3. для реализации свопинга</li> </ul> <p>29. Объем страницы:</p> <ul style="list-style-type: none"> <li>1. выбирается по возможности максимальный</li> </ul>
--	---

	<p>2. выбирается минимальным</p> <p>3. для процессоров x86 стандартно равен 4 кбайта</p> <p>30. Кэширование – это:</p> <ol style="list-style-type: none"> <li>1. способ функционирования дисковых устройств</li> <li>2. способ работы с ОП</li> <li>3. способ взаимного функционирования двух типов запоминающих устройств</li> </ol> <p>31. Что может выступать в качестве кэша для ОП:</p> <ol style="list-style-type: none"> <li>1. дисковые устройства</li> <li>2. быстродействующая статическая память</li> <li>3. виртуальная память</li> </ol> <p>32. Атаки класса «отказ в обслуживании» направлены на:</p> <ol style="list-style-type: none"> <li>1. полный или частичный вывод ОС из строя</li> <li>2. вывод из строя аппаратуры ПК</li> <li>3. полное или частичное удаление установленного ПО</li> </ol> <p>33. Какой вид многозадачности не существует?</p> <ol style="list-style-type: none"> <li>1. Вытесняющая многозадачность.</li> <li>2. Кооперативная (не вытесняющая) многозадачность.</li> <li>3. Симметричная многозадачность.</li> </ol> <p>34. Существуют ли классификация ядер ОС по особенностям выполнения ядра в многопроцессорных системах? (учитывая, что такие системы ядром поддерживаются)</p> <ol style="list-style-type: none"> <li>1. Да</li> <li>2. Нет</li> </ol> <p>35. Где должен располагаться код для обнаружения оборудования? (учитывая современные устройства)</p> <ol style="list-style-type: none"> <li>1. В ядре (или обязательных модулях, серверах для немонолитных архитектур).</li> <li>2. Вне ядра, в драйверах.</li> </ol> <p>36. Какое ядро современных ОС поддерживает Multiboot Specification?</p> <ol style="list-style-type: none"> <li>1. Windows</li> <li>2. SunOS 82</li> <li>3. MacOS</li> <li>4. Linux</li> <li>5. Все ядра BSD</li> </ol> <p>37. Что означает аббревиатура PIC в контексте ОС?</p> <ol style="list-style-type: none"> <li>1. Programmable Interrupt Controller</li> <li>2. Past Implemented Code</li> <li>3. Position Independent Code</li> <li>4. Portable Incompatible Code</li> </ol> <p>38. Какие основные преимущества микроядерной архитектуры?</p> <ol style="list-style-type: none"> <li>1. Упрощение переносимости</li> <li>2. Улучшение безопасности</li> <li>3. Повышенная отказоустойчивость и степень структурированности</li> <li>4. Все выше перечисленное</li> </ol> <p>39. Предшественником какого современного семейства ОС была ОС Minix Эндрю Таненбаума?</p> <ol style="list-style-type: none"> <li>1. BSD</li> <li>2. Windows</li> <li>3. Linux</li> </ol> <p>40. Нашли ли экзоядерные ОС широкое применение в современной вычислительной технике?</p> <ol style="list-style-type: none"> <li>1. Да</li> <li>2. Нет</li> </ol> <p>41. В какой из ОС впервые был реализован стек протоколов TCP/IP?</p> <ol style="list-style-type: none"> <li>1. BSD</li> <li>2. Windows</li> <li>3. Linux</li> <li>4. DOS</li> </ol> <p>42. Выберите не подходящее утверждение об отношении DOS к первым версиям Windows?</p> <ol style="list-style-type: none"> <li>1. В Windows можно было запускать приложения DOS</li> <li>2. Многие функции Windows делегировались соответствующим функциям DOS (то есть для этого производилось переключение режимов работы ЦПУ)</li> </ol>
--	--

	<p>3. Поддержка приложений DOS была ограниченной и неполной (при эмуляции на VDM, в рамках режима V86)</p> <p>43. В какой ОС поддержка графического интерфейса пользователя (GUI) интегрирована непосредственно в ядро?</p> <ol style="list-style-type: none"> <li>1. Windows</li> <li>2. BSD</li> <li>3. Linux</li> </ol> <p>44. Укажите типы сообщений, которые могут использоваться в микроядерных ОС.</p> <ol style="list-style-type: none"> <li>1. Синхронные и асинхронные.</li> <li>2. Только синхронные.</li> <li>3. Только асинхронные.</li> </ol> <p>45. В чём главный недостаток монолитных ядер?</p> <ol style="list-style-type: none"> <li>1. Их нельзя модифицировать во время работы</li> <li>2. Со временем они настолько разрастаются, что резко усложняется внесение каких-либо изменений</li> <li>3. Они занимают слишком много оперативной памяти</li> </ol> <p>46. Укажите основное средство межпроцессного взаимодействия в микроядерных архитектурах.</p> <ol style="list-style-type: none"> <li>1. Потоки</li> <li>2. Удалённые вызовы процедур (RPC, Remote Procedure Call)</li> <li>3. Сообщения</li> </ol> <p>47. Какая нотация вызовов функций принята в системных вызовах Windows?</p> <ol style="list-style-type: none"> <li>1. Смесь нотаций языков C и Pascal (обратный порядок аргументов, очистка стека функцией)</li> <li>2. Нотация языка Pascal (прямой порядок аргументов, очистка стека функцией)</li> <li>3. Нотация языка C (обратный порядок аргументов, очистка стека вызывающим кодом)</li> </ol> <p>48. Достаточно ли установки антивирусного пакета для того, чтобы считать ОС защищенной:</p> <ol style="list-style-type: none"> <li>1. да</li> <li>2. нет</li> <li>3. зависит от конкретных условий работы</li> </ol> <p>49. Для обеспечения безопасности системы должны использоваться средства, которые при отказе переходят в состояние:</p> <ol style="list-style-type: none"> <li>1. максимальной защиты</li> <li>2. минимальной защиты</li> </ol> <p>50. При организации защиты в системе необходимо руководствоваться принципом:</p> <ol style="list-style-type: none"> <li>1. максимальной защиты</li> <li>2. минимальной защиты</li> <li>3. баланса возможного ущерба от угрозы и затрат на ее предотвращение</li> </ol> <p>51. Слабости парольной защиты:</p> <ol style="list-style-type: none"> <li>1. трудность распознавания</li> <li>2. возможность раскрытия пароля путем подбора</li> <li>3. возможность обхода парольной защиты</li> </ol> <p>52. Процесс авторизации – это процесс</p> <ol style="list-style-type: none"> <li>1. ввода пользователем учетной информации</li> <li>2. доказательства того, что пользователь тот, за кого себя выдает</li> <li>3. выполнения действий, необходимых для того, чтобы пользователь мог начать работу в системе</li> </ol> <p>53. В асимметричных системах шифрования:</p> <ol style="list-style-type: none"> <li>1. ключ шифрования совпадает с ключом расшифрования</li> <li>2. ключ шифрования отличается от ключа расшифрования</li> <li>3. ключи генерируются случайным образом</li> </ol> <p>54. Правила разграничения доступа не должны позволять:</p> <ol style="list-style-type: none"> <li>1. присутствия ничейных объектов в системе</li> <li>2. присутствия объектов, недоступных для администраторов системы</li> <li>3. присутствия всем доступных объектов</li> </ol> <p>55. Файловая система является частью:</p> <ol style="list-style-type: none"> <li>1. дисковых систем</li> <li>2. драйверов дисков</li> </ol>
--	--



	<p>3. ОС</p> <p>4. пользовательских программ</p> <p>56. Какую структуру образуют файлы в ФС (файловой системе) FAT?</p> <ol style="list-style-type: none"> <li>1. древовидную</li> <li>2. сетевую</li> <li>3. реляционную</li> <li>4. плоскую</li> </ol> <p>57. Определите, какое это имя файла: USER\DO\FEDYA.DOC:</p> <ol style="list-style-type: none"> <li>1. полное</li> <li>2. простое</li> <li>3. относительное</li> </ol> <p>58. Одна ФС в системах Windows занимает, как правило:</p> <ol style="list-style-type: none"> <li>1. 1 физический диск</li> <li>2. 1 логический диск</li> <li>3. 1 раздел диска</li> </ol> <p>59. В ФС FAT атрибуты файлов хранятся</p> <ol style="list-style-type: none"> <li>1. вместе с файлом</li> <li>2. в каталогах</li> <li>3. в индексных дескрипторах</li> <li>4. в таблицах FAT</li> </ol> <p>60. Диски – это память:</p> <ol style="list-style-type: none"> <li>1. с последовательным доступом</li> <li>2. с индексно-последовательным доступом</li> <li>3. с прямым доступом</li> </ol> <p>61. Какой разметки нет на диске?</p> <ol style="list-style-type: none"> <li>1. дорожек</li> <li>2. кластеров</li> <li>3. цилиндров</li> <li>4. секторов</li> </ol> <p>62. Минимальная единица, участвующая в операциях обмена с дисковым устройством:</p> <ol style="list-style-type: none"> <li>1. байт</li> <li>2. сектор</li> <li>3. дорожка</li> <li>4. цилиндр</li> </ol> <p>63. Размер логического диска:</p> <ol style="list-style-type: none"> <li>1. меньше или равен размеру раздела</li> <li>2. равен размеру раздела</li> <li>3. больше или равен размеру раздела</li> </ol> <p>64. ОС Windows поддерживают следующие типы разделов:</p> <ol style="list-style-type: none"> <li>1. основной</li> <li>2. базовый</li> <li>3. подкачки</li> <li>4. дополнительный</li> </ol> <p>65. Раздел, с которого загружается ОС при запуске компьютера называется:</p> <ol style="list-style-type: none"> <li>1. загрузочным</li> <li>2. основным</li> <li>3. активным</li> </ol> <p>66. Минимальный фактический размер файла на диске равен:</p> <ol style="list-style-type: none"> <li>1. 1 биту</li> <li>2. 1 байту</li> <li>3. 1 сектору</li> <li>4. 1 кластеру</li> </ol> <p>67. На диске не может быть кластера размером:</p> <ol style="list-style-type: none"> <li>1. 512 байт</li> <li>2. 1024 байта</li> <li>3. 1536 байт</li> <li>4. 2048 байт</li> </ol> <p>68. Числовое значение –12, 16, 32 – в ФС FAT отражает:</p> <ol style="list-style-type: none"> <li>1. размер кластера на диске</li> <li>2. разрядность элемента в таблице FAT</li> <li>3. допустимое количество символов в имени файла</li> </ol> <p>69. Максимальный размер диска, поддерживаемого FAT16:</p>
--	--

