Министерство образования и науки Астраханской области
Государственное автономное образовательное учреждение
Астраханской области высшего образования
«Астраханский государственный архитектурно-строительный университет»
(ГАОУ АО ВО «АГАСУ»)

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Наименование дисциплины
Программирование и разработка программного обеспечения
(указывается наименование в соответствии с учебным планом)

По направлению подготовки
09.04.02 «ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ»
(указывается наименование направления подготовки в соответствии с ФГОС)

Направленность (профиль)
«Информационные системы и технологии в строительстве и архитектуре»
(указывается наименование профиля в соответствии с ОПОП)

Кафедра Системы автоматизированного проектирования и моделирования

Квалификация (степень) выпускника **магистр**

Астрахань – 2019

Разработчик:

к.т.н., доцент _____ _____ /Т.В. Хоменко/

(занимаемая должность, ученая степень, ученое звание) (подпись) (инициалы, фамилия)

Рабочая программа рассмотрена и утверждена на заседании кафедры *«Системы автоматизированного проектирования и моделирования»*

Протокол № 10 от 25.05. 2019 г.

Заведующий кафедрой _____ / Т.В. Хоменко
(подпись)

Согласовано:

Председатель МКН *«Информационные системы и технологии»*
направленность (профиль)
«Информационные системы и технологии в строительстве и архитектуре»

_____ /Т.В. Хоменко
(подпись) (инициалы, фамилия)

Начальник УМУ _____ /Н.В. Рассохина/
(подпись) (инициалы, фамилия)

Специалист УМУ _____ / Ю.А. Дудникова /
(подпись) (инициалы, фамилия)

Начальник УИТ _____ / С.В. Тригорьев /
(подпись) (инициалы, фамилия)

Заведующий научной библиотекой _____ /Р.С. Кайдикешова/
(подпись) (инициалы, фамилия)

Содержание

1. Цель освоения дисциплины

Целью освоения дисциплины «Программирование и разработка программного обеспечения» является формирование компетенций обучающихся в соответствии с требованиями Федерального государственного образовательного стандарта высшего образования по направлению подготовки 09.04.02 «Информационные системы и технологии».

2. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

В результате освоения дисциплины обучающийся должен овладеть следующими компетенциями:

ПК-15 – Способен создавать текущие и перспективные проекты в области применения информационных технологий, вести поэтапный контроль исполнения проекта;

ПК-16 – Способен вести сдачу проекта, собирать и анализировать мнения и замечания заказчика по выполнению проекта и предлагать соответствующие решения.

В результате освоения дисциплин, формирующих компетенции ПК-15, ПК-16, обучающийся должен овладеть следующими результатами обучения:

знать:

- основы конфигурационного управления проекта в области ИТ (ПК-15.1);
- основы управления изменениями в проекте (ПК-16.1);

уметь:

- планировать работы в проекте в области ИТ (ПК-15.2.);
- планировать работы в проекте (ПК-16.2);

иметь практический опыт:

- управления сборкой программных базовых элементов конфигурации ИС (ПК-15.3);
- согласования плана управления изменениями с заинтересованными сторонами проекта (ПК-16.3).

3. Место дисциплины в структуре ОПОП магистратуры

Дисциплина Б1.В.03 «Программирование и разработка программного обеспечения» реализуется в рамках Блока 1 «Дисциплины (модули)» части, формируемой участниками образовательных отношений.

Дисциплина базируется на основах, полученных в рамках изучения следующих дисциплин: «Информатика», «Алгоритмы и структуры данных», «Основы программирования».

4. Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по типам учебных занятий) и на самостоятельную работу обучающихся

Форма обучения	Очная	Заочная
1	2	3
Трудоемкость в зачетных единицах:	2 семестр – 4 з.е.; всего – 4 з.е.	3 семестр – 4 з.е.; всего - 4 з.е.
Лекции (Л)	2 семестр –14 часов; всего - 14 часов	3 семестр – 4 часа; всего - 4 часа
Лабораторные занятия (ЛЗ)	2 семестр – 42 часа; всего - 42 часа	3 семестр – 12 часов; всего - 12 часов
Практические занятия (ПЗ)	учебным планом не предусмотрены	учебным планом не предусмотрены
Самостоятельная работа (СР)	2 семестр – 88 часов; всего – 88 часов	3 семестр – 128 часов; всего - 128 часов
Форма текущего контроля:		
Контрольная работа	учебным планом не предусмотрена	учебным планом не предусмотрена
Форма промежуточной аттестации:		
Зачет	учебным планом не предусмотрен	учебным планом не предусмотрен
Экзамен	семестр – 2	семестр – 3
Зачет с оценкой	учебным планом не предусмотрен	учебным планом не предусмотрен
Курсовая работа	учебным планом не предусмотрена	учебным планом не предусмотрена
Курсовой проект	учебным планом не предусмотрен	учебным планом не предусмотрен

5. Содержание дисциплины, структурированное по разделам с указанием отведенного на них количества академических часов и видов учебных занятий

5.1. Разделы дисциплины и трудоемкость по типам учебных занятий и работы обучающихся (в академических часах)

5.1.1. Очная форма обучения

№ п/п	Раздел дисциплины (по семестрам)	Всего часов на раздел	Семестр	Распределение трудоемкости раздела (в часах) по типам учебных занятий и работы обучающихся				Форма текущего контроля и промежуточной аттестации
				контактная			СР	
				Лекции	Лабор. занятия	Практ. занятия		
1	2	3	4	5	6	7	8	9
1	Раздел 1. Структурное программирование	36	2	4	16	-	16	Экзамен
2	Раздел 2. Модульное программирование	36		2	6	-	28	
3	Раздел 3. Объектно-ориентированное программирование	36		6	14	-	16	
4	Раздел 4. Разработка прикладного программного обеспечения	36		2	6		28	
Итого:		144		14	42	-	88	

5.1.2. Заочная форма обучения

№ п/п	Раздел дисциплины (по семестрам)	Всего часов на раздел	Семестр	Распределение трудоемкости раздела (в часах) по типам учебных занятий и работы обучающихся				Форма текущего контроля и промежуточной аттестации
				контактная			СР	
				Лекции	Лабор. занятия	Практ. занятия		
1	2	3	4	5	6	7	8	9
1	Раздел 1. Структурное программирование	36	3	2	6	-	28	Экзамен
2	Раздел 2. Модульное программирование	36		-	1	-	35	
3	Раздел 3. Объектно-ориентированное программирование	36		2	4	-	30	
4	Раздел 4. Разработка прикладного программного обеспечения	36		-	1		35	
Итого:		144		4	12	-	128	

5.2. Содержание дисциплины, структурированное по разделам

5.2.1. Содержание лекционных занятий

№	Наименование раздела дисциплины	Содержание
1.	Раздел 1. Структурное программирование	Планирование работы в проекте в области ИТ. Понятие алгоритма и способы его записи. Классификация языков программирования. Структура программы на языке Pascal. Операторы присваивания, ветвления, цикла и варианта. Простые и составные типы данных. Правила оформления и работа с массивами данных. Методы сортировки. Стандартные функции и процедуры языка Pascal. Операции над строками и множествами. Интерпретаторы и компиляторы. Структура описания и вызова процедур и функций. Фактические и формальные параметры. Глобальные и локальные переменные подпрограмм. Рекурсивные процедуры и функции. Текстовые и типизированные файлы.
2.	Раздел 2. Модульное программирование	Основы конфигурации управления проектом: комбинированный тип данных фиксированного числа компонент (полей) разного типа. Стандартные модули Turbo Pascal. Исполняемый модуль как набор ресурсов, разрабатываемых и хранимых независимо от использующих их программ. Структура программного модуля. Разработка, отладка и модификация программного модуля. Управление изменениями в проекте.
3.	Раздел 3. Объектно-ориентированное программирование	Управление сборкой программных базовых элементов конфигурации ИС: особенности объектно-ориентированных языков программирования. Модульная структура программ. Динамические структуры данных. Структуры, указатели и рекурсивные типы данных. Программирование линейных списков. Элементы объектно-ориентированного программирования. Классы. Наследование. Полиморфизм и динамические объекты. Визуальные компоненты для работы с данными. Общая характеристика визуальных компонентов. Форма – главный компонент приложения. Особенности модальных форм. Однострочный и многострочный редакторы. Простой и комбинированный списки. Сложные элементы интерфейса. Средства для работы с файлами.
4.	Раздел 4. Разработка прикладного программного обеспечения	Создание перспективных проектов в области применения современных технологий программирования. Архитектура «клиент-сервер». Средства разработки Windows-приложений на платформе .NET. Технология Windows Presentation Foundation (WPF). Silverlight – технология для разработки клиентских полнофункциональных веб-приложений (Rich Internet Applications). Технология Windows Communication Foundation (WCF) – модель программирования и среда исполнения для создания, конфигурации и развертывания распределённых сервис-ориентированных приложений. Технология Windows Workflow Foundation (WWF) – технология для разработки бизнес-процессов. Согласование плана управления изменениями по разработке, отладке и модификации программного продукта с заинтересованными сторонами проекта.

5.2.2. Содержание лабораторных занятий

№	Наименование раздела дисциплины	Содержание
1.	Раздел 1. Структурное программирование	Лабораторная работа №1. Управление сборкой программных базовых элементов линейных и разветвляющихся алгоритмов
		Лабораторная работа №2. Управление сборкой программных базовых элементов с оператором выбора
		Лабораторная работа №3. Управление сборкой программных базовых элементов циклических алгоритмов
		Лабораторная работа №4. Управление обработкой программных базовых элементов одномерных и двумерных массивов
		Лабораторная работа №5. Управление сортировкой массивов. Поиск элемента массива
		Лабораторная работа №6. Управление сборкой программных базовых элементов с строковыми типами данных
		Лабораторная работа №7. Управление сборкой программных базовых элементов процедур и функций
		Лабораторная работа №8. Управление сборкой рекурсивных подпрограмм
2.	Раздел 2. Модульное программирование	Лабораторная работа №9. Основы управления: структура программного модуля
		Лабораторная работа №10. Управление сборкой программных элементов с типизированными файлами последовательного доступа
		Лабораторная работа №11. Поэтапный контроль исполнения проекта при разработке, отладке и подключении программного модуля
3.	Раздел 3. Объектно-ориентированное программирование	Лабораторная работа № 12 Основы управления: классы и объекты
		Лабораторная работа № 13 Основы управления: наследование и виртуальные функции
		Лабораторная работа № 14 Основы управления: иерархия объектов и итераторов
		Лабораторная работа № 15 Поэтапный контроль исполнения проекта при обработке событий
4.	Раздел 4. Разработка прикладного программного обеспечения	Лабораторная работа № 16 Планирование работы при перегрузке операций и выводе таблицы средствами MS Word и диаграммы MS EXCEL
		Лабораторная работа № 17 Поэтапный контроль исполнения проекта при реализации шаблонов функций и классов
		Лабораторная работа № 18 Поэтапный контроль исполнения проекта при разработке, отладке и модификации программного продукта

5.2.3. Содержание практических занятий

Учебным планом не предусмотрены.

5.2.4. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине

Очная форма обучения

№	Наименование раздела дисциплины	Содержание	Учебно-методическое обеспечение
1	2	3	4
1.	Раздел 1. Структурное программирование	Изучение теоретического материала по рекомендованной в рабочей программе литературе. Подготовка к выполнению и защите лабораторных работ. Подготовка к экзамену.	[1], [3], [5], [8]
2.	Раздел 2. Модульное программирование	Подготовка к выполнению лабораторных работ по написанию программных модулей. Подготовка к защите лабораторных работ. Подготовка к экзамену.	[2], [4], [7], [8]
3.	Раздел 3. Объектно-ориентированное программирование	Изучение теоретического и практического материала по рекомендованной в рабочей программе литературе. Подготовка к выполнению лабораторных работ. Подготовка к экзамену.	[1], [3], [4], [5], [6], [7]
4.	Раздел 4. Разработка прикладного программного обеспечения	Изучение теоретического и практического материала по рекомендованной в рабочей программе литературе. Подготовка к выполнению лабораторных работ по отладке и модификации программ. Подготовка к экзамену.	[1], [3], [4], [5], [6], [7]

Заочная форма обучения

№	Наименование раздела дисциплины	Содержание	Учебно-методическое обеспечение
1	2	3	4
1.	Раздел 1. Структурное программирование	Изучение теоретического материала по рекомендованной в рабочей программе литературе. Подготовка к выполнению и защите лабораторных работ. Подготовка к экзамену.	[1], [3], [5], [8]
2.	Раздел 2. Модульное программирование	Подготовка к выполнению лабораторных работ по написанию программных модулей. Подготовка к экзамену.	[2], [4], [7], [8]
3.	Раздел 3. Объектно-ориентированное программирование	Изучение теоретического и практического материала по рекомендованной в рабочей программе литературе. Подготовка к выполнению лабораторных работ. Подготовка к экзамену.	[1], [3], [4], [5], [6], [7]
4.	Раздел 4. Разработка прикладного программного обеспечения	Изучение теоретического и практического материала по рекомендованной в рабочей программе литературе. Подготовка к выполнению лабораторных работ по отладке и модификации программ. Подготовка к экзамену.	[1], [3], [4], [5], [6], [7]

5.2.5. Темы контрольных работ

Учебным планом не предусмотрены.

5.2.6. Темы курсовых работ

Учебным планом не предусмотрены.

6. Методические указания для обучающихся по освоению дисциплины

Организация деятельности студента
Лекция В ходе лекционных занятий необходимо вести конспектирование учебного материала, обращать внимание на категории, формулировки, раскрывающие содержание тех или иных явлений и процессов, научные выводы и практические рекомендации. Необходимо задавать преподавателю уточняющие вопросы с целью уяснения теоретических положений, разрешения спорных ситуаций. Целесообразно дорабатывать свой конспект лекции, делая в нем соответствующие записи из литературы, рекомендованной преподавателем и предусмотренной учебной программой.
Лабораторное занятие Работа в соответствии с методическими указания по выполнению лабораторных работ.
Самостоятельная работа Самостоятельная работа студента над усвоением учебного материала по учебной дисциплине может выполняться в помещениях для самостоятельной работы, а также в домашних условиях. Содержание самостоятельной работы студента определяется учебной программой дисциплины, методическими материалами, заданиями и указаниями преподавателя. Самостоятельная работа в аудиторное время может включать: – конспектирование (составление тезисов) лекций; – работу со справочной и методической литературой; – участие в тестировании и др. Самостоятельная работа во внеаудиторное время может состоять из: – повторения лекционного материала; – подготовки к лабораторным занятиям; – изучения учебной и научной литературы; – выполнения заданий, выданных на лабораторных занятиях; – подготовки к тестированию; – выделения наиболее сложных и проблемных вопросов по изучаемой теме, получения разъяснений и рекомендаций по данным вопросам от преподавателей кафедры на их еженедельных консультациях. – проведения самоконтроля путем ответов на вопросы текущего контроля знаний, решения представленных в учебно-методических материалах кафедры задач и тестов.
Подготовка к экзамену Подготовка студентов к экзамену включает три стадии: – самостоятельная работа в течение учебного семестра; – непосредственная подготовка в дни, предшествующие экзамену; – подготовка к ответу на вопросы, содержащиеся в билете.

7. Образовательные технологии

Перечень образовательных технологий, используемых при изучении дисциплины «Программирование и разработка программного обеспечения».

Традиционные образовательные технологии

Обучение дисциплине «Программирование и разработка программного обеспечения» проводится с использованием традиционных образовательных технологий, ориентирующихся на организацию образовательного процесса, предполагающую прямую трансляцию знаний от преподавателя к студенту (преимущественно на основе объяснительно-иллюстративных методов обучения). Учебная деятельность студента носит в таких условиях, как правило, репродуктивный характер. Формы учебных занятий по дисциплине «Программирование и разработка программного обеспечения» с использованием традиционных технологий:

Лекция – последовательное изложение материала в дисциплинарной логике, осуществляемое преимущественно вербальными средствами (монолог преподавателя).

Лабораторное занятие – организация учебной работы с реальными материальными и информационными объектами, экспериментальная работа с аналоговыми моделями реальных объектов.

Интерактивные технологии

По дисциплине «Программирование и разработка программного обеспечения» лекционные занятия проводятся с использованием следующих интерактивных технологий:

Лекция-визуализация - представляет собой визуальную форму подачи лекционного материала средствами ТСО или аудиовидеотехники (видео-лекция). Чтение такой лекции сводится к развернутому или краткому комментированию просматриваемых визуальных материалов (в виде схем, таблиц, графов, графиков, моделей). Лекция-визуализация помогает студентам преобразовывать лекционный материал в визуальную форму, что способствует формированию у них профессионального мышления за счет систематизации и выделения наиболее значимых, существенных элементов.