	<ol style="list-style-type: none"> <li>1. практически неограничен</li> <li>2. 512 Мбайт</li> <li>3. 2 Гбайта</li> </ol> <p>70. Недостатки ФС FAT:</p> <ol style="list-style-type: none"> <li>1. сложность реализации</li> <li>2. не поддерживают разграничения доступа к файлам и каталогам</li> <li>3. не поддерживают длинных имен файлов</li> <li>4. не содержат средств поддержки отказоустойчивости</li> </ol> <p>71. Какие функции выполняет операционная система?</p> <ol style="list-style-type: none"> <li>1. обеспечение организации и хранения файлов</li> <li>2. организация диалога с пользователем, управления аппаратурой и ресурсами компьютера</li> <li>3. все выше перечисленные</li> </ol> <p>72. Где находится BIOS?</p> <ol style="list-style-type: none"> <li>1. в оперативно-запоминающем устройстве (ОЗУ)</li> <li>2. на винчестере</li> <li>3. на CD-ROM</li> <li>4. в постоянно-запоминающем устройстве (ПЗУ)</li> </ol> <p>73. Папка, в которую временно попадают удалённые объекты, называется ...</p> <ol style="list-style-type: none"> <li>1. Корзина</li> <li>2. Оперативная</li> <li>3. Портфель</li> <li>4. Блокнот</li> </ol> <p>74. Текущий диск - это ...</p> <ol style="list-style-type: none"> <li>1. диск, с которым пользователь работает в данный момент времени</li> <li>2. CD-ROM</li> <li>3. жесткий диск</li> <li>4. диск, в котором хранится операционная система</li> </ol> <p>75. ОС Windows поддерживает длинные имена файлов. Длинным именем файла считается ...</p> <ol style="list-style-type: none"> <li>1. любое имя файла без ограничения на количество символов в имени файла</li> <li>2. любое имя файла латинскими буквами, не превышающее 255 символов</li> <li>3. любое имя файла, не превышающее 255 символов</li> </ol> <p>76. Внутренние команды - это ...</p> <ol style="list-style-type: none"> <li>1. команды, предназначенные для создания файлов и каталогов</li> <li>2. команды, встроенные в DOS</li> <li>3. команды, которые имеют расширения .sys, .exe, .com</li> </ol> <p>77. Загрузчик операционной системы MS DOS служит для ...</p> <ol style="list-style-type: none"> <li>1. загрузки программ в оперативную память ЭВМ</li> <li>2. обработки команд, введенных пользователем</li> <li>3. считывания в память модулей операционной системы io.sys и msdos.sys</li> <li>4. подключения устройств ввода-вывода</li> </ol> <p>78. Какие команды DOS называются внешними?</p> <ol style="list-style-type: none"> <li>1. команды, предназначенные только для работы с периферийными устройствами</li> <li>2. команды, хранящиеся на диске в виде отдельных программ и вызываемые по мере необходимости</li> <li>3. все команды, которые можно реализовать с помощью DOS</li> </ol> <p>79. BIOS - это ...</p> <ol style="list-style-type: none"> <li>1. игровая программа</li> <li>2. диалоговая оболочка</li> <li>3. базовая система ввода-вывода</li> <li>4. командный язык операционной системы</li> </ol> <p>80. Операционная система сети включает в себя управляющие и обслуживающие программы. К управляющим относятся</p> <ol style="list-style-type: none"> <li>1. Межпрограммный доступ</li> <li>2. Доступ отдельных прикладных программ к ресурсам сети</li> <li>3. Синхронизация работы прикладных программных средств</li> <li>4. Обмен информации между программами и др.</li> <li>5. Все выше перечисленные</li> </ol> <p>81. Какой вид многозадачности не существует?</p> <ol style="list-style-type: none"> <li>1. Вытесняющая многозадачность.</li> <li>2. Кооперативная (не вытесняющая) многозадачность.</li> </ol>
--	--

	<p>3. Симметричная многозадачность.</p> <p>82. Существуют ли классификация ядер ОС по особенностям выполнения ядра в многопроцессорных системах? (учитывая, что такие системы ядром поддерживаются)</p> <ol style="list-style-type: none"> <li>1. Да</li> <li>2. Нет</li> </ol> <p>83. Где должен располагаться код для обнаружения оборудования? (учитывая современные устройства)</p> <ol style="list-style-type: none"> <li>1. В ядре (или обязательных модулях, серверах для монолитных архитектур).</li> <li>2. Вне ядра, в драйверах.</li> </ol> <p>84. Какое ядро современных ОС поддерживает Multiboot Specification?</p> <ol style="list-style-type: none"> <li>1. Windows</li> <li>2. SunOS 82</li> <li>3. MacOS</li> <li>4. Linux</li> <li>5. Все ядра BSD</li> </ol> <p>85. Что означает аббревиатура PIC в контексте ОС?</p> <ol style="list-style-type: none"> <li>1. Programmable Interrupt Controller</li> <li>2. Past Implemented Code</li> <li>3. Position Independent Code</li> <li>4. Portable Incompatible Code</li> </ol> <p>86. Какие основные преимущества микроядерной архитектуры?</p> <ol style="list-style-type: none"> <li>1. Упрощение переносимости</li> <li>2. Улучшение безопасности</li> <li>3. Повышенные отказоустойчивость и степень структурированности</li> <li>4. Все выше перечисленное</li> </ol> <p>87. Предшественником какого современного семейства ОС была ОС Minix Эндрю Таненбаума?</p> <ol style="list-style-type: none"> <li>1. BSD</li> <li>2. Windows</li> <li>3. Linux</li> </ol> <p>88. Нашли ли экзоядерные ОС широкое применение в современной вычислительной технике?</p> <ol style="list-style-type: none"> <li>1. Да</li> <li>2. Нет</li> </ol> <p>89. В какой из ОС впервые был реализован стек протоколов TCP/IP?</p> <ol style="list-style-type: none"> <li>1. BSD</li> <li>2. Windows</li> <li>3. Linux</li> <li>4. DOS</li> </ol> <p>90. Выберите не подходящее утверждение об отношении DOS к первым версиям Windows?</p> <ol style="list-style-type: none"> <li>1. В Windows можно было запускать приложения DOS</li> <li>2. Многие функции Windows делегировались соответствующим функциям DOS (то есть для этого производилось переключение режимов работы ЦПУ)</li> <li>3. Поддержка приложений DOS была ограниченной и неполной (при эмуляции на VDM, в рамках режима V86)</li> </ol> <p>91. В какой ОС поддержка графического интерфейса пользователя (GUI) интегрирована непосредственно в ядро?</p> <ol style="list-style-type: none"> <li>1. Windows</li> <li>2. BSD</li> <li>3. Linux</li> </ol> <p>92. Укажите типы сообщений, которые могут использоваться в микроядерных ОС.</p> <ol style="list-style-type: none"> <li>1. Синхронные и асинхронные.</li> <li>2. Только синхронные.</li> <li>3. Только асинхронные.</li> </ol> <p>93. В чём главный недостаток монолитных ядер?</p> <ol style="list-style-type: none"> <li>1. Их нельзя модифицировать во время работы</li> <li>2. Со временем они настолько разрастаются, что резко усложняется внесение каких-либо изменений</li> <li>3. Они занимают слишком много оперативной памяти</li> </ol>
--	--

	<p>94. Укажите основное средство межпроцессного взаимодействия в микроядерных архитектурах.</p> <ol style="list-style-type: none"> <li>1. Потоки</li> <li>2. Удалённые вызовы процедур (RPC, Remote Procedure Call)</li> <li>3. Сообщения</li> </ol> <p>95. Какая нотация вызовов функций принята в системных вызовах Windows?</p> <ol style="list-style-type: none"> <li>1. Смесь нотаций языков C и Pascal (обратный порядок аргументов, очистка стека функцией)</li> <li>2. Нотация языка Pascal (прямой порядок аргументов, очистка стека функцией)</li> <li>3. Нотация языка C (обратный порядок аргументов, очистка стека вызывающим кодом)</li> </ol> <p>96. Для выполнения каких операций оптимизирована серверная операционная система Novell NetWare?</p> <ol style="list-style-type: none"> <li>1. доступ к файлам</li> <li>2. доступ к файлам и печать</li> <li>3. почтовая служба</li> </ol> <p>97. Какие из этих ОС могут использоваться для построения одноранговых сетей?</p> <ol style="list-style-type: none"> <li>1. NetWare</li> <li>2. Windows 95/98</li> <li>3. MS-DOS</li> </ol> <p>98. Какие задачи не выполняет ОС при обмене с периферийным устройством?</p> <ol style="list-style-type: none"> <li>1. решает, может ли быть выполнена требуемая операция обмена</li> <li>2. передает запрос драйверу ПУ</li> <li>3. принимает информацию из сети от устройства управления ПУ</li> </ol> <p>99. Сколько выделенных серверов может одновременно работать в сети?</p> <ol style="list-style-type: none"> <li>1. нет специальных ограничений</li> <li>2. только один</li> <li>3. по числу требуемых в сети служб — для каждой сетевой службы отдельный выделенный сервер</li> </ol> <p>100. Пусть сеть состоит из идентичных компьютеров, на которых установлены однотипные ОС. За одним из компьютеров административно закреплены функции по обслуживанию запросов остальных компьютеров (все пользователи сети хранят свои файлы на диске этого компьютера). К какому типу сети вы отнесете эту сеть?</p> <ol style="list-style-type: none"> <li>1. сеть с выделенным сервером</li> <li>2. одноранговая сеть</li> <li>3. гибридная сеть</li> </ol>				
<b>Пакет преподавателя</b>	<p>Ответы:</p> <ol style="list-style-type: none"> <li>1. 1</li> <li>2. 1</li> <li>3. 3</li> <li>4. 1</li> <li>5. 3</li> <li>6. 1</li> <li>7. 3</li> <li>8. 1</li> <li>9. 3</li> <li>10. 3</li> <li>11. 3</li> <li>12. 4</li> <li>13. 1</li> <li>14. 2</li> <li>15. 3</li> <li>16. 2</li> <li>17. 3</li> <li>18. 4</li> <li>19. 2</li> <li>20. 3</li> <li>21. 3</li> </ol>	<ol style="list-style-type: none"> <li>22. 1</li> <li>23. 2</li> <li>24. 1</li> <li>25. 3</li> <li>26. 3</li> <li>27. 3</li> <li>28. 1</li> <li>29. 3</li> <li>30. 3</li> <li>31. 1</li> <li>32. 1</li> <li>33. 2</li> <li>34. 3</li> <li>35. 1</li> <li>36. 1</li> <li>37. 4</li> <li>38. 3</li> <li>39. 3</li> <li>40. 3</li> <li>41. 2</li> </ol>	<ol style="list-style-type: none"> <li>42. 1</li> <li>43. 3</li> <li>44. 1</li> <li>45. 1</li> <li>46. 2</li> <li>47. 3</li> <li>48. 1</li> <li>49. 3</li> <li>50. 1</li> <li>51. 1</li> <li>52. 4</li> <li>53. 3</li> <li>54. 3</li> <li>55. 3</li> <li>56. 2</li> <li>57. 1</li> <li>58. 3</li> <li>59. 1</li> <li>60. 1</li> <li>61. 2</li> </ol>	<ol style="list-style-type: none"> <li>62. 3</li> <li>63. 1</li> <li>64. 1</li> <li>65. 3</li> <li>66. 4</li> <li>67. 3</li> <li>68. 2</li> <li>69. 3</li> <li>70. 2</li> <li>71. 2</li> <li>72. 4</li> <li>73. 1</li> <li>74. 1</li> <li>75. 3</li> <li>76. 2</li> <li>77. 3</li> <li>78. 2</li> <li>79. 3</li> <li>80. 5</li> <li>81. 3</li> </ol>	<ol style="list-style-type: none"> <li>82. 1</li> <li>83. 1</li> <li>84. 4</li> <li>85. 3</li> <li>86. 3</li> <li>87. 3</li> <li>88. 2</li> <li>89. 1</li> <li>90. 3</li> <li>91. 1</li> <li>92. 1</li> <li>93. 2</li> <li>94. 3</li> <li>95. 1</li> <li>96. 2</li> <li>97. 2</li> <li>98. 3</li> <li>99. 2</li> <li>100. 2</li> </ol>

Критерии оценки	Отлично	Ответ полный и правильный, показывающий прочные знания в области профессиональной деятельности. В тесте с выбором варианта может быть допущена 1 ошибка
	Хорошо	ответ полный и правильный, показывающий прочные знания в области профессиональной деятельности, при этом допущены две-три ошибки в тесте с выбором, исправленные самостоятельно по требованию преподавателя
	Удовлетворительно	ответ полный, но при этом допущены 4-5 ошибок в тесте с выбором
	Неудовлетворительно	при ответе обнаружено непонимание обучающимся основного содержания и допущены более 5 ошибок в тесте с выбором ответа