Проблемная лекция – форма изложения материала, предполагающее постановку проблемных и дискуссионных вопросов, освещение различных научных подходов, авторские комментарии, связанные с различными моделями интерпретации изучаемого материала.

Лекция с разбором конкретных ситуаций – форма, при которой преподаватель на обсуждение ставит не вопросы, а конкретную ситуацию. Ситуация представляется устно или в очень короткой видеозаписи, диафильме, содержащих достаточную информацию для оценки характерного явления и обсуждения. Слушатели анализируют и обсуждают ее сообща, всей аудиторией. Основным содержанием занятия является лекционный материал, а потому преподаватель направляет тему дискуссию для получения достоверных выводов.

По дисциплине «Программирование и разработка программного обеспечения» лабораторные занятия проводятся с использованием следующих интерактивных технологий:

Работа в малых группах – это одна из самых популярных стратегий, так как она дает всем обучающимся возможность участвовать в работе, практиковать навыки сотрудничества, межличностного общения (в частности, умение активно слушать, вырабатывать общее мнение, разрешать возникающие разногласия).

Исследовательский проект – структура приближена к формату научного исследования (доказательство актуальности темы, определение научной проблемы, предмета и объекта исследования, целей и задач, методов, источников, выдвижение гипотезы, обобщение результатов, выводы, обозначение новых проблем).

Лабораторное занятие в форме практикума – организация учебной работы, направленная на решение комплексной учебно-познавательной задачи, требующей от студента применения как научно-теоретических знаний, так и практических навыков.

8. Учебно-методическое и информационное обеспечение дисциплины

8.1. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины

а) основная учебная литература:

1. Рик Гаско. Объектно-ориентированное программирование. Настольная книга программиста. Ред.: Комлев Н.Ю., изд.: Москва, Солон-пресс, 2018. – 298 с.
2. Гавриков М. М., Гринченков Д. В., Иванченко А. Н. Теоретические основы разработки и реализации языков программирования. Учеб. пособие. Изд.: Москва, Кнорус, 2016. -184 с.
3. Карпенков С. Х. Технические средства информационных технологий: учебное пособие - М., Берлин: Директ-Медиа, 2015.http://biblioclub.ru/index.php?page=book_red&id=275367
4. Грошев А.С. Информационные технологии: лабораторный практикум - М., Берлин: Директ-Медиа, 2015. http://biblioclub.ru/index.php?page=book_red&id=434666

б) дополнительная учебная литература:

5. Тарасов С. В. СУБД для программиста. Базы данных изнутри. Изд.: Москва, СОЛОН-Пресс, 2018. – 320 с.
6. Майстренко А. В., Майстренко Н. В. Информационные технологии в науке, образовании и инженерной практике: учебное пособие - Тамбов: Издательство ФГБОУ ВПО «ТГТУ», 2014. http://biblioclub.ru/index.php?page=book_red&id=277993

в) перечень учебно-методического обеспечения:

7. Садчиков, П.Н. Методические указания по выполнению лабораторных работ по дисциплине «Программирование и разработка программного обеспечения». АГАСУ. 2019. 49 с. http://moodle.aucu.ru

г) перечень онлайн курсов:

8. Инженерия программного обеспечения
   https://www.intuit.ru/studies/higher_education/3406/info

8.2. Перечень необходимого лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства, используемого при осуществлении образовательного процесса по дисциплине

- 7-Zip
- Office 365 A1
- Adobe Acrobat Reader DC
- Google Chrome
- VLC media player
- Apache Open Office
- Office Pro Plus Russian OLPNL Academic Edition
- Kaspersky Endpoint Security
- Internet Explorer
- Visual Studio
- Microsoft SQL Server 2016 Express
- Microsoft Azure Dev Tools for Teaching
- Lazarus
- PascalABC.NET.

8.3. Перечень современных профессиональных баз данных и информационных справочных систем, доступных обучающимся при освоении дисциплины

1. Электронная информационно-образовательная среда Университета:

образовательный портал (http://moodle.aucu.ru)

2. Электронно-библиотечная система «Университетская библиотека» (https://biblioclub.ru/)
3. Электронно-библиотечная система «IPRbooks» (www.iprbookshop.ru)
4. Научная электронная библиотека (http://www.elibrary.ru/)
5. Консультант + (http://www.consultant-urist.ru/)
6. Федеральный институт промышленной собственности (https://www1.fips.ru/)
7. Патентная база USPTO (https://www.uspto.gov/patents-application-process/search-patents)

9. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

№ п\п	Наименование специальных помещений и помещений для самостоятельной работы	Оснащенность специальных помещений и помещений для самостоятельной работы
1.	Учебная аудитория для проведения учебных занятий: 414056, г. Астрахань, ул. Татищева, 18, аудитория №207, №209, №211	аудитория №207 Комплект учебной мебели Компьютеры – 15 шт. Стационарный мультимедийный комплект Доступ к информационно – телекоммуникационной сети «Интернет»
		аудитория №209 Комплект учебной мебели Компьютеры – 15 шт. Стационарный мультимедийный комплект Доступ к информационно – телекоммуникационной сети «Интернет»
		аудитория №211 Комплект учебной мебели Компьютеры – 15 шт. Стационарный мультимедийный комплект Доступ к информационно – телекоммуникационной сети «Интернет»
2.	Помещение для самостоятельной работы: 414056, г. Астрахань, ул. Татищева, 18, аудитория №201	аудитория №201 Комплект учебной мебели Компьютеры – 4 шт. Доступ к информационно – телекоммуникационной сети «Интернет»
	414056, г. Астрахань, ул. Татищева, 18б, аудитория №308	аудитория №308 Комплект учебной мебели Компьютеры – 11 шт. Доступ к информационно – телекоммуникационной сети «Интернет»

10. Особенности организации обучения по дисциплине для инвалидов и лиц с ограниченными возможностями здоровья

Для обучающихся из числа инвалидов и лиц с ограниченными возможностями здоровья на основании письменного заявления дисциплина «Программирование и разработка программного обеспечения» реализуется с учетом особенностей психофизического развития, индивидуальных возможностей и состояния здоровья.

Лист внесения дополнений и изменений
в рабочую программу учебной дисциплины

_____«Программирование и разработка программного обеспечения»_____
(наименование дисциплины)

на 20__- 20__ учебный год

Рабочая программа пересмотрена на заседании кафедры
«Системы автоматизированного проектирования и моделирования»,

протокол № ____ от _____20__г.

Зав. кафедрой

_____ _____ /_____/
ученая степень, ученое звание подпись И.О. Фамилия

В рабочую программу вносятся следующие изменения:
1._____
2._____
3._____
4._____
5._____

Составители изменений и дополнений:
_____ _____ /_____/
ученая степень, ученое звание подпись И.О. Фамилия

_____ _____ /_____/
ученая степень, ученое звание подпись И.О. Фамилия

Председатель МКН «Информационные системы и технологии»
профиль «Информационные системы и технологии в строительстве и архитектуре»

_____ _____ /_____/
ученая степень, ученое звание подпись И.О. Фамилия

«____»_____20___г.

Министерство образования и науки Астраханской области
Государственное автономное образовательное учреждение
Астраханской области высшего образования
«Астраханский государственный архитектурно-строительный университет»
(ГАОУ АО ВО «АГАСУ»)

ОЦЕНОЧНЫЕ И МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ

Наименование дисциплины
Программирование и разработка программного обеспечения
(указывается наименование в соответствии с учебным планом)

По направлению подготовки
09.04.02 «ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ»
(указывается наименование направления подготовки в соответствии с ФГОС)

Направленность (профиль)
«Информационные системы и технологии в строительстве и архитектуре»
(указывается наименование профиля в соответствии с ОПОП)

Кафедра Системы автоматизированного проектирования и моделирования

Квалификация выпускника **магистр**

Астрахань – 2019

Разработчик:

к.т.н., доцент		Т.Н. Сапожкова
(занимаемая должность, ученая степень, ученое звание)	(подпись)	(инициалы, фамилия)

Оценочные и методические материалы рассмотрены и утверждены на заседании кафедры *«Системы автоматизированного проектирования и моделирования»*

Протокол № 10 от 25.05 2019 г.

Заведующий кафедрой / _____ / Т.В. Хоменко
(подпись)

Согласовано:

Председатель МКН *«Информационные системы и технологии»*
направленность (профиль)
«Информационные системы и технологии в строительстве и архитектуре»

_____ / Т.В. Хоменко
(подпись) (инициалы, фамилия)

Начальник УМУ _____ К.В. Аксютина
(подпись) (инициалы, фамилия)

Специалист УМУ _____ / Д.А. Федишева
(подпись) (инициалы, фамилия)

Содержание

1. Оценочные и методические материалы для проведения текущего контроля успеваемости и промежуточной аттестации обучающихся по дисциплине

Оценочные и методические материалы являются неотъемлемой частью рабочей программы дисциплины (далее РПД) и представлены в виде отдельного документа.

1.1. Перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы

Индекс и формулировка компетенции N	Индикаторы достижений компетенций, установленные ОПОП	Номер раздела дисциплины (в соответ. с п.5.1 РПД)				Формы контроля с конкретизацией задания
		1	2	3	4	
1	2	3	4	5	6	7
ПК-15 – Способен создавать текущие и перспективные проекты в области применения информационных технологий, вести поэтапный контроль исполнения проекта	Знать:					Экзамен вопросы 1-15 Защита лабораторных работ № 1 - №11 Тестирование вопросы 1-18
	основы конфигурационного управления проекта в области ИТ	X	X	X	X	
	Уметь:					
	планировать работы в проекте в области ИТ	X	X	X	X	
	Иметь практический опыт:					
	управления сборкой программных базовых элементов конфигурации ИС	X	X	X	X	
ПК-16 – Способен вести сдачу проекта, собирать и анализировать мнения и замечания заказчика по выполнению проекта и предлагать соответствующие решения	Знать:					Экзамен вопросы 16-36 Защита лабораторных работ №12 - №18 Тестирование вопросы 19-30
	основы управления изменениями в проекте	X	X	X	X	
	Уметь:					
	планировать работы в проекте	X	X	X	X	
	Иметь практический опыт:					
	согласования плана управления изменениями с заинтересованными сторонами проекта	X	X	X	X	

1.2. Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описание шкал оценивания

1.2.1. Перечень оценочных средств текущего контроля успеваемости

Наименование оценочного средства	Краткая характеристика оценочного средства	Представление оценочного средства в фонде
Защита лабораторной работы	Средство, позволяющее оценить умение и владение обучающегося излагать суть поставленной задачи, самостоятельно применять стандартные методы решения поставленной задачи с использованием имеющейся лабораторной базы, проводить анализ полученного результата работы. Рекомендуется для оценки умений и владений студентов	Темы лабораторных работ и требования к их защите
Тест	Система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающегося	Фонд тестовых заданий

1.2.2. Описание показателей и критериев оценивания компетенций по дисциплине на различных этапах их формирования, описание шкал оценивания

Компетенция, этапы освоения компетенции	Планируемые результаты обучения	Показатели и критерии оценивания результатов обучения			
		Ниже порогового уровня (не удовл.)	Пороговый уровень (удовл.)	Продвинутый уровень (хорошо)	Высокий уровень (отлично)
1	2	3	4	5	6
ПК-15 – Способен создавать текущие и перспективные проекты в области применения информационных технологий, вести поэтапный контроль исполнения проекта	Знает: основы конфигурационного управления проекта в области ИТ	Обучающийся не знает основ конфигурационного управления проектом в области информационных технологий	Обучающийся не знает основ конфигурационного управления проектом в области информационных технологий	Обучающийся знает основы конфигурационного управления проектом в области информационных технологий для строительной сферы и архитектуры в типовых ситуациях	Обучающийся знает основы конфигурационного управления проектом в области информационных технологий для строительной сферы и архитектуры, способен вести поэтапный контроль исполнения проекта

	Умеет: планировать работы в проекте в области ИТ	Обучающийся не умеет планировать работы в проекте в области ИТ, работать с записями на исправление несоответствий	Обучающийся умеет планировать работу в проекте в области информационных технологий, делает ошибки при разработке плановой документации, несоответствий	Обучающийся умеет планировать работу в проекте в области информационных технологий, работать с запросами на исправление несоответствий в типовых ситуациях	Обучающийся умеет планировать работу в проекте в области информационных технологий, вести поэтапный контроль исполнения проекта в ситуациях повышенной сложности
	Имеет практический опыт: управления сборкой программных базовых элементов конфигурации ИС	Обучающийся не имеет практического опыта мониторинга и управления работами проекта в соответствии с установленными регламентами	Обучающийся имеет практический опыт мониторинга, но делает ошибки при управлении работами над проектом в соответствии с установленными регламентами	Обучающийся имеет практический опыт мониторинга и управления работами над проектом в соответствии с установленными регламентами в типовых ситуациях	Обучающийся имеет практический опыт мониторинга и управления работами над проектом в соответствии с установленными регламентами в ситуациях повышенной сложности
ПК-16 – Способен вести сдачу проекта, собирать и анализировать мнения и замечания заказчика по выполнению проекта и предлагать соответствующие решения	Знает: основы управления изменениями в проекте	Обучающийся не знает основы управления изменениями в проекте, не понимает принципы исследования предметной области	Обучающийся не твердо знает основы управления изменениями в проекте и не вполне понимает принципы и методы исследования предметной области	Обучающийся знает и понимает структуру методов и средств управления изменениями в проекте, принципы исследования предметной области в типовых ситуациях	Обучающийся знает и понимает структуру методов и средств управления изменениями в проекте, принципы исследования предметной области в ситуациях повышенной сложности

	Умеет: планировать работы в проекте	Обучающийся не умеет планировать работы в проекте, контролировать выданные поручения	Обучающийся умеет планировать работы в проекте, контролировать поручения по заранее известному алгоритму	Обучающийся умеет планировать работы в проекте, контролировать выданные поручения в типовых ситуациях	Обучающийся умеет планировать работы в проекте, контролировать выданные поручения в ситуациях повышенной сложности
	Имеет практический опыт: согласования плана управления изменениями с заинтересованными сторонами проекта	Обучающийся не имеет практического опыта согласования плана управления изменениями с заинтересованными сторонами проекта	Обучающийся имеет практический опыт согласования плана управления изменениями, при этом не придерживаясь полученного плана	Обучающийся имеет практический опыт согласования плана управления изменениями по анализу требований отдельной предметной области, придерживаясь полученного плана	Обучающийся имеет практический опыт согласования плана управления изменениями с заинтересованными сторонами проекта по анализу требований в соответствии с полученным планом по широкому спектру областей знаний

1.2.3. Шкала оценивания

Уровень достижений	Отметка в 5-бальной шкале	Зачтено/ не зачтено
высокий	«5»(отлично)	зачтено
продвинутый	«4»(хорошо)	зачтено
пороговый	«3»(удовлетворительно)	зачтено
ниже порогового	«2»(неудовлетворительно)	не зачтено

2. Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков, характеризующих этапы формирования компетенций в процессе освоения образовательной программы

ТИПОВЫЕ ЗАДАНИЯ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ:

2.1. Экзамен

а) типовые вопросы к экзамену (см. приложение 1)

б) критерии оценки

При оценке знаний на экзамене учитывается:

1. Уровень сформированности компетенций.
2. Уровень усвоения теоретических положений дисциплины, правильность формулировки основных понятий и закономерностей.
3. Уровень знания фактического материала в объеме программы.
4. Логика, структура и грамотность изложения вопроса.
5. Умение связать теорию с практикой.
6. Умение делать обобщения, выводы.

№ п/п	Оценка	Критерии оценки
1	Отлично	Ответы на поставленные вопросы излагаются логично, последовательно и не требуют дополнительных пояснений. Полно раскрываются причинно-следственные связи между явлениями и событиями. Делаются обоснованные выводы. Демонстрируются глубокие знания базовых нормативно-правовых актов. Соблюдаются нормы литературной речи.
2	Хорошо	Ответы на поставленные вопросы излагаются систематизировано и последовательно. Базовые нормативно-правовые акты используются, но в недостаточном объеме. Материал излагается уверенно. Раскрыты причинно-следственные связи между явлениями и событиями. Демонстрируется умение анализировать материал, однако не все выводы носят аргументированный и доказательный характер. Соблюдаются нормы литературной речи.
3	Удовлетворительно	Допускаются нарушения в последовательности изложения. Имеются упоминания об отдельных базовых нормативно-правовых актах. Неполно раскрываются причинно-следственные связи между явлениями и событиями. Демонстрируются поверхностные знания вопроса, с трудом решаются конкретные задачи. Имеются затруднения с выводами. Допускаются нарушения норм литературной речи.
4	Неудовлетворительно	Материал излагается непоследовательно, сбивчиво, не представляет определенной системы знаний по дисциплине. Не раскрываются причинно-следственные связи между явлениями и событиями. Не проводится анализ. Выводы отсутствуют. Ответы на дополнительные вопросы отсутствуют. Имеются заметные нарушения норм литературной речи.