**КИМ № 14**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ КОМПЬЮТЕРНОГО ТЕСТИРОВАНИЯ**

<i>Раздел модуля 4. Системное программирование</i>	<i>МДК.01.04 Системное программирование</i>
<i>Тема 1.4.1. Программирование на языке низкого уровня</i>	<p>1. <b>Подсистемы управления ресурсами.</b></p> <p>2. <b>Управление процессами.</b> Определение процесса. Создание процессов. Завершение процессов.</p> <p>3. <b>Управление потоками.</b> Определение потока. Контекст потока. Состояния потока.</p> <p>4. <b>Параллельная обработка потоков.</b> Диспетчеризация и планирование потоков. Приостановка и возобновление потоков. Динамическое изменение приоритетов потоков.</p> <p>5. <b>Создание процессов и потоков.</b> Создание потоков. Завершение потоков. Обслуживание потоков. Синхронизация потоков и процессов.</p> <p>6. <b>Обмен данными между процессами. Передача сообщений.</b> Наследование дескрипторов. Дублирование дескрипторов. Псевдодескрипторы процессов.</p> <p>7. <b>Анонимные и именованные каналы.</b> Работа с анонимными каналами в Windows. Анонимные каналы. Создание анонимных каналов. Создание клиентов с анонимным каналом. Обмен данными по анонимному каналу. Перенаправление стандартного ввода-вывода. Именованные каналы. Создание именованных каналов. Соединение сервера с клиентом. Соединение клиентов с именованным каналом. Обмен данными по именованному каналу. Копирование данных из именованного канала. Передача транзакций по именованному каналу. Определение и изменение состояния именованного канала. Получение информации об именованном канале.</p> <p>8. <b>Сетевое программирование сокетов.</b> Понятие сокета. Программирование сетевых задач. Межпроцессное взаимодействие. Сетевое программирование с сокетами и каналами. Идентификация машины. Серверы и клиенты. Тестирование программ без сети. Дейтаграммы. Чтение фала с сервера.</p> <p>9. <b>Динамически подключаемые библиотеки DLL.</b> Концепция механизма отображения файлов в память. Создание и открытие объекта, отображающего файл. Обмен данными между процессами через отображаемый в память файл. Сброс вида в файл. Концепция динамически подключаемых библиотек. Создание DLL. Динамическая загрузка и отключение DLL. Использование DLL. Использование файла определений. Статическая загрузка DLL. Динамическая локальная память потока. Распределение и освобождение локальной памяти потока. Запись и чтение из локальной памяти потока. Статическая локальная память потока.</p> <p>10. <b>Сервисы.</b> Концепция сервиса. Структура сервиса. Организация функции main. Организация функции ServiceMain. Организация обработчика управляющих команд. Открытия доступа к базе данных сервисов. Установка сервиса. Открытие доступа к сервису. Запуск сервиса. Определение и изменение состояния сервиса. Определение и изменение конфигурации сервиса. Определение имени сервиса. Управление сервисом. Блокирование базы данных сервисов.</p> <p>11. <b>Виртуальная память. Выделение памяти процессам.</b> Концепция виртуальной памяти. Организация виртуальной памяти. Алгоритмы замещения страниц. Рабочее множество процесса. Организация виртуальной памяти в Windows. Состояния виртуальной памяти процесса. Резервирование, распределение и освобождение виртуальной памяти. Блокирование виртуальных страниц в реальной памяти.</p>

		Изменение атрибутов доступа к виртуальной странице. Работа с кучей в Windows. 12. <b>Работа с буфером экрана.</b> Буфер экрана. Создание и активация буфера экрана. Определение и установка параметров буфера экрана. Функции для работы с курсором. Чтение и установка атрибутов консоли. Прокрутка буфера экрана.
<b>Форма контроля</b>		<i>компьютерное тестирование</i>
<b>Вид контроля</b>		Индивидуальная работа Выполнить тест по теме.
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Компьютерное тестирование выполняется в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Запустить тест №1
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК
<b>Источники</b>		<i>Основные источники:</i> 1. Мезенцев К.Н. Автоматизированные информационные системы, 2013 г. 2. Селищев Н. 1с: Бухгалтерия 8.2 для бухгалтера, 2014 г. 3. Алексеев А., Дерут О. 1С: Предприятие 8. Описание встроенного языка. Москва, фирма 1С. – 2012 г. 350 с. <i>Дополнительные источники:</i> 1. Учебник по 1С [Электронный ресурс] // Электронные данные. – Режим доступа: <a href="http://www.mista.ru/tutor_1c/">http://www.mista.ru/tutor_1c/</a>
<b>Вариант</b>		1 Виртуальные адреса являются результатом работы: а) пользователя; б) транслятора; в) компоновщика; г) ассемблера.

	<p>2 Какого типа адреса могут быть одинаковыми в разных процессах:</p> <ul style="list-style-type: none"> <li>а) виртуальные;</li> <li>б) физические;</li> <li>в) реальные;</li> <li>г) сегментные.</li> </ul> <p>3 Недостатки распределения памяти фиксированными разделами:</p> <ul style="list-style-type: none"> <li>а) сложность реализации;</li> <li>б) сложность защиты;</li> <li>в) ограничение на число одновременно выполняющихся процессов;</li> <li>г) фрагментация памяти.</li> </ul> <p>4 Все файлы и каталоги в системе NTFS однозначно идентифицируются:</p> <ul style="list-style-type: none"> <li>а) именем;</li> <li>б) индексным дескриптором;</li> <li>в) номером записи в MFT;</li> <li>г) системным идентификатором.</li> </ul> <p>5 Состояния, которые не определены для потока в системе:</p> <ul style="list-style-type: none"> <li>а) выполнение;</li> <li>б) синхронизация;</li> <li>в) ожидание;</li> <li>г) готовность.</li> </ul> <p>6 Смены состояний в системе:</p> <ul style="list-style-type: none"> <li>а) выполнение → готовность;</li> <li>б) ожидание → выполнение;</li> <li>в) ожидание → готовность;</li> <li>г) готовность → ожидание.</li> </ul> <p>7 Принципы подсистемы планирования потоков в ОС Windows NT:</p> <ul style="list-style-type: none"> <li>а) квантование;</li> <li>б) относительные приоритеты;</li> <li>в) абсолютные приоритеты;</li> <li>г) вытеснение.</li> </ul> <p>8 Моменты перепланировки использования ЦП могут быть связаны с событиями:</p> <ul style="list-style-type: none"> <li>а) прерывания от таймера в связи с истечением кванта времени;</li> <li>б) завершение операции ввода/вывода;</li> <li>в) окончание выполнения цикла в программе;</li> <li>г) обнаружение деления на ноль в программе.</li> </ul> <p>9 Способы, которыми шины выполняют прерывания:</p> <ul style="list-style-type: none"> <li>а) векторный;</li> <li>б) скалярный;</li> <li>в) опрашиваемый;</li> <li>г) вызываемый.</li> </ul> <p>10 Синхронными прерываниями можно считать:</p> <ul style="list-style-type: none"> <li>а) внешние;</li> <li>б) внутренние;</li> <li>в) программные;</li> <li>г) динамические</li> </ul>
<b>Пакет преподавателя</b>	<p>Ответы:</p> <p>13. б</p> <p>14. а</p>



	15. в 16. б 17. г 18. б 19. а 20. в 21. г 22. б	
Критерии оценки	Отлично	Ответ полный и правильный, показывающий прочные знания в области профессиональной деятельности. В тесте с выбором варианта может быть допущена 1 ошибка
	Хорошо	ответ полный и правильный, показывающий прочные знания в области профессиональной деятельности, при этом допущены две-три ошибки в тесте с выбором, исправленные самостоятельно по требованию преподавателя
	Удовлетворительно	ответ полный, но при этом допущены 4-5 ошибок в тесте с выбором
	Неудовлетворительно	при ответе обнаружено непонимание обучающимся основного содержания и допущены более 5 ошибок в тесте с выбором ответа

**КИМ № 15**  
**КОМПЛЕКТ ЗАДАНИЙ ДЛЯ ПРАКТИЧЕСКОГО ЗАДАНИЯ**

<i>Раздел модуля 4. Системное программирование</i>	<i>МДК.01.04 Системное программирование</i>
<i>Тема 1.4.1. Программирование на языке низкого уровня</i>	<p><b>1. Подсистемы управления ресурсами.</b></p> <p><b>2. Управление процессами.</b> Определение процесса. Создание процессов. Завершение процессов.</p> <p><b>3. Управление потоками.</b> Определение потока. Контекст потока. Состояния потока.</p> <p><b>4. Параллельная обработка потоков.</b> Диспетчеризация и планирование потоков. Приостановка и возобновление потоков. Динамическое изменение приоритетов потоков.</p> <p><b>5. Создание процессов и потоков.</b> Создание потоков. Завершение потоков. Обслуживание потоков. Синхронизация потоков и процессов.</p> <p><b>6. Обмен данными между процессами. Передача сообщений.</b> Наследование дескрипторов. Дублирование дескрипторов. Псевдодескрипторы процессов.</p> <p><b>7. Анонимные и именованные каналы.</b> Работа с анонимными каналами в Windows. Анонимные каналы. Создание анонимных каналов. Создание клиентов с анонимным каналом. Обмен данными по анонимному каналу. Перенаправление стандартного ввода-вывода. Именованные каналы. Создание именованных каналов. Соединение сервера с клиентом. Соединение клиентов с именованным каналом. Обмен данными по именованному каналу. Копирование данных из именованного канала. Передача транзакций по именованному каналу. Определение и изменение состояния именованного канала. Получение информации об именованном канале.</p> <p><b>8. Сетевое программирование сокетов.</b> Понятие сокета. Программирование сетевых задач. Межпроцессное взаимодействие. Сетевое программирование с сокетами и каналами. Идентификация машины. Серверы и клиенты. Тестирование программ без сети. Дейтаграммы. Чтение файла с сервера.</p> <p><b>9. Динамически подключаемые библиотеки DLL.</b> Концепция механизма отображения файлов в память. Создание и открытие объекта, отображающего файл. Обмен данными между процессами через отображаемый в память файл. Сброс вида в файл. Концепция динамически подключаемых библиотек. Создание DLL. Динамическая загрузка и отключение DLL. Использование DLL. Использование файла определений. Статическая загрузка DLL. Динамическая локальная память потока. Распределение и освобождение локальной памяти потока. Запись и чтение из локальной памяти потока. Статическая локальная память потока.</p> <p><b>10. Сервисы.</b> Концепция сервиса. Структура сервиса. Организация функции main. Организация функции ServiceMain. Организация обработчика управляющих команд. Открытия доступа к базе данных сервисов. Установка сервиса. Открытие доступа к сервису. Запуск сервиса. Определение и изменение состояния сервиса. Определение и изменение конфигурации сервиса. Определение имени сервиса. Управление сервисом. Блокирование базы данных сервисов.</p> <p><b>11. Виртуальная память. Выделение памяти процессам.</b> Концепция виртуальной памяти. Организация виртуальной памяти. Алгоритмы замещения страниц. Рабочее множество процесса. Организация виртуальной памяти в Windows. Состояния виртуальной памяти процесса. Резервирование, распределение и освобождение виртуальной памяти. Блокирование виртуальных страниц в реальной памяти.</p>