ТИПОВЫЕ ЗАДАНИЯ ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕГО КОНТРОЛЯ:

2.2. Защита лабораторной работы

а) типовые задания лабораторных работ (см. приложение 2);

б) критерии оценки.

При оценке знаний на защите лабораторной работы учитывается:

1. Уровень сформированности компетенций.
2. Уровень усвоения теоретических положений дисциплины, правильность формулировки основных понятий и закономерностей.
3. Уровень знания фактического материала в объеме программы.
4. Логика, структура и грамотность изложения вопроса.
5. Умение связать теорию с практикой.
6. Умение делать обобщения, выводы.

№ п/п	Оценка	Критерии оценки
1	2	3
1	Отлично	Студент правильно называет метод исследования, правильно называет прибор, правильно демонстрирует методику исследования /измерения, правильно оценивает результат.
2	Хорошо	Студент правильно называет метод исследования, правильно называет прибор, допускает единичные ошибки в демонстрации методики исследования /измерения и оценке его результатов
3	Удовлетворительно	Студент неправильно называет метод исследования, но при этом дает правильное название прибора. Допускает множественные ошибки в демонстрации методики исследования /измерения и оценке его результатов
4	Неудовлетворительно	Студент неправильно называет метод исследования, дает неправильное название прибора. Не может продемонстрировать методику исследования /измерения, а также оценить результат

2.3. Тест

а) типовые вопросы к проведению тестирования (см. приложение 2);

б) критерии оценки.

При оценке знаний по результатам тестов учитывается:

1. Уровень сформированности компетенций.
2. Уровень усвоения теоретических положений дисциплины, правильность формулировки основных понятий и закономерностей.
3. Уровень знания фактического материала в объеме программы.
4. Логика, структура и грамотность изложения вопроса.
5. Умение связать теорию с практикой.
6. Умение делать обобщения, выводы.

№ п/п	Оценка	Критерии оценки
1	2	3
1	Отлично	если выполнены следующие условия: - даны правильные ответы не менее чем на 90% вопросов теста, исключая вопросы, на которые студент должен дать свободный ответ; - на все вопросы, предполагающие свободный ответ, студент дал правильный и полный ответ.
2	Хорошо	если выполнены следующие условия: - даны правильные ответы не менее чем на 75% вопросов теста, исключая вопросы, на которые студент должен дать свободный ответ; - на все вопросы, предполагающие свободный ответ, студент дал правильный ответ, но допустил незначительные ошибки и не показал необходимой полноты.
3	Удовлетворительно	если выполнены следующие условия: - даны правильные ответы не менее чем на 50% вопросов теста, исключая вопросы, на которые студент должен дать свободный ответ; - на все вопросы, предполагающие свободный ответ, студент дал непротиворечивый ответ, или при ответе допустил значительные неточности и не показал полноты.
4	Неудовлетворительно	если студентом не выполнены условия, предполагающие оценку «Удовлетворительно».
5	Зачтено	Выставляется при соответствии параметрам экзаменационной шкалы на уровнях «отлично», «хорошо», «удовлетворительно».
6	Не зачтено	Выставляется при соответствии параметрам экзаменационной шкалы на уровне «неудовлетворительно».

3. Перечень и характеристики процедуры оценивания знаний, умений, навыков, характеризующих этапы формирования компетенций

Процедура проведения текущего контроля успеваемости и промежуточной аттестации обучающихся по дисциплине регламентируется локальным нормативным актом.

Перечень и характеристика процедур текущего контроля
и промежуточной аттестации по дисциплине

№	Наименование оценочного средства	Периодичность и способ проведения процедуры оценивания	Виды выставляемых оценок	Форма учета
1.	Экзамен	Раз в семестр, по окончании изучения дисциплины	По пятибалльной шкале	Ведомость, зачетная книжка, портфолио
2.	Защита лабораторной работы	Систематически на занятиях	По пятибалльной шкале	Лабораторная тетрадь, журнал успеваемости преподавателя
3.	Тест	Систематически на занятиях	По пятибалльной шкале или зачтено / не зачтено	Журнал успеваемости преподавателя

Приложение 1

Экзамен

Типовые вопросы и задания

ПК-15

1. Планирование работы в проекте в области ИТ: выбор языка программирования.
2. Основы конфигурации управления проектом: структура программы.
3. Основы конфигурации программы и управление проектом: простые и составные.
4. Основы конфигурации управления проектом: константы, арифметические операции, условные выражения.
5. Управление сборкой программных базовых элементов при определении и объявление функций.
6. Управление сборкой программных базовых элементов при описании и вызове процедур и функций.
7. Поэтапный контроль исполнения проекта при идентификации фактических и формальных параметров.
8. Основы конфигурации управления проектом: статическая типизация и преобразования типов.
9. Поэтапный контроль исполнения проекта при идентификации глобальных и локальных переменных подпрограмм.
10. Управление конфигурацией и выполнением программы при передаче аргументов по значению и по ссылке.
11. Поэтапный контроль исполнения проекта при разработке, отладке и подключении рекурсивных функций.
12. Управление конфигурацией в графическом режиме программы.
13. Поэтапный контроль исполнения проекта при использовании в программе данных текстовых и типизированных файлов.
14. Управление сборкой программных базовых элементов конфигурации ИС при разделении программы на файлы.
15. Поэтапный контроль исполнения проекта при использовании в программе указателей на функцию как на возвращаемое значение.

ПК-16

16. Анализ мнений и замечаний заказчика по выполнению проекта при использовании сложных элементов интерфейса.
17. Управление изменениями в проекте при использовании динамических объектов.
18. Определение классов при планировании работы в проекте.
19. Конструкторы и инициализация объектов при планировании работы в проекте.
20. Объявление и определение функций класса при планировании работы в проекте.
21. Анализ мнений и замечаний заказчика по выполнению проекта при управлении доступом. Инкапсуляция
22. Управление изменениями в проекте при использовании дружественных функций и классов.
23. Анализ мнений и замечаний заказчика по выполнению проекта при использовании виртуальных функций и их переопределении.

24. Обработка исключений при планировании работы в проекте.

25. Типы последовательных контейнеров при планировании работы в проекте.

26. Базовые типы для работы с потоками при планировании работы в проекте.

27. Анализ мнений и замечаний заказчика по выполнению проекта при чтении и записи текстовых файлов.

28. Анализ мнений и замечаний заказчика по выполнению проекта при переопределении операторов ввода и вывода.

29. Комбинированный тип данных фиксированного числа компонент разного типа.

30. Исполняемый модуль как набор ресурсов, разрабатываемых и хранимых независимо от использующих их программ.

31. Структура программного модуля.

32. Управление изменениями в проекте при реализации особенностей объектно-ориентированных языков программирования.

33. Анализ мнений и замечаний заказчика по выполнению проекта при использовании визуальных компонентов для работы с данными.

34. Управление изменениями в проекте при реализации модальных форм.

35. Анализ мнений и замечаний заказчика по выполнению проекта при использовании однострочных и многострочных редакторов.

36. Анализ мнений и замечаний заказчика по выполнению проекта при построении взаимоотношений «клиент-сервер».

Защита лабораторных работ

Типовые задания

ПК-15

Лабораторная работа №1

УПРАВЛЕНИЕ СБОРКОЙ ПРОГРАММНЫХ БАЗОВЫХ ЭЛЕМЕНТОВ ЛИНЕЙНЫХ И РАЗВЕТВЛЯЮЩИХСЯ АЛГОРИТМОВ

Цель: Ознакомиться с системой программирования Turbo Pascal, получить основные навыки работы с ней, освоить приёмы создания, компиляции и исполнения программы, научиться создавать программы для решения линейных задач.

Теоретические сведения

Работа в системе программирования Turbo Pascal:

- Активация меню: Клавиша <F10> (щелчок левой кнопкой мыши на пункте меню)
- Ввод текста: текст вводится на английском языке. После ввода каждой строки нажимается клавиша Enter. Смена алфавита: на русский язык - <Ctrl>+<Shift>(справа), на английский язык - <Ctrl>+<Shift>(слева)
- Компиляция: меню Compile, команда Compile (клавиши <Alt>+<F9>)
- Запуск программы: меню Run, команда Run (клавиши <Ctrl>+<F9>)
- Открытие нового окна: Каждая программа создается в новом окне. Чтобы открыть новое окно, выберите меню File, команду New
- Переключение между окнами: Клавиши <Alt>+<номер окна>
- Открытие сохраненной программы: меню File, команда Open (клавиша<F3>). В открывшемся окне диалога в поле «Имя» введите полное имя файла. Выберите команду Open.
- Выход из программы: меню File, команда Exit (клавиши <Alt>+<X>).