		Изменение атрибутов доступа к виртуальной странице. Работа с кучей в Windows. <b>12. Работа с буфером экрана.</b> Буфер экрана. Создание и активация буфера экрана. Определение и установка параметров буфера экрана. Функции для работы с курсором. Чтение и установка атрибутов консоли. Прокрутка буфера экрана.
<b>Форма контроля</b>		Выполнение практического задания
<b>Вид контроля</b>		Индивидуальная работа
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия выполнения задания</b>		Практическая работа проводится в аудитории, время проведения работы 1 час 30 минут
<b>Инструкция для студентов</b>		Получить задание и выполнить практическую работу,
<b>Оборудование и оснащение</b>		Для проведения работы применяется следующее оснащение: – оборудование: – ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с. Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a> Дополнительные источники: 1. Подбельский В. Язык С#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика, 2013. – 408 с. - ISBN: 9785279035342
<b>Вариант</b>		1. Написать Wait-функцию, которая приостанавливает планирование некоторого потока до момента освобождения одного или нескольких синхронизирующих объектов ядра. 2. Написать функцию WaitForSingleObject, которая используется для

	<p>ожидания завершения потока.</p> <p>3. Написать функцию, в которой предполагается, что поток ожидает завершения одного из трех других потоков.</p> <p>4. Открыть именованный канал для выполнения асинхронных операций.</p> <p>5. Использовать проекцию файла для инверсии содержимого файла.</p>
<b>Пакет преподавателя</b>	<p>Проверяется правильность выполнения задания, согласно критериям</p> <p>1. Функция WaitForSingleObject блокирует процесс до освобождения одного синхронизирующего объекта ядра.</p> <pre> DWORD WaitForSingleObject( HANDLE hHandle,      // описатель синхронизирующего объекта ядра DWORD dwMilliseconds // максимальное время ожидания ); 2. int g_x_count = 0; CRITICAL_SECTION cs; HANDLE hCountThread = 0; DWORD WINAPI CountThread(PVOID) { EnterCriticalSection( &amp; cs ); g_x_count++; LeaveCriticalSection( &amp; cs ); return 0; } void main() { InitializeCriticalSection( &amp; cs ); hCountThread = CreateThread( 0, 0, CountThread, 0, 0, 0); if ( hCountThread == 0 ) { // поток не создан DeleteCriticalSection( &amp; cs ); return; } DWORD dwWaitResult = WaitForSingleObject( hCountThread, 1000 ); switch ( dwWaitResult ) { case WAIT_OBJECT_0: // поток завершен MessageBox( 0, "Success.", "Wait", MB_OK ); break; case WAIT_TIMEOUT: // истекло заданное время ожидания MessageBox( 0, "Timeout.", "Wait", MB_OK ); break; case WAIT_FAILED: //неправильный вызов функции MessageBox( 0, "Failure.", "Wait", MB_OK ); break; } DeleteCriticalSection( &amp; cs ); } 3. HANDLE handles[3]; handles[0] = hThread_1; handles[1] = hThread_2; handles[2] = hThread_3; DWORD dwWaitResult = WaitForMultipleObjects(3,handles,FALSE,5000); switch ( dwWaitResult ) { case WAIT_FAILED: ... break; case WAIT_TIMEOUT: ... </pre>

```

break;
case WAIT_OBJECT_0 + 0:
// завершен поток
Thread_1
break;
case WAIT_OBJECT_0 + 1:
// завершен поток
Thread_2
break;
case WAIT_OBJECT_0 + 2:
// завершен поток
Thread_3
break;
}
4.
OVERLAPPED os;
LPCTSTR lpszPipeName = "\\\\.\\pipe\\pipe_one";
HANDLE hPipe = CreateNamedPipe(
lpszPipeName,
FILE_FLAG_OVERLAPPED | PIPE_ACCESS_DUPLEX,
PIPE_TYPE_MESSAGE | PIPE_READMODE_MESSAGE | PIPE_WAIT,
1,
0,
0,
1000,
NULL);
memset( & os, 0, sizeof( OVERLAPPED ) );
os.hEvent = CreateEvent( NULL, TRUE, FALSE, NULL );
ResetEvent( os.hEvent );
ConnectNamedPipe( hPipe, & os );
if ( GetLastError() == ERROR_IO_PENDING ) {
DWORD dwWaitResult = WaitForSingleObject( os.hEvent, INFINITE );
if ( dwWaitResult == WAIT_OBJECT_0 ) {
// подключение к каналу
} else {
// ошибка или тайм - аут
}
}
5.
#include <windows.h>
void main() {
HANDLE hFile = 0, hFileMapping = 0; DWORD dwSize = 0;
PBYTE pbFile = 0; BYTE bByte = 0; TCHAR * pstrFile;
// Открываем файл
hFile = CreateFile(
"C:\\test.txt",
GENERIC_READ | GENERIC_WRITE,
0,
0,
OPEN_EXISTING,
FILE_ATTRIBUTE_NORMAL,
0);
// Размер файла
dwSize = GetFileSize(hFile, 0);
// Создаем проекцию
hFileMapping = CreateFileMapping(
hFile,
0,
PAGE_READWRITE,
0,

```

	<pre> dwSize, 0); // Проецируем файл на память pstrFile = (TCHAR*)MapViewOfFile( hFileMapping, FILE_MAP_WRITE, 0, 0, dwSize); // Выполняем операции с памятью, которые записываются в файл pstrFile = _strev(pstrFile); // Отключаем проекцию от памяти UnmapViewOfFile(pbFile); // Закрываем объекты ядра CloseHandle(hFileMapping); CloseHandle(hFile); } </pre>	
Критерии оценки	Отлично	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу.
	Хорошо	Задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче или образцу, но при этом допущены несущественные неточности, устраненные без помощи преподавателя.
	Удовлетворительно	Задание выполнено не в полном объеме или не полностью соответствует поставленной задаче или образцу, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя.
	Неудовлетворительно	Задание не выполнено и полностью не соответствует поставленной задаче или образцу, допущены существенные неточности, которые обучающийся не может устранить.

### **3. Комплект КИМ для промежуточной аттестации**

Промежуточная аттестация проводится в форме экзамена

*Промежуточная аттестация в форме экзамена проводится в дни, освобожденные от других форм учебной нагрузки, по отдельному расписанию за счет времени, отведенного учебным планом на промежуточную аттестацию.*

*Экзамен – это форма промежуточного контроля, целью которой является оценка теоретических знаний и практических навыков, способности студента к мышлению, приобретение навыков самостоятельной работы, умение синтезировать полученные знания и применять их при решении практических. При проведении промежуточной аттестации в форме экзамена уровень освоения оценивается оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».*

*При проведении промежуточной аттестации используются следующие КИМ:*

*- экзаменационные билеты.*

*Экзаменационные билеты оформляются по установленному образцу и хранятся в папке соответствующей образовательной программы в кабинете предметно-цикловой комиссии.*

**ПЕРЕЧЕНЬ ВОПРОСОВ К ДИФФЕРЕНЦИРОВАННОМУ ЗАЧЕТУ ПО  
МДК 01.01. РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ**

<b>Форма контроля</b>		Дифференцированный зачет
<b>Вид контроля</b>		промежуточная аттестация
<b>Объекты оценки:</b> ответы на вопросы к дифференцированному зачету		
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b> <i>(Указываются коды общих компетенций и коды их структурных элементов (дескрипторов, умений, знаний), которые проверяются данным КИМом)</i>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия проведения</b>		Аудитория, ПК. Время подготовки студента к ответу 1 час 30 минут
<b>Инструкция для студентов</b>		1. За 1 час 30 минут подготовить ответы на теоретические вопросы. 2. Защитить ответ преподавателю.
<b>Оборудование и оснащение</b>		Учебная аудитория, ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с.  Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a>  Дополнительные источники: 1. Подбельский В. Язык С#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика,



	2013. – 408 с. - ISBN: 9785279035342	
<b>Перечень экзаменационных вопросов (заданий зачета)</b>		
<b>Критерии оценки</b>	Отлично	ставится обучающемуся, проявившему всесторонние и глубокие знания учебного материала, освоившему основную и дополнительную литературу, обнаружившему творческие способности в понимании, изложении и практическом использовании усвоенных знаний. Оценка «отлично» соответствует высокому уровню освоения дисциплины (или МДК).
	Хорошо	ставится обучающемуся, проявившему полное знание учебного материала, освоившему основную рекомендованную литературу, обнаружившему стабильный характер знаний и умений и способному к их самостоятельному применению, и обновлению в ходе последующего обучения и практической деятельности. Оценка «хорошо» соответствует достаточному уровню освоения дисциплины (или МДК).
	Удовлетворительно	ставится обучающемуся, проявившему знания основного учебного материала в объеме, необходимом для последующего обучения и предстоящей практической деятельности, знакомому с основной рекомендованной литературой, допустившему неточности при ответе, но в основном обладающему необходимыми знаниями и умениями для их устранения при корректировке со стороны преподавателя. Оценка «удовлетворительно» соответствует достаточному уровню освоения дисциплины (или МДК).
	Неудовлетворительно	ставится обучающемуся, обнаружившему существенные пробелы в знании основного учебного материала, допустившему принципиальные ошибки при применении теоретических знаний, которые не позволяют ему продолжить обучение или приступить к практической деятельности без дополнительной подготовки по данной дисциплине (или МДК). Оценка «неудовлетворительно»

		соответствует низкому уровню освоения дисциплины (или МДК).
--	--	---

## **Приложение 1**

*(Образец перечня вопросов и практических заданий для дифференцированного зачёта)*

Государственное бюджетное профессиональное образовательное учреждение  
«Южно-Уральский государственный колледж»

Рассмотрено на заседании предметно-цикловой  
комиссии Информационных технологий

УТВЕРЖДАЮ:  
Зам. директора по учебной работе Калиновская Т.С.

\_\_\_\_\_  
Председатель ПЦК  
\_\_\_\_\_/Назарова Н.А. /

«\_\_\_\_\_» \_\_\_\_\_ 20\_\_\_\_ г.

\_\_\_\_\_  
Протокол № \_\_\_\_ от \_\_\_\_\_ 20\_\_\_\_  
г.

### **Вопросы (задания) к дифференцированному зачёту**

По учебной дисциплине МДК 01.01. Разработка программных модулей

\_\_\_\_\_  
Специальность 09.02.07 Информационные системы и программирование  
20\_\_\_\_ - 20\_\_\_\_ учебный год  
Преподаватель (преподаватели)  
\_\_\_\_\_

**Перечень вопросов и практических задач**

*(прикладывается перечень вопросов и практических задач в сквозном порядке)*

1. Назовите элементы интерфейса программы
2. Сформулируйте технологию ввода кода программы
3. Перечислите этапы алгоритма сохранения и запуска проекта
4. Сформулируйте назначение вкладок «Свойства», «События»
5. Назовите основные свойства компонентов «LABEL», «BUTTON»
6. Назначение целочисленных типов данных
7. Назначение вещественного типа данных
8. Назначение денежного типа данных
9. Назначение вариантного типа данных
10. Назначение символьного типа данных
11. Назначение интервального типа данных
12. Назначение перечисляемого типа данных
13. Основные стандартные математические функции
14. Основные свойства компоненты «EDIT»
15. Формат записи составного оператора
16. Формат записи условного оператора
17. Форма записи оператора варианта
18. Свойства компоненты «TListBox»
19. Назначение компоненты TComboBox.
20. Режимы работы компоненты «Поле со списком».
21. Назначение компоненты TCheckBox.
22. Назначение стандартных панелей сообщений.
23. Назначение компоненты TScrollBar и ее свойства
24. Назначение компоненты TPanel и ее свойства