Разработать программу нахождения значения следующего выражения:

№1 **Ошибка!**-a^3c+b-2 ; №2 **Ошибка!Ошибка! * Ошибка! - Ошибка!** ;

№3 **Ошибка!**x arctg xy ; №4 **Ошибка! - Ошибка!** ;

№5 3 – 4x + (y - $\sqrt{;|x|}$); №6 x - **Ошибка!**+ **Ошибка!**;

№7 ln| (y - $\sqrt{;|x|}$)(x - **Ошибка!**); №8 e^x – x – 2 + $(1 + x)^2$;

№9 **Ошибка!**; №10 **Ошибка!** ;

№11 e^x - **Ошибка!**; №12 **Ошибка!** ;

№13 **Ошибка!**+16xcos(xy) –2; №14 sin**Ошибка!**- sin**Ошибка!Ошибка!**;

№15 x - ln x + **Ошибка!** .

Лабораторная работа №2

УПРАВЛЕНИЕ СБОРКОЙ ПРОГРАММНЫХ БАЗОВЫХ ЭЛЕМЕНТОВ С ОПЕРАТОРОМ ВЫБОРА

Цель: научиться составлять программы с использование оператора множественного выбора Case, продолжить освоение работы в системе программирования Turbo Pascal.

Теоретические сведения

Оператор варианта Case является обобщением оператора if и позволяет сделать выбор из произвольного числа имеющихся вариантов. Он состоит из выражения, называемого ключом выбора, и списка операторов, каждому из которых предшествует список констант

выбора. Как и в операторе if, в операторе Case может присутствовать слово else, имеющее тот же смысл. Ключ выбора может быть целого, символьного или логического типа. Список констант должен иметь тот же тип, что и ключ выбора, и может состоять из одной константы, списка констант, разделенных запятыми или списка констант, заданных диапазоном.

Общий вид:

```
case <ключ выбора> of
        <список 1>: <оператор 1>;
        <список 2>: <оператор 2>;
                  . . .
        <список N>: <оператор N>
     else <оператор>
end;
```

Лабораторная работа №3
УПРАВЛЕНИЕ СБОРКОЙ ПРОГРАММНЫХ БАЗОВЫХ ЭЛЕМЕНТОВ ЦИКЛИЧЕСКИХ АЛГОРИТМОВ

Цель: закрепить практические навыки работы с системой Turbo Pascal, научиться правильно использовать оператор цикла с параметром; научиться составлять программы решения задач с использованием циклических структур.

Теоретические сведения

Для реализации в языке Pascal используется составной оператор, состоящий из операторов for, to, downto, do и при необходимости из операторных скобок. Переменная параметр обязательно объявляется в декларационной части программы и может принадлежать одному из порядковых типов. Если при изменении переменной параметра необходимо использовать переход к следующему значению, то используется оператор to; если переход необходимо осуществить к предыдущему значению, то используется оператор downto. Тогда в общем виде цикл записывается так:

```
    for I:=I0 to In do
begin
  <оператор 1>;
  <оператор 2>;
  ...
  <оператор n>;
end;
```

где I_0 и I_n – начальное и конечное значения.

Цикл с предусловием.

Для реализации циклов с предусловием используется составной оператор, включающий оператор while, do, операторные скобки.

В общем виде цикл реализуется записью:

```
while <условие> do <действие>;
```

Если тело цикла содержит более одного действия, то необходимо использовать операторные скобки:

```
while <условие> do
   begin
     <оператор 1>;
     <оператор 2>;
     ...
```

<оператор n>;
end;

Цикл с постусловием.

Для реализации цикла используется составной оператор, состоящий из операторов repeat и until. В общем виде цикл записывается так:

repeat
<действие>;
until <условие>;

Лабораторная работа №4
УПРАВЛЕНИЕ ОБРАБОТКОЙ ПРОГРАММНЫХ БАЗОВЫХ ЭЛЕМЕНТОВ ОДНОМЕРНЫХ И ДВУМЕРНЫХ МАССИВОВ

Цель: научиться описывать, заполнять, выводить и обрабатывать одномерные и двумерные массивы.

Теоретические сведения

Массив – группа элементов одного типа, объединенных под общим именем.

Описание массивов

Массивы описываются в разделе описания переменных Var.

Общий вид описания одномерного массива:

<имя массива>: array [<нач. индекс>..<кон. индекс>] of <тип элемента>;

где имя - имя переменной-массива; array - ключевое слово, обозначающее, что переменная является массивом; нижний индекс и верхний индекс - целые числа, определяющие диапазон изменения индексов (номеров) элементов массива и, неявно, количество элементов (размер) массива; тип - тип элементов массива.

Общий вид описания двумерного массива:

<имя массива>:array[<m_1>..<m_2>,<n_1>..<n_2>] of <тип>;

где Имя - имя массива; array - слово языка Pascal, показывающее, что описываемый элемент данных - массив; m_1, m_2, n_1, n_2- константы или выражения типа INTEGER, определяющие диапазон изменения индексов и, следовательно, число элементов массива; Тип - тип элементов массива.

Заполнение массива

Под вводом массива понимается ввод значений элементов массива. Ввод удобно реализовать при помощи инструкции FOR. Чтобы пользователь программы знал, ввода какого элемента массива ожидает программа, следует организовать вывод подсказок перед вводом очередного элемента массива. В подсказке обычно указывают индекс элемента массива. Заполнение массива можно производить:

1) с клавиатуры: For i:=1 to n do readln(a[i]);
2) через датчик случайных чисел: Randomize; For i:=1 to n do begin a[i]:=random(i);

Если требуется, чтобы значения элементов массива выбирались из определенного интервала [a,b], то a+Random(b-a+1);

3) через оператор присваивания (по формуле): For i:=1 to n do a[i]:=i*3;

Вывод массива

Если в программе необходимо вывести значения всех элементов массива, то для этого удобно использовать инструкцию FOR, переменная-счётчик которой может быть реализована как индекс элемента массива. Например, For i:=1 to n do writeln(a[i]);

Удаление элементов из одномерного массива.

Для того, чтобы удалить из массива k-ый элемент нужно: найти номер элемента k;

сдвинуть все элементы, начиная с k-го, на один элемент влево; последнему элементу присвоить значение, равное 0; уменьшить количество элементов массива на единицу.

Вставка элемента в одномерный массив.

Вставлять элемент можно до или после данного элемента, номер этого элемента можно вводить с клавиатуры или искать при определенных условиях. Пусть k - это номер элемента, после которого мы должны вставить элемент x. Тогда вставка осуществляется следующим образом: первые k элементов массива остаются без изменения, все элементы, начиная с (k+1)-го, необходимо сдвинуть на один назад, на место (k+1)-го элемента записываем значение x; увеличить количество элементов в массиве на единицу.

Лабораторная работа №5
УПРАВЛЕНИЕ СОРТИРОВКОЙ МАССИВОВ. ПОИСК ЭЛЕМЕНТА МАССИВА

Цель: научиться производить сортировку массивов различными способами, а также отыскивать необходимый элемент в массиве.

Теоретические сведения

Алгоритмы поиска элемента массива.

Пусть значения элементов линейного массива x сформированы. Требуется среди чисел x[1], x[2], …, x[n] найти такое, что x[j] = max(x[1], x[2], …, x[n]). Если в задаче требуется найти порядковый номер этого элемента, то значит необходимо найти еще и значение индекса j. Основная идея алгоритма состоит в том, что переменной max присваивается значение любого элемента массива (чаще всего первого по порядку). В случае нахождения порядкового номера переменной ind присваивается значение индекса этого элемента (т.е. 1).

Далее просматриваются все элементы массива, значения которых сравниваются со значением переменной max. Если окажется, что значение какого-либо элемента массива превосходит значение переменной max, то переменная max меняет свое значение на значение большего элемента. Если поиск наибольшего элемента идет в двумерном массиве, то необходимо просматривать поочередно все элементы каждой из строк.

Методы сортировки массивов.

1. Метод «пузырька»

Идея метода: весь массив просматривается несколько раз, причем при каждом просмотре сравниваются значения двух соседних элементов. Если эти значения следуют не в порядке возрастания, то производится их перестановка. Так происходит до тех пор, пока не будет сделано ни одной перестановки.