**ПЕРЕЧЕНЬ ВОПРОСОВ К ЭКЗАМЕНУ ПО  
МДК 01.02. ПОДДЕРЖКА И ТЕСТИРОВАНИЕ ПРОГРАММНЫХ МОДУЛЕЙ**

<b>Форма контроля</b>		экзамен
<b>Вид контроля</b>		промежуточная аттестация
<b>Объекты оценки:</b> ответы на вопросы к экзамену		
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b> <i>(Указываются коды общих компетенций и коды их структурных элементов (дескрипторов, умений, знаний), которые проверяются данным КИМом)</i>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия проведения</b>		Аудитория, ПК. Время подготовки студента к ответу 1 час 30 минут
<b>Инструкция для студентов</b>		1. За 1 час 30 минут подготовить ответы на теоретические вопросы. 2. Защитить ответ преподавателю.
<b>Оборудование и оснащение</b>		Учебная аудитория, ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с.  Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a>  Дополнительные источники: 1. Подбельский В. Язык С#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика,

	2013. – 408 с. - ISBN: 9785279035342	
<b>Перечень экзаменационных вопросов (заданий зачета)</b>		
<b>Критерии оценки</b>	Отлично	ставится обучающемуся, проявившему всесторонние и глубокие знания учебного материала, освоившему основную и дополнительную литературу, обнаружившему творческие способности в понимании, изложении и практическом использовании усвоенных знаний. Оценка «отлично» соответствует высокому уровню освоения дисциплины (или МДК).
	Хорошо	ставится обучающемуся, проявившему полное знание учебного материала, освоившему основную рекомендованную литературу, обнаружившему стабильный характер знаний и умений и способному к их самостоятельному применению, и обновлению в ходе последующего обучения и практической деятельности. Оценка «хорошо» соответствует достаточному уровню освоения дисциплины (или МДК).
	Удовлетворительно	ставится обучающемуся, проявившему знания основного учебного материала в объеме, необходимом для последующего обучения и предстоящей практической деятельности, знакомому с основной рекомендованной литературой, допустившему неточности при ответе, но в основном обладающему необходимыми знаниями и умениями для их устранения при корректировке со стороны преподавателя. Оценка «удовлетворительно» соответствует достаточному уровню освоения дисциплины (или МДК).
	Неудовлетворительно	ставится обучающемуся, обнаружившему существенные пробелы в знании основного учебного материала, допустившему принципиальные ошибки при применении теоретических знаний, которые не позволяют ему продолжить обучение или приступить к практической деятельности без дополнительной подготовки по данной дисциплине (или МДК). Оценка «неудовлетворительно»

		соответствует низкому уровню освоения дисциплины (или МДК).
--	--	---

## **Приложение 1**

*(Образец перечня вопросов и практических заданий для дифференцированного зачёта)*

Государственное бюджетное профессиональное образовательное учреждение  
«Южно-Уральский государственный колледж»

Рассмотрено на заседании предметно-цикловой  
комиссии Информационных технологий

УТВЕРЖДАЮ:  
Зам. директора по учебной работе Калиновская Т.С.

\_\_\_\_\_  
Председатель ПЦК  
\_\_\_\_\_/Назарова Н.А. /

«\_\_\_\_» \_\_\_\_\_ 20 \_\_\_\_ г.

Протокол № \_\_\_\_ от \_\_\_\_\_ 20 \_\_\_\_  
г.

### **Вопросы (задания) к экзамену**

По учебной дисциплине МДК 01.02. Поддержка и тестирование программных  
модулей\_\_\_\_\_

Специальность 09.02.07 Информационные системы и программирование

20\_\_\_\_ - 20\_\_\_\_ учебный год

Преподаватель (преподаватели)

\_\_\_\_\_



**Перечень вопросов и практических задач**

*(прикладывается перечень вопросов и практических задач в сквозном порядке)*

1. Перечислите основные аспекты качества программного обеспечения.  
Приведите примеры характеристик, относящихся к каждому из аспектов.
2. Расскажите о методах контроля качества программного обеспечения.  
Объясните разницу между валидацией и верификацией.
3. Дайте определение термина «тестирование». Объясните разницу между тестированием и отладкой. Опишите общую схему тестирования.
4. Дайте определение термина «тест». Расскажите о двух стратегиях тестирования.
5. Перечислите уровни тестирования программного обеспечения. Опишите, что проверяется на этих уровнях на примере.
6. Опишите на примере проектирование тестовых наборов данных с помощью метода разбиения на классы эквивалентности.
7. Опишите на примере проектирование тестовых наборов данных с помощью метода граничных значений.
8. Опишите на примере проектирование тестовых наборов данных методом покрытия операторов.
9. Опишите на примере проектирование тестовых наборов данных методом покрытия условий.
10. Опишите на примере проектирование тестовых наборов данных методом комбинаторного покрытия условий.
11. Опишите, в чем состоит функциональное тестирование.
12. Опишите, что проверяется при тестировании взаимодействия.
13. Объясните, что такое нагрузочное и стрессовое тестирование, укажите разницу между ними.
14. Перечислите основные принципы тестирования.
15. Объясните, что такое регрессионное и повторное тестирование, укажите разницу между ними.
16. Поясните разницу между ручным и автоматизированным тестированием.
17. Опишите жизненный цикл программного дефекта.
18. Дайте определение термина «надежность программного обеспечения».  
Перечислите подхарактеристики надежности.
19. Охарактеризуйте понятия: ошибка, дефект, отказ. Поясните разницу между ними на примере какой-либо программы.
20. Расскажите об оценке экономической эффективности программного продукта. Приведите пример.
21. Перечислите основные виды ошибок программных средств.
22. Перечислите методы отладки и приведите примеры.
23. Опишите, что такое модульное тестирование.
24. Опишите, что такое интеграционное тестирование.

25. Назовите цели документирования программных средств.
26. Перечислите и опишите классы документов программных средств.

**ПЕРЕЧЕНЬ ВОПРОСОВ К ДИФФЕРЕНЦИРОВАННОМУ ЗАЧЕТУ ПО  
МДК 01.03. РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ**

<b>Форма контроля</b>		Дифференцированный зачет
<b>Вид контроля</b>		промежуточная аттестация
<b>Объекты оценки:</b> ответы на вопросы к дифференцированному зачету		
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b> <i>(Указываются коды общих компетенций и коды их структурных элементов (дескрипторов, умений, знаний), которые проверяются данным КИМом)</i>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия проведения</b>		Аудитория, ПК. Время подготовки студента к ответу 1 час 30 минут
<b>Инструкция для студентов</b>		1. За 1 час 30 минут подготовить ответы на теоретические вопросы. 2. Защитить ответ преподавателю.
<b>Оборудование и оснащение</b>		Учебная аудитория, ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с.  Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a>  Дополнительные источники: 1. Подбельский В. Язык C#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика,

	2013. – 408 с. - ISBN: 9785279035342	
<b>Перечень экзаменационных вопросов (заданий зачета)</b>		
<b>Критерии оценки</b>	Отлично	ставится обучающемуся, проявившему всесторонние и глубокие знания учебного материала, освоившему основную и дополнительную литературу, обнаружившему творческие способности в понимании, изложении и практическом использовании усвоенных знаний. Оценка «отлично» соответствует высокому уровню освоения дисциплины (или МДК).
	Хорошо	ставится обучающемуся, проявившему полное знание учебного материала, освоившему основную рекомендованную литературу, обнаружившему стабильный характер знаний и умений и способному к их самостоятельному применению, и обновлению в ходе последующего обучения и практической деятельности. Оценка «хорошо» соответствует достаточному уровню освоения дисциплины (или МДК).
	Удовлетворительно	ставится обучающемуся, проявившему знания основного учебного материала в объеме, необходимом для последующего обучения и предстоящей практической деятельности, знакомому с основной рекомендованной литературой, допустившему неточности при ответе, но в основном обладающему необходимыми знаниями и умениями для их устранения при корректировке со стороны преподавателя. Оценка «удовлетворительно» соответствует достаточному уровню освоения дисциплины (или МДК).
	Неудовлетворительно	ставится обучающемуся, обнаружившему существенные пробелы в знании основного учебного материала, допустившему принципиальные ошибки при применении теоретических знаний, которые не позволяют ему продолжить обучение или приступить к практической деятельности без дополнительной подготовки по данной дисциплине (или МДК). Оценка «неудовлетворительно»

		соответствует низкому уровню освоения дисциплины (или МДК).
--	--	---

## **Приложение 1**

*(Образец перечня вопросов и практических заданий для дифференцированного зачёта)*

Государственное бюджетное профессиональное образовательное учреждение  
«Южно-Уральский государственный колледж»

Рассмотрено на заседании предметно-цикловой  
комиссии Информационных технологий

УТВЕРЖДАЮ:  
Зам. директора по учебной работе Калиновская Т.С.

\_\_\_\_\_  
Председатель ПЦК  
\_\_\_\_\_/Назарова Н.А. /

«\_\_\_\_\_» \_\_\_\_\_ 20 \_\_\_\_ г.

Протокол № \_\_\_\_ от \_\_\_\_\_ 20 \_\_\_\_  
г.

### **Вопросы (задания) к дифференцированному зачету**

По учебной дисциплине МДК 01.03. Разработка мобильных приложений

\_\_\_\_\_  
Специальность 09.02.07 Информационные системы и программирование  
20 \_\_\_\_ - 20 \_\_\_\_ учебный год  
Преподаватель (преподаватели)  
\_\_\_\_\_

**Перечень вопросов и практических задач**

*(прикладывается перечень вопросов и практических задач в сквозном порядке)*

**Теоретические вопросы**

1. Выявление функциональных требований. 2. Особенности разработки макетов интерфейсов для мобильных приложений. Концепции Human Interface Guidelines и Material Design. 3. Основные инструменты макетирования. 4. Инструменты прототипирования мобильных приложений. Понятие интерактивного прототипа. 5. Определение целей и задач разработки. Целевая аудитория проекта. Определение рамок проекта. 6. Функциональные и нефункциональные характеристики проекта. Корректное описание требований к продукту. 7. Способы схематичного отображения вариантов и сценариев использования мобильных приложений. UML-диаграммы сценариев использования. 8. Понятие базовой и альтернативной последовательности действий. 9. Инструменты тестирования и аналитики мобильных приложений. 10. Особенности жизненного цикла разработки мобильных приложений. 11. Гибкие методологии разработки программного обеспечения. Методология Agile. Специфика применения гибких методологий в разработке мобильных приложений. Понятия: Канбан-доска, Спринт и Backlog. 12. Способы описания архитектуры разработанного программного обеспечения в текстовой документации и презентации. UML-диаграммы классов. 13. Особенности описания процесса разработки программного обеспечения в текстовой документации и презентации. 14. Способы описания функциональных возможностей мобильных приложений в текстовой документации и презентации.

**Перечень практических заданий:**

- 1) Создание макетов интерфейсов приложения по выявленным функциональным характеристикам. Требования: а) Соответствие макетов интерфейсов концепции Human Interface Guidelines для iOS и Material Design для Android. б) Использование одного или нескольких основных инструментов макетирования. 2) Создание интерактивного прототипа по разработанным макетам.
- 2) 1) Определение целей и задач разработки. Целевая аудитория проекта. Определение рамок проекта. 2) Функциональные и нефункциональные характеристики проекта. 3) Способы схематичного отображения вариантов и сценариев использования мобильных приложений. UML-диаграммы сценариев использования. 4) Понятие базовой и альтернативной последовательности действий. Формат описания шаблонов экранов и контента. Формат описания API сервера. 5) Инструменты тестирования и

аналитики мобильных приложений.