2. Метод простых обменов.

Идея метода: весь массив просматривается несколько раз и при каждом просмотре ищется минимальный по значению элемент, который меняется местами с первым, вторым, третьим, …, предпоследним элементов массива и исключается из дальнейшего рассмотрения.

3. Метод вставки и сдвига.

Идея метода: делается предположение, что первые q элементов массива упорядочены, и если q+1-ый элемент меньше, чем какой-либо из первых q, то он записывается на свое место среди упорядоченных, при этом «хвост» массива «сдвигается» к концу.

Лабораторная работа №6

УПРАВЛЕНИЕ СБОРКОЙ ПРОГРАММНЫХ БАЗОВЫХ ЭЛЕМЕНТОВ С СТРОКОВЫМИ ТИПАМИ ДАННЫХ

Цель: научиться составлять программы решения задач с использованием символьных и строковых типов данных.

Теоретические сведения

Строка – последовательность символов. В языке Pascal для строк отведен свой тип данных: string. String – строка размером 255 символов. String[n] – строка размером n символов, где n должно быть больше нуля и меньше или равно 255.

Так же мы можем и вручную присваивать переменной строкового типа значения через оператор присваивания: Stroka:=’строка’;

Таким образом, мы присвоили переменной stroka значение «строка», которое находится между апострофами.

В языке Паскаль типом-множеством называется множество различных сочетаний элементов исходного (базового) типа. Число элементов исходного множества в Turbo-Pascal не может быть больше 256, а порядковые номера элементов должны находиться в пределах от 0 до 255.

Для объявления типа-множества используются зарезервированные слова set of, после которых указываются элементы этого множества, как правило в виде перечисления или диапазона. Объявить тип-множество можно в разделе программы Type или при объявлении переменной в разделе Var.

Результатом операций объединения, разности и пересечения является множество. Результатом операций проверки эквивалентности и вхождения будет значение логического типа.

Лабораторная работа №7
УПРАВЛЕНИЕ СБОРКОЙ ПРОГРАММНЫХ БАЗОВЫХ ЭЛЕМЕНТОВ ПРОЦЕДУР И ФУНКЦИЙ

Цель: научиться составлять программы решения задач с использованием процедур.

Теоретические сведения

В языке Паскаль имеется два вида подпрограмм - процедуры и функции.

Структура описания процедур до некоторой степени похожа на структуру Паскаль-программы: у них также имеются заголовок, раздел описаний и исполняемая часть. Раздел описаний содержит те же подразделы, что и раздел описаний программы: описания констант, типов, меток, процедур, функций, переменных. Исполняемая часть содержит собственно операторы процедур. Одна и та же подпрограмма может вызываться неоднократно, выполняя одни и те же действия с разными наборами входных данных. Параметры, использующиеся при записи текста подпрограммы в разделе описаний, называют формальными, а те, что используются при ее вызове - фактическими.

При вызове фактические параметры как бы подставляются вместо формальных, стоящих на тех же местах в заголовке. В стандарте языка Паскаль передача параметров может производиться двумя способами - по значению и по ссылке. Параметры, передаваемые по значению, называют параметрами-значениями, передаваемые по ссылке - параметрами-переменными. Последние отличаются тем, что в заголовке процедуры (функции) перед ними

ставится служебное слово var. При первом способе (передача по значению) значения фактических параметров копируются в соответствующие формальные параметры. При изменении этих значений в ходе выполнения процедуры (функции) исходные данные (фактические параметры) измениться не могут. При втором способе (передача по ссылке) все изменения, происходящие в теле процедуры (функции) с формальными параметрами, приводят к немедленным аналогичным изменениям соответствующих им фактических параметров.

Функции, это такие подпрограммы, результатом которых обязательно является некоторое значение. Описание функции во многом совпадает с описанием процедуры. Но если имя процедуры используется только для её вызова, то с именем функции, кроме того, связывается её результат.

Функция предполагает обязательную передачу информации из подпрограммы в программу через имя функции. Поэтому раздел операторов обязательно должен содержать хотя бы один оператор, в котором имени функции присваивается значение результата. В противном случае функция не возвратит результат (вернее возвратит произвольный результат).

Лабораторная работа №8
УПРАВЛЕНИЕ СБОРКОЙ РЕКУРСИВНЫХ ПОДПРОГРАММ

Цель: научиться составлять программы решения задач с использованием рекурсивных процедур и функций.

Теоретические сведения

Рекурсия - это способ описания функции или процессов через самих себя (когда процедура или функция сама себя вызывает). В Паскале можно пользоваться именами лишь тогда, когда в тексте программы этому предшествует их описание. Рекурсия является единственным исключением из этого правила. Имя рекурсивной функции можно использовать сразу же, не закончив его описания.

В Паскале возможно применения рекурсии в процедурах и функциях.

Лабораторная работа №9
ОСНОВЫ УПРАВЛЕНИЯ: СТРУКТУРА ПРОГРАММНОГО МОДУЛЯ

Цель: научиться составлять программы решения задач с использованием стандартных модулей и «библиотек» среды Turbo Pascal, в которых хранятся все её процедуры и функции.

Теоретические сведения

Библиотека, содержащая процедуры и функции для поддержки графического режима, носит имя GRAPH.TPU.

Для того, чтобы компилятор «узнавал» названия процедур и функций, содержащихся в библиотеке GRAPH.TPU, необходимо после заголовка программы разместить строчку следующего вида: Uses Graph. До сих пор экран всегда находился в текстовом режиме, поэтому можно было видеть только символы. Для рисования прямых, окружностей и пр. необходимо перевести экран в графический режим.

Для включения графического режима используется процедура InitGraph(Gd, Gm: integer; Path: string) три параметра: Gd является кодом графического адаптера, позволяющего использовать несколько графических режимов, отличающихся количеством цветов и разрешающей способностью, Gm предназначен для того, чтобы указать какой из режимов следует включить, Path является строкой, содержащей путь к файлу, в котором содержится драйвер, необходимый для работы мониторов в графическом режиме.

Лабораторная работа №10
УПРАВЛЕНИЕ СБОРКОЙ ПРОГРАММНЫХ ЭЛЕМЕНТОВ С ТИПИЗИРОВАННЫМИ ФАЙЛАМИ ПОСЛЕДОВАТЕЛЬНОГО ДОСТУПА

Цель: научиться работать с файлами в программе Turbo Pascal.

Теоретические сведения

Файловый тип переменной — это структурированный тип, представляющий собой совокупность однотипных элементов, количество которых заранее не определено.

Структура описания файловой переменной:

Var <имя переменной>: File Of <тип элемента>;

где <тип элемента> может быть любым, кроме файлового.

Текстовый файл можно рассматривать как последовательность символов, разбитую на строки длиной от 0 до 256 символов. Для описания используется стандартный тип Text:

Var F: text; {F - файловая переменная}. Каждая строка завершается маркером конца строки.

Вызов Read(F, Ww), где Ww – переменная типа word, осуществляет чтение из файла F последовательности цифр, которая затем интерпретируется в число, значение которого и будет присвоено переменной Ww. В случае если вместо последовательности цифр идет любая другая последовательность символов, использование такого оператора приводит к ошибке выполнения программы. Открытие текстового файла можно произвести двумя стандартными способами:

- поставить в соответствие файловой переменной имя файла (процедура Assign), открыть новый текстовый файл (процедура Rewrite);
- поставить в соответствие файловой переменной имя файла (процедура Assign), открыть уже существующий файл (процедура Reset).

Текстовый файл в силу своей специфики во время работы допускает только один вид операции: чтение или запись. В связи с этим для работы с текстовыми файлами используется еще одна процедура открытия файла: Append(var F : text); Эта процедура открывает уже существующий файл и позиционирует указатель обработки на конец файла. После этого в текстовый файл можно только добавлять информацию, причем только в конец файла. Для обработки текстовых файлов используются процедуры Read и Write, обеспечивающие соответственно чтение и запись одной строки и более в текстовый файл. Использование специальных разделителей строк позволило ввести в состав языковых средств еще две процедуры: Readln, выполняющую те же действия, что и Read, и дополнительно – чтение до маркера конца строки и переход к новой строке; Writeln, обеспечивающую запись всех величин с обязательной установкой маркера конца строки в файл.

Процедура Read обеспечивает ввод данных общим потоком из одной строки, а Readln приводит к обязательному переходу к следующей строке текстового файла, т. е. ввод данных осуществляется из различных строк. Все вышесказанное в равной мере относится к операциям записи с помощью процедур Write и Writeln.

При организации операций ввода-вывода используются специальные языковые средства в виде функций Eoln, Eof, SeekEoln, SeekEof.

Лабораторная работа №11

ПОЭТАПНЫЙ КОНТРОЛЬ ИСПОЛНЕНИЯ ПРОЕКТА ПРИ РАЗРАБОТКЕ, ОТЛАДКЕ И ПОДКЛЮЧЕНИИ ПРОГРАММНОГО МОДУЛЯ

Цель: научиться составлять программы с использованием модулей в программе Turbo Pascal.

Теоретические сведения

Модуль — это набор ресурсов (функций, процедур, констант, переменных, типов и т.д.), разрабатываемых и хранимых независимо от использующих их программ. Модуль может содержать достаточно большой набор процедур и функций, а также других ресурсов для разработки программ. Существуют стандартные модули Турбо Паскаля (system, crt, graph и т.д.).

Модуль имеет следующую структуру:

```
Unit <имя модуля>;   {заголовок модуля}
Interface {интерфейсная часть}
Implementation {раздел реализации}
Begin
{раздел инициализации модуля}
End.
```

После служебного слова Unit записывается имя модуля, которое должно совпадать с именем файла, содержащего данный модуль.

В разделе Interface объявляются все ресурсы, которые будут в дальнейшем доступны программисту при подключении модуля.

В разделе Implementation описываются все подпрограммы, которые были ранее объявлены. Кроме того, в нем могут содержаться свои константы, переменные, типы, подпрограммы и т.д., которые носят вспомогательный характер и используются для написания основных подпрограмм. В отличие от ресурсов, объявленных в разделе Interface, все, что дополнительно объявляется в Implementation, уже не будет доступно при подключении модуля. При описании основной подпрограммы достаточно указать ее имя, а затем записать тело подпрограммы. Раздел инициализации содержит операторы, которые должны быть выполнены сразу же после запуска программы, использующей модуль.

ПК-16

Лабораторная работа № 12

ОСНОВЫ УПРАВЛЕНИЯ: КЛАССЫ И ОБЪЕКТЫ

Цель. Получить практические навыки реализации классов на C++.

Основное содержание работы

Написать программу, в которой создаются и разрушаются объекты, определенного пользователем класса. Выполнить исследование вызовов конструкторов и деструкторов.

Порядок выполнения работы.

1. Определить пользовательский класс в соответствии с вариантом задания (смотри приложение).
2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.
3. Определить в классе деструктор.
4. Определить в классе компоненты-функции для просмотра и установки полей данных.
5. Определить указатель на компоненту-функцию.
6. Определить указатель на экземпляр класса.
7. Написать демонстрационную программу, в которой создаются и разрушаются объекты пользовательского класса и каждый вызов конструктора и деструктора сопровождается выдачей соответствующего сообщения (какой объект какой конструктор или деструктор вызвал).
8. Показать в программе использование указателя на объект и указателя на компоненту-функцию.

Содержание отчета.

1. Титульный лист: название дисциплины; номер и наименование работы; фамилия, имя, отчество студента; дата выполнения.
2. Постановка задачи. Следует дать конкретную постановку, т.е. указать, какой класс должен быть реализован, какие должны быть в нем конструкторы, компоненты-функции и т.д.
3. Определение пользовательского класса с комментариями.
4. Реализация конструкторов и деструктора.
5. Фрагмент программы, показывающий использование указателя на объект и указателя на функцию с объяснением.
6. Листинг основной программы, в котором должно быть указано, в каком месте и какой конструктор или деструктор вызываются.

Варианты заданий

Описания членов - данных пользовательских классов

1. СТУДЕНТ имя – char* курс – int пол – int(bool)	2. СЛУЖАЩИЙ имя – char* возраст – int рабочий стаж – int	3. КАДРЫ имя – char* номер цеха – int разряд – int
4. ИЗДЕЛИЕ имя – char* шифр – char* количество – int	5. БИБЛИОТЕКА имя – char* автор – char* стоимость – float	6. ЭКЗАМЕН имя студента – char* дата – int оценка – int
1. АДРЕС имя – char* улица – char* номер дома – int	8. ТОВАР имя – char* количество – int стоимость – float	2. КВИТАНЦИЯ номер – int дата – int сумма – float
10. ЦЕХ имя – char* начальник – char* количество работающих – int	11. ПЕРСОНА имя – char* возраст – int пол – int(bool)	12. АВТОМОБИЛЬ марка – char* мощность – int стоимость – float

13. СТРАНА имя – char* форма правления – char* площадь – float	14. ЖИВОТНОЕ имя – char* класс – char* средний вес – int	15. КОРАБЛЬ имя – char* водоизмещение – int тип – char*

Лабораторная работа № 13
ОСНОВЫ УПРАВЛЕНИЯ: НАСЛЕДОВАНИЕ И ВИРТУАЛЬНЫЕ ФУНКЦИИ

Цель. Получить практические навыки создания иерархии классов и использования статических компонентов класса.

Основное содержание работы

Написать программу, в которой создается иерархия классов. Включить полиморфные объекты в связанный список, используя статические компоненты класса. Показать использование виртуальных функций.

Порядок выполнения работы.

1. Определить иерархию классов (в соответствии с вариантом).
2. Определить в классе статическую компоненту - указатель на начало связанного списка объектов и статическую функцию для просмотра списка.
3. Реализовать классы.
4. Написать демонстрационную программу, в которой создаются объекты различных классов и помещаются в список, после чего список просматривается.
5. Сделать соответствующие методы не виртуальными и посмотреть, что будет.
6. Реализовать вариант, когда объект добавляется в список при создании, т.е. в конструкторе (смотри пункт 6 следующего раздела).

Содержание отчета.

1. Титульный лист: название дисциплины; номер и наименование работы; фамилия, имя, отчество студента; дата выполнения.
2. Постановка задачи. Следует дать конкретную постановку, т.е. указать, какие классы должны быть реализованы, какие должны быть в них конструкторы, компоненты-функции и т.д.
3. Иерархия классов в виде графа.
4. Определение пользовательских классов с комментариями.
5. Реализация конструкторов с параметрами и деструктора.
6. Реализация методов для добавления объектов в список.
7. Реализация методов для просмотра списка.
8. Листинг демонстрационной программы.

Варианты заданий.

Перечень классов:

1) студент, преподаватель, персона, завкафедрой;
2) служащий, персона, рабочий, инженер;
3) рабочий, кадры, инженер, администрация;
4) деталь, механизм, изделие, узел;
5) организация, страховая компания, судостроительная компания, завод;
6) журнал, книга, печатное издание, учебник;
7) тест, экзамен, выпускной экзамен, испытание;
8) место, область, город, мегаполис;

9) игрушка, продукт, товар, молочный продукт;
10) квитанция, накладная, документ, чек;
11) автомобиль, поезд, транспортное средство, экспресс;
12) двигатель, двигатель внутреннего сгорания, дизель, турбореактивный двигатель;
13) республика, монархия, королевство, государство;
14) млекопитающие, парнокопытные, птицы, животное;
15) корабль, пароход, парусник, корвет.

Лабораторная работа № 14
ОСНОВЫ УПРАВЛЕНИЯ: ИЕРАРХИЯ ОБЪЕКТОВ И ИТЕРАТОРОВ

Цель. Получить практические навыки создания объектов-групп и использования методов-итераторов.

Порядок выполнения работы.

1. Дополнить иерархию классов лабораторной работы № 2 классами “группа”.

Например, для предметной области ФАКУЛЬТЕТ можно предложить классы “факультет”, “студенческая группа”, “кафедра”. Рекомендуется создать абстрактный класс – “подразделение”, который будет предком всех групп и абстрактный класс TObject, находящийся во главе всей иерархии.

2. Написать для класса-группы метод-итератор.

3. Написать процедуру или функцию, которая выполняется для всех объектов, входящих в группу (смотри примеры в приложении).

4. Написать демонстрационную программу, в которой создаются, показываются и разрушаются объекты-группы, а также демонстрируется использование итератора.

Содержание отчета.

1. Титульный лист.
2. Постановка задачи.
3. Иерархия классов.
4. Иерархия объектов.
5. Определение классов (добавленных или измененных по сравнению с лабораторной работой № 2).
6. Реализация для одного не абстрактного класса-группы всех методов.
7. Реализация итератора.
8. Реализация передаваемой итератору функции.
9. Листинг демонстрационной программы.

Варианты