- 3) Создание дизайна интерфейсов приложения по выявленным функциональным характеристикам. 2) Создание интерактивного прототипа по разработанным макетам.
- 4) Разработка программного продукта, соответствующего выявленным функциональным и нефункциональным требованиям. Нефункциональные требования, соблюдение которых является обязательным для всех программных решений: 1) Поддержка смартфонов и планшетов (необходимые поддерживаемые устройства, версии операционных систем и разрешения экранов должны быть определены самостоятельно в зависимости от специфики конкретного продукта); 2) Соответствие дизайна мобильного программного решения концепции Human Interface Guidelines для iOS; 3) Соответствие дизайна мобильного программного решения концепции Material Design для Android; 4) Использование анимации для элементов интерфейса; 5) Использование инструмента Fabric для распространения разработанного программного продукта пользователям для тестирования.
- 5) Вопросы: 1) Постановка проблемы. Выявление целей и задач проекта. 2) Описание архитектуры разработанного программного обеспечения. UML-диаграммы классов. 3) Описание процесса разработки программного обеспечения. 4) Описание функциональных возможностей разработанного мобильного приложения. 5) Описание результата выполненной работы и планов на дальнейшее развитие продукта.
- 6) Разработка презентации по разработанному программному продукту. Требования: 1) Постановка проблемы. Обозначение целей и задач проекта. 2) Описание архитектуры разработанного программного обеспечения. 3) Описание процесса разработки программного обеспечения. 4) Описание функциональных возможностей разработанного мобильного приложения. 5) Описание результата выполненной работы и планов на дальнейшее развитие продукта.



**ПЕРЕЧЕНЬ ВОПРОСОВ К ДИФФЕРЕНЦИРОВАННОМУ ЗАЧЕТУ ПО  
МДК 01.04. СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ**

<b>Форма контроля</b>		Дифференцированный зачет
<b>Вид контроля</b>		промежуточная аттестация
<b>Объекты оценки:</b> ответы на вопросы к дифференцированному зачету		
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b> <i>(Указываются коды общих компетенций и коды их структурных элементов (дескрипторов, умений, знаний), которые проверяются данным КИМом)</i>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия проведения</b>		Аудитория, ПК. Время подготовки студента к ответу 1 час 30 минут
<b>Инструкция для студентов</b>		1. За 1 час 30 минут подготовить ответы на теоретические вопросы. <b>3.</b> Защитить ответ преподавателю.
<b>Оборудование и оснащение</b>		Учебная аудитория, ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н. Федорова. – М.: Академия, 2016. – 336 с.  Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a>  Дополнительные источники: 1. Подбельский В. Язык С#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика,

	2013. – 408 с. - ISBN: 9785279035342	
<b>Перечень экзаменационных вопросов (заданий зачета)</b>		
<b>Критерии оценки</b>	Отлично	ставится обучающемуся, проявившему всесторонние и глубокие знания учебного материала, освоившему основную и дополнительную литературу, обнаружившему творческие способности в понимании, изложении и практическом использовании усвоенных знаний. Оценка «отлично» соответствует высокому уровню освоения дисциплины (или МДК).
	Хорошо	ставится обучающемуся, проявившему полное знание учебного материала, освоившему основную рекомендованную литературу, обнаружившему стабильный характер знаний и умений и способному к их самостоятельному применению, и обновлению в ходе последующего обучения и практической деятельности. Оценка «хорошо» соответствует достаточному уровню освоения дисциплины (или МДК).
	Удовлетворительно	ставится обучающемуся, проявившему знания основного учебного материала в объеме, необходимом для последующего обучения и предстоящей практической деятельности, знакомому с основной рекомендованной литературой, допустившему неточности при ответе, но в основном обладающему необходимыми знаниями и умениями для их устранения при корректировке со стороны преподавателя. Оценка «удовлетворительно» соответствует достаточному уровню освоения дисциплины (или МДК).
	Неудовлетворительно	ставится обучающемуся, обнаружившему существенные пробелы в знании основного учебного материала, допустившему принципиальные ошибки при применении теоретических знаний, которые не позволяют ему продолжить обучение или приступить к практической деятельности без дополнительной подготовки по данной дисциплине (или МДК). Оценка «неудовлетворительно»

		соответствует низкому уровню освоения дисциплины (или МДК).
--	--	---

## **Приложение 1**

*(Образец перечня вопросов и практических заданий для дифференцированного зачёта)*

Государственное бюджетное профессиональное образовательное учреждение  
«Южно-Уральский государственный колледж»

Рассмотрено на заседании предметно-цикловой  
комиссии Информационных технологий

УТВЕРЖДАЮ:  
Зам. директора по учебной работе Калиновская Т.С.

\_\_\_\_\_  
Председатель ПЦК  
\_\_\_\_\_/Назарова Н.А. /

«\_\_\_\_\_» \_\_\_\_\_ 20\_\_\_\_ г.

Протокол № \_\_\_\_ от \_\_\_\_\_ 20\_\_\_\_  
г.

### **Вопросы (задания) к экзамену**

По учебной дисциплине МДК 01.04. Системное программирование

\_\_\_\_\_  
Специальность 09.02.07 Информационные системы и программирование  
20\_\_\_\_ - 20\_\_\_\_ учебный год  
Преподаватель (преподаватели)  
\_\_\_\_\_

**Перечень вопросов и практических задач**

*(прикладывается перечень вопросов и практических задач в сквозном порядке)*

1. Логические величины, операции, выражения.
2. Печать элементов списка
3. Дан массив A из n целых чисел. Найти сумму максимального и минимального элемента в массиве. (Поиск максимума и минимума реализовать с помощью подпрограмм-функций).
4. Подпрограмма – процедура.
5. Стеки. Объявление стека.
6. Дан файл целых чисел. Выбрать наибольшее из чисел, принадлежащее интервалу [a,b]. Концы интервала a и b вводятся с клавиатуры.
7. Подпрограмма- функция.
8. Инициализация стека. Добавление элемента в стек.
9. Дан текстовый файл F1. Переписать его содержимое в файл F2, сохраняя строчную структуру и удаляя пустые строки.
10. Рекурсия.
11. Проверка стека на пустоту. Извлечение элемента из стека.
12. Дан текстовый файл F1. Переписать его содержимое в файл F2, сохраняя строчную структуру и удаляя пустые строки.
13. Основные понятия структурного программирования.
14. Очереди. Объявление очереди.
15. Даны две символьные строки S1 и S2, содержащие только строчные латинские буквы. Построить строку S3, в которую войдут только общие символы S1 и S2 в алфавитном порядке и без повторений
16. Модуль. Структура модуля.
17. Создание и заполнение внешнего файла.
18. Даны две символьные строки S1 и S2, содержащие только строчные латинские буквы. Построить строку S3, в которую войдут только общие символы S1 и S2 в алфавитном порядке и без повторений.
19. Модуль. Структура модуля.
20. Чтение данных из внешнего файла.
21. Дан файл целых чисел. Определить, сколько раз в нем повторяется максимальное значение.
22. Списки. Объявление списка.
23. Текстовые файлы.
24. Дан файл целых чисел. Определить, сколько раз в нем повторяется максимальное значение.
25. Добавление элемента в начало списка
26. Чтение данных из внешнего файла.
27. По координатам вершин треугольника вычислить его периметр, используя подпрограмму вычисления длины отрезка, соединяющего две точки. (длина отрезка=  $\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$ ), где (x1,y1)- координаты одной точки, (x2,y2)-координаты второй точки отрезка).

28. Подпрограмма – процедура.
29. Создание и заполнение внешнего файла.
30. По координатам вершин треугольника вычислить его периметр, используя подпрограмму вычисления длины отрезка, соединяющего две точки. (длина отрезка =  $\text{sgrt}(\text{sgr}(x_2 - x_1) + \text{sgr}(y_2 - y_1))$ , где  $(x_1, y_1)$  – координаты одной точки,  $(x_2, y_2)$  – координаты второй точки отрезка).
31. Текстовые файлы.
32. Добавление элемента в начало списка
33. Дан текстовый файл Note.txt. Определить длину самой длинной строки этого файла.
34. Чтение данных из внешнего файла.
35. Списки. Объявление списка.
36. Дан текстовый файл Note.txt. Определить длину самой длинной строки этого файла.
37. Проверка очереди на пустоту. Извлечение элемента из очереди.
38. Создание и заполнение внешнего файла.
39. Дан текстовый файл Note.txt. Определить длину самой длинной строки этого файла.
40. Инициализация очереди. Добавление элемента в очередь.
41. Модуль. Структура модуля.
42. Дан список L, из N целых чисел. Удалить первое вхождение максимального элемента в списке.
43. Очереди. Объявление очереди.
44. Основные понятия структурного программирования.
45. Дан список L, из N целых чисел. Удалить первое вхождение максимального элемента в списке.
46. Инициализация стека. Добавление элемента в стек.
47. Логические величины, операции, выражения.
48. Дан список L, из N целых чисел. Удалить первое вхождение максимального элемента в списке.
49. Стек. Объявление стека.
50. Подпрограмма – функция.
51. Определить среднее арифметическое чисел, хранящихся в файле Note.txt.
52. Печать элементов списка
53. Подпрограмма – процедура.
54. По заданным значениям X, Y и D вычислить
55. Вычисление MIN и MAX из двух величин оформить в виде подпрограмм – функций.
56. Добавление элемента в начало списка
57. Текстовые файлы.
58. По заданным значениям X, Y и D вычислить
59. Вычисление MIN и MAX из двух величин оформить в виде подпрограмм – функций.
60. Подпрограмма – процедура.
61. Текстовые файлы.

62. Составить рекурсивную подпрограмму вычисления  $N!$

# КИМ № 20

## ПЕРЕЧЕНЬ ЭКЗАМЕНАЦИОННЫХ ВОПРОСОВ

<b>Форма контроля</b>		Экзамен
<b>Вид контроля</b>		промежуточная аттестация
<b>Объекты оценки:</b>		
<b>Спецификация ПК</b>	ПК 1.6	ПД1.6-1, ПД1.6-2 ПУ1.6-1, ПУ1.6-2 ПЗ1.6-1, ПЗ1.6-2
	ПК 1.2	ПД4.1-1, ПД4.1-2, ПД4.1-3 ПУ4.1-1, ПУ4.1-2, ПУ4.1-3 ПЗ4.1-1, ПЗ4.1-2
<b>Спецификация ОК</b> <i>(Указываются коды общих компетенций и коды их структурных элементов (дескрипторов, умений, знаний), которые проверяются данным КИМом)</i>	ОК 1	ОД.01-1, ОД.01-2, ОД.01-3, ОД.01-4, ОД.01-5, ОД.01-6, ОД.01-7, ОД.01-8 ОУ.01-1, ОУ.01-2, ОУ.01-3, ОУ.01-4, ОУ.01-5, ОУ.01-6, ОУ.01-7, ОУ.01-8 ОЗ.01-1, ОЗ.01-2, ОЗ.01-3
	ОК 2	ОД.02-1, ОД.02-2, ОД.02-3 ОУ.02-1, ОУ.02-2, ОУ.02-3 ОЗ.02-1, ОЗ.02-2, ОЗ.02-3
	ОК 3	ОД.03-1, ОД.03-2, ОД.03-3 ОУ.03-1 ОЗ.03-1, ОЗ.03-2, ОЗ.03-3
	ОК 4	ОД.04-1, ОД.04-2, ОД.05-1 ОУ.04-1, ОУ.04-2, ОУ.05-1 ОЗ.04-1, ОЗ.05-1
	ОК 5	ОД.05-1, ОД.05-2 ОУ.05-1 ОЗ.05-1, ОЗ.05-2
	ОК 9	ОД.09-1, ОД.09-2 ОЗ.09-1, ОЗ.09-2 ОУ.09-2
	ОК 10	ОД.10-1, ОД.10-2, ОД.10-3, ОД.10-4, ОД.10-5 ОУ.10-1, ОУ.10-2, ОУ.10-4, ОУ.10-5 ОЗ.10-1, ОЗ.10-2, ОЗ.10-3, ОЗ.10-4, ОЗ.10-5
<b>Условия проведения</b>		Аудитория, экзаменационные билеты. Время подготовки студента к ответу 1 час 30 минут
<b>Инструкция для студентов</b>		1. Выбрать билет (билет содержит 2 теоретический вопроса и 2 практических задания). <b>4.</b> За 1 час 30 минут подготовить ответ на билет. <b>5.</b> Защитить ответ преподавателю.
<b>Оборудование и оснащение</b>		Учебная аудитория, ПК, ПО
<b>Источники</b>		Основные источники: Печатные издания 1. Федорова Г.Н. Разработка программных модулей программного обеспечения для компьютерных систем: учебник. Среднее профессиональное образование, профессиональная подготовка / Г.Н Федорова. – М.: Академия, 2016. – 336 с.  Электронные издания (электронные ресурсы) 1. Учебники по программированию <a href="http://programm.ws/index.php">http://programm.ws/index.php</a>  Дополнительные источники: 1. Подбельский В. Язык С#. Базовый курс. Издание второе, переработанное и дополненное. Издательство: Финансы и статистика,



	2013. – 408 с. - ISBN: 9785279035342	
<b>Перечень экзаменационных вопросов (заданий зачета)</b>		
<b>Критерии оценки</b>	Отлично	ставится обучающемуся, проявившему всесторонние и глубокие знания учебного материала, освоившему основную и дополнительную литературу, обнаружившему творческие способности в понимании, изложении и практическом использовании усвоенных знаний. Оценка «отлично» соответствует высокому уровню освоения дисциплины (или МДК).
	Хорошо	ставится обучающемуся, проявившему полное знание учебного материала, освоившему основную рекомендованную литературу, обнаружившему стабильный характер знаний и умений и способному к их самостоятельному применению, и обновлению в ходе последующего обучения и практической деятельности. Оценка «хорошо» соответствует достаточному уровню освоения дисциплины (или МДК).
	Удовлетворительно	ставится обучающемуся, проявившему знания основного учебного материала в объеме, необходимом для последующего обучения и предстоящей практической деятельности, знакомому с основной рекомендованной литературой, допустившему неточности при ответе, но в основном обладающему необходимыми знаниями и умениями для их устранения при корректировке со стороны преподавателя. Оценка «удовлетворительно» соответствует достаточному уровню освоения дисциплины (или МДК).
	Неудовлетворительно	ставится обучающемуся, обнаружившему существенные пробелы в знании основного учебного материала, допустившему принципиальные ошибки при применении теоретических знаний, которые не позволяют ему продолжить обучение или приступить к практической деятельности без дополнительной подготовки по данной дисциплине (или МДК). Оценка «неудовлетворительно»

		соответствует низкому уровню освоения дисциплины (или МДК).
--	--	---

## Приложение 3

(Образец перечня экзаменационных вопросов и практических заданий)

Государственное бюджетное профессиональное образовательное учреждение  
«Южно-Уральский государственный колледж»

Рассмотрено на заседании предметно-цикловой  
комиссии

УТВЕРЖДАЮ:  
Зам. директора по учебной работе

\_\_\_\_\_  
Председатель ПЦК

\_\_\_\_\_  
« \_\_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_ г.

\_\_\_\_\_/Назарова Н.А. /

Протокол № \_\_\_\_ от \_\_\_\_\_ 20 \_\_\_\_  
г.

### Экзаменационные вопросы

По профессиональному модулю ПМ.01. РАЗРАБОТКА МОДУЛЕЙ ПРОГРАММНОГО  
ОБЕСПЕЧЕНИЯ ДЛЯ КОМПЬЮТЕРНЫХ СИСТЕМ

\_\_\_\_\_  
Специальность 09.02.07 Информационные системы и программирование

20 \_\_\_\_ - 20 \_\_\_\_ учебный год

Преподаватель (преподаватели)

\_\_\_\_\_

## **Перечень вопросов и практических задач**

*(прикладывается перечень вопросов и практических задач в сквозном порядке)*

*Теоретические вопросы:*

*Раздел 1. Разработка программных модулей.*

1. Жизненный цикл ПО. Основные этапы разработки ПО.
2. Модели жизненного цикла программного средства.
3. Формализация задачи и разработка алгоритма.
4. ЭВМ исполнитель алгоритмов.
5. Постановка задачи на разработку программного средства.
6. Составление программы на языке программирования.
7. Структура и способы описания языков программирования высокого уровня.
8. Подпрограмма – процедура.
9. Формальные и фактические параметры.
10. Локальные и глобальные переменные.
11. Подпрограмма функция.
12. Разработка программного продукта с использованием подпрограммы процедуры.
13. Модульное программирование.
14. Методы разработки программных модулей.
15. Осуществление разработки кода программного модуля на современных языках программирования.
16. Решение задач с использованием стека.
17. Реализация процедур и функций работы с бинарным деревом.
18. Разработка программного продукта с использованием модуля.
19. Оформление документации на программное средство.
20. Объектноориентированное проектирование.
21. Документирование результатов анализа и проектирования.
22. Основы языка UML (Unified Modeling Language).
23. Создание абстрактных типов данных. Диаграмма объекта.
24. Принципы объектноориентированного анализа: абстрагирование, инкапсуляция, наследование, полиморфизм, модульность, сохраняемость, параллелизм.
25. Структура программы на языке C#. Проект.
26. Компиляция программы и сборка исполняемого модуля.
27. Размещение программы и данных в памяти.
28. Структура исполняемого модуля.
29. Стандартная библиотека функций языка C#.
30. Компиляция программы и сборка исполняемого модуля.
31. Размещение программы и данных в памяти.
32. Виртуальные функции и абстрактные базовые классы.
33. Множественное наследование.
34. Контейнеры и итераторы в библиотеке STL (Standard Template Library).
35. Ассоциативные массивы.

36. Критерии оценки качества программы.
37. Использование инструментальных средств автоматизации процесса оформления документации.

## *Раздел 2. Поддержка и тестирование программных модулей.*

1. Дать определение тестированию ПО.
2. Понятие исчерпывающего тестирования.
3. Тестирование в 50 – 60х годах XXв.
4. Тестирование в 70х годах XXв.
5. Тестирование в 80х годах XXв.
6. Тестирование в 90х годах XXв.
7. Тестирование в нулевые годы XXIв.
8. Основные характеристики современного этапа развития технологий тестирования.
9. Типичные виды деятельности тестировщика.
10. Жизненный цикл тестирования.
11. Классификация тестирования по запуску кода на исполнение.
12. Классификация тестирования по доступу к коду и архитектуре приложения.
13. Классификация тестирования по степени автоматизации.
14. Классификация тестирования по уровню детализации приложения.
15. Классификация тестирования по степени важности тестируемых функций.
16. Классификация тестирования по принципам работы с приложением.
17. Понятия Чек-листа, ментальных карт, концепт-карт. Свойства чек-листа.
18. Автоматизация тестирования.
19. Определение критериев тестирования. Сравнение заданных критериев на вложенность.
20. Основы формализма описания критериев.
21. Основы мутационного подхода и его недостатки.
22. Понятия пошагового и монолитного подходов. Достоинства и недостатки подходов.
23. Стратегии тестирования. Оптимальный критерий завершения тестирования.
24. Этапы тестирования, подлежащие автоматизации.
25. Построение тестов с помощью символьного исполнения программ.
26. Оценка полноты набора тестов. Основные подходы.

## *Раздел 3. Разработка мобильных приложений.*

1. Xcode. Создание проекта. Доступные шаблоны. Особенности DEBUG и RELEASE сборок. Общий принцип работы приложений.
2. Objective-C. Базовый синтаксис. Иерархия классов. Основные классы. UILabel, UIButton, UIImageView, UIView, UITableView, UIScrollView, NSObject, UIBarButtonItem, UITextField, UITextView, UISlider - особенности и применение.
3. Objective-C properties. Управление памятью. Retain count. Retain cycle. Модификаторы.
4. Категории. Расширения. Протоколы. Наследование. Переопределение, перегрузка методов. Конструкторы.

5. Основные контроллеры: UIViewController, UINavigationController, UITableViewController, UIPageViewController, UITabBarController. Различия и сходства, особенности.
6. Жизненный цикл UIViewController, UIApplication.
7. Interface Builder. Связи IBOutlet, IBAction. Возможные actions для контролов, их различия.
8. Storyboard. Xib-файлы. UIStoryboardSegue. Применение и ограничения.
9. AutoresizingMask. Возможности и ограничения.
10. AutoLayout и Size Classes. Возможности и ограничения.
11. Способ создания контроллеров из кода. Создание проекта полностью из кода. Переопределение точки старта. Плюсы и минусы.
12. NSArray, NSDictionary, NSMutableArray, NSMutableDictionary. Оценка сложности. NSInteger, NSNumber.
13. Форматированный вывод строк. NSLog. C-строки и их совместимость с Objective-C строками.
14. Паттерн делегирования и использование источника данных. Примеры.
15. UITableView. Data source, Delegate. Типы таблиц и их особенности. Типы ячеек. Accessory type. Связь с UIScrollView.
16. UITextField. Keyboard types. Основные возможности и случаи применения. UITextView.
17. UIAlertController, его типы и применение.
18. Паттерны MVC, Singleton, Observer. Примеры их применения в iOS.
19. Обмен данными в реализации паттерна MVC.
20. KVO. KVC. Collection operations. Notification center.
21. Блоки кода. Особенности и их применение. Рекурсивный вызов блока.
22. Анимация. Анимируемые свойства. Способы применения анимаций. Опции и особенности.
23. Жесты в UIView. UIGestureRecognizer и его дочерние классы.
24. Персистентность данных. Сериализация, десериализация.
25. NSUserDefaults, NSCoder, NSArray, NSDictionary, Archiving.
26. SQLite. Составление запросов. Запросы с параметрами. Binding.
27. Многопоточность. GCD, NSOperationQueue, NSOperation, NSTimer. Асинхронное выполнение селекторов.

#### *Раздел 4. Системное программирование.*

1. Основные принципы, заложенные в современное системное программное обеспечение.
2. Средства разработки системного программного обеспечения.
3. Процессы. Задания и рабочие наборы.
4. Поток. Многопоточность и MFC. Локальная память потоков. Нити. APC.
5. Способы запуска программы. Ожидание завершения программы, работа с кодом завершения.
6. Создание потока при помощи Windows API, стандартной библиотеки C++. Работа с потоками в MFC.
7. Получение дескрипторов процесса и потока. Использование нитей.

Альтернативные потоки.

8. Определение проблемы синхронизации. Критические секции.
9. Методы синхронизации: блокированные переменные, мьютексы, семафоры, мониторы и другие объекты синхронизации.
10. Безопасная синхронизация. Использование вызова WaitForMultipleObjects. Ожидание объектов в настроенном состоянии.
11. Таймер синхронизации. Синхронизация в MFC.
12. Асинхронный файловый ввод/вывод. Использование потоков. Перекрывающийся ввод/вывод. Порты завершения ввода/вывода.
13. Открытие файла. Синхронные операции чтения/записи файла. Определение EOF при синхронном вводе/выводе. Дублирование дескрипторов файлов. Закрытие файла.
14. Разновидности асинхронного ввода/вывода. Асинхронный ввод/вывод с использованием отдельного программного потока. Определение EOF при асинхронном вводе/выводе.
15. Использование функций типа ReadFileEx и WriteFileEx. Отображение файлов на оперативную память.
16. Потоки и IPC. Обзор механизмов IPC.
17. Реализация памяти общего доступа при помощи DLL.
18. Анонимные каналы (pipes). Именованные каналы.
19. Почтовые слоты. Сокеты. Вызов удаленных процедур RPC. Microsoft Message Queue (MSMQ).
20. Страничная организация памяти. Использование функции VirtualAlloc. Работа с атрибутами страниц.
21. Способы работы с разреженной памятью. Использование нескольких пулов свободной памяти. Изменение уровня защиты страниц. Использование исключений.
22. Куча по умолчанию. Выделение и освобождение памяти в куче. Уплотнение кучи. Проверка корректности данных, расположенных в куче. Увеличение производительности программы с использованием нескольких куч.
23. Цели системы безопасности. Права и привилегии. Заполнение структуры атрибутов безопасности. Работа с идентификаторами SID, ACE и ACL.
24. Типы защищаемых объектов. Использование дескриптора безопасности. Токены и выполнение действий от другого имени. Построение списков ACL.
25. Реализация защиты собственных объектов. Привилегии. Kerberos.
26. Устройство реестра. Открытие ключа реестра. Определение имен подключей.
27. Использование реестра вместо INI-файлов. Создание REG-файлов. Информация о типах файлов. Документирование информации в журналах.
28. Источники событий. Создание файлов сообщений. Системные сообщения. Работа с журналом.
29. Расширения графической оболочки. Основы ATL и MFC. Программы, использующие Icon Tray. Умные указатели. Ярлыки. Консоль MMC.
30. Ярлыки Интернета. Использование Internet Explorer. Использование WebPost API.
31. Поддержка Интернета, встроенная в MFC.

32. Обзор службы Active Directory.
33. Создание простого консольного приложения. Создание консоли для программы с графическим интерфейсом.
34. Создание и использование вспомогательных консольных буферов. Обработка событий, связанных с консолью.
35. Определение дескриптора окна консоли. Использование MFC из консольных программ. Методы доступа к консолям.
36. Службы. Внутреннее строение служб. Доступ к службе. Отладка служб.
37. Объектно-ориентированная служба.

### *Практические задания:*

#### *Раздел 1. Разработка программных модулей.*

1. Разработка программного продукта для предметной области «Учет клиентов компании, предоставляющей услуги мобильной связи» с применением языка программирования C#.
2. Разработка программного продукта для предметной области «Учет клиентов в регистратуре» с применением языка программирования C#.
3. Разработка программного продукта для предметной области «Поликлиника» с применением языка программирования C#.
4. Разработка программного продукта для предметной области «Библиотека» с применением языка программирования C#.
5. Разработка тестирующей программы по дисциплине «Информатика» с кодом на языке программирования C#.
6. Разработка программного продукта для предметной области «Учет товаров в магазине» с применением языка программирования C#.
7. Разработка тестирующей программы по дисциплине «Операционные системы» с кодом на языке программирования C#.
8. Разработка программного продукта для предметной области «Ветеринарная клиника» с применением языка программирования C#.
9. Разработка программного продукта для предметной области «Расчет плотности населения города с учетом численности населения районов» с применением языка программирования C#.
10. Разработка программного продукта для предметной области «Учет продаж автомобилей на предприятии» с применением языка программирования C#.
11. Разработка программного продукта для предметной области «Туристическое агентство» с применением языка программирования C#.
12. Разработка программного продукта для предметной области «Учет постояльцев гостиницы» с применением языка программирования C#.
13. Разработка программного продукта «Расчет потребляемых калорий» с применением языка программирования C#.
14. Разработка игрового приложения «Пятнашки» с применением языка



программирования C#.

15. Разработка приложения для производственных расчетов с применением языка программирования C#.

### *Раздел 2. Поддержка и тестирование программных модулей.*

Разработать спецификации тестовых случаев, соответствующие тесты и провести тестирование для классов (по вариантам). Составить отчет в следующей форме:

Название тестового случая:

Тестирующий:

Тест пройден: Да/Нет (PASS/FAIL)

Степень важности ошибки:

Фатальная (3 уровень - crash)

Серьезная (2 уровень - расхождение в спецификации)

Незначительная (1 уровень - незначительная ошибка)

Описание проблемы:

Как воспроизвести ошибку:

Предлагаемое исправление (необязательно):

Комментарий тестировщика (необязательно):

Варианты классов:

Класс (Тип)

TBearingParam (Примитивный)

TAxleParam (Примитивный)

TCommand (Примитивный)

TLog (Примитивный)

TCommandQueue (Непримитивный)

TStore (Непримитивный)

TTerminalBearing (Непримитивный)

TTerminalAxle (Непримитивный)

TModel (Непримитивный)

MainForm (Непримитивный)

### *Раздел 3. Разработка мобильных приложений.*

1. Создать интерфейсы мобильного приложения для ОС iOS без использования Interface Builder.

Требования:

1. Реализация минимум 10 различных элементов интерфейса без использования Interface Builder;

2. Соответствие реализованных элементов принципам iOS Human Interface Guidelines.

2. Создать пользовательский интерфейс мобильного приложения для ОС iOS с использованием инструмента Interface Builder.

Требования к интерфейсам:

1. Соответствие требованиям iOS Human Interface Guidelines.

2. Использование не менее 10 различных компонентов Interface Builder.
3. Разработать мобильное приложение с функционалом перемещения между экранами мобильного приложения для ОС iOS. Сценарии использования UINavigationController и UITabBarController.

Требования к приложению:

1. Использование UINavigationController;
2. Использование UITabBarController;
3. Реализация сценария передачи данных с одного экрана приложения на следующий.
4. Разработать мобильное приложение с функционалом отображения списка объектов. Добавление и удаление ячеек таблицы во время выполнения программы.

Требования к приложению:

1. Использование UITableViewController или UIViewController с компонентом UITableView;
2. Реализация сценария добавления элемента в отображаемый список;
3. Реализация сценария удаления элемента из отображаемого списка.
5. Расширить класс NSArray. Реализация методов для: 1) подсчета количества строковых переменных в массиве, 2) объединения строковых переменных в одну строку, 3) поиска максимального элемента в массиве.
6. Создать мобильное приложение с автоматической адаптацией интерфейса для разных размеров экранов с использованием Auto Resizing Mask.

Требования к приложению:

1. Корректное отображение в вертикальном режиме;
2. Корректное отображение в горизонтальном режиме.
7. Создать мобильное приложение с автоматической адаптацией интерфейса для разных размеров экранов с использованием Auto Layout.

Требования к приложению:

1. Корректное отображение в вертикальном режиме;
2. Корректное отображение в горизонтальном режиме.
8. Реализовать методы расширения класса NSArray для: 1) возврата массива из элементов, которые вернул блок кода; 2) возврата объекта, который получился суммированием всех элементов массива
9. Разработать мобильное приложение с функционалом получения данных с сервера и отображения данных в виде списка объектов. Использовать решения json-server в качестве тестового сервера.

Требования к приложению:

1. Получение и отображение списка объектов;
2. Добавление объекта в список и отправка данных на сервер;

3. Удаление и изменение объекта списка иправка данных на сервер
10. Разработать мобильное приложение "Тетрис". Реализация функционала игры "Тетрис" в мобильном приложении для ОС iOS.

Требования:

1. Реализация 5 типов фигур;
  2. Реализация анимированного появления и исчезновения фигур;
  3. Реализация управления фигурами посредством жестов.
11. Разработать мобильное приложение для ОС iOS с функционалом сохранения настроек приложения.

Требования:

1. Реализация минимум 3-х способов хранения настроек;
  2. Реализация хранения минимум 5 различных типов настроек.
12. Разработать мобильное приложение для ОС iOS с функционалом подгрузки и отображения данных в разных потоках. Использовать решения json-server в качестве тестового сервера.

Требования к приложению:

1. Получение и отображение списка объектов;
  2. Добавление объекта в список и отправка данных на сервер;
  3. Удаление и изменение объекта списка иправка данных на сервер.
13. Разработать мобильное приложение для ОС iOS с функционалом получения данных с сервера в фоновом режиме.

Требования к приложению:

1. Соответствие требованиям политики Apple по использованию сервисов в фоновом режиме;
2. Реализация функционала получения данных с сервера в фоновом режиме.

#### *Раздел 4. Системное программирование.*

1. Дан массив A из n целых чисел. Найти сумму максимального и минимального элемента в массиве. Поиск максимума и минимума реализовать с помощью подпрограмм функций.
2. Дан файл целых чисел. Выбрать наибольшее из чисел, принадлежащее интервалу [a,b]. Концы интервала a и b вводятся с клавиатуры.
3. Дан текстовый файл F1. Переписать его содержимое в файл F2, сохраняя строчную структуру и удаляя пустые строки.
4. Дан файл целых чисел F1. Создать два новых файла F2 и F3 из отрицательных и положительных чисел соответственно.
5. Даны две символьные строки S1 и S2, содержащие только строчные латинские буквы. Построить строку S3, в которую войдут только общие символы S1 и S2 в алфавитном порядке и без повторений.

6. Даны две символьные строки S1 и S2, содержащие только строчные латинские буквы. Построить строку S3, в которую войдут только общие символы S1 и S2 в обратном порядке и без повторений.
7. Дан файл целых чисел. Определить, сколько раз в нем повторяется максимальное значение.
8. Дан файл целых чисел. Определить, сколько раз в нем повторяется минимальное значение.
9. По координатам вершин треугольника вычислить его периметр, используя подпрограмму вычисления длины отрезка, соединяющего две точки. (длина отрезка=  $\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$ ), где (x1,y1) координаты одной точки, (x2,y2) координаты второй точки отрезка).
10. Дан файл целых чисел F1. Создать два новых файла F2 и F3 из положительных и отрицательных чисел соответственно.
11. Даны два файла целых чисел. Определить, в каком из них больше положительных значений.
12. Составить рекурсивную подпрограмму вычисления N!
13. Дана вещественная матрица размера m\*n. Найти значение наибольшего по модулю элемента матрицы и указать его местоположение в матрице.
14. Определить среднее арифметическое чисел, хранящихся в файле Note.txt.
15. Дан список L из N целых чисел. Удалить первое вхождение максимального элемента в списке.
16. Дан текстовый файл Note.txt. Определить длину самой длинной строки этого файла.
17. Дан файл целых чисел. Выбрать наименьшее из чисел, принадлежащее интервалу [a,b]. Концы интервала a и b вводятся с клавиатуры.
18. Даны два файла целых чисел. Определить, в каком из них меньше положительных значений.
19. Даны два файла целых чисел. Определить, в каком из них больше отрицательных значений.
20. Даны два файла целых чисел. Определить, в каком из них больше нулевых значений.
21. Даны два файла целых чисел. Определить, в каком из них меньше отрицательных значений.
22. Даны два файла целых чисел. Определить, в каком из них меньше нулевых значений.
23. Дана вещественная матрица размера m\*n. Найти значение наименьшего по модулю элемента матрицы и указать его местоположение в матрице.
24. Определить среднее геометрическое чисел, хранящихся в файле Note.txt.

25. Дан список  $L$  из  $N$  целых чисел. Удалить первое вхождение минимального элемента в списке.
26. Дан список  $L$  из  $N$  целых чисел. Удалить последнее вхождение максимального элемента в списке.
27. Дан список  $L$  из  $N$  целых чисел. Удалить последнее вхождение минимального элемента в списке.
28. Дан текстовый файл Note.txt. Определить длину самой короткой строки этого файла.

*Приложение 2*  
*(Образец экзаменационного билета)*

Государственное бюджетное профессиональное образовательное учреждение  
«Южно-Уральский государственный колледж»

<b>РАССМОТРЕНО:</b> На заседании предметно-цикловой комиссии Информационных технологий Председатель ПЦК Назарова Н.А. Протокол № ____ от « ____ » _____ г.	<b>ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ</b> № _____ <b>По учебной дисциплине</b> (МДК) _____ <b>Специальность</b> _____ <b>Курс</b> _____	<b>УТВЕРЖДАЮ:</b> Зам. директора по учебной работе _____ « ____ » _____ г.
1.		
2.		
3.		
4.		

Преподаватель (преподаватели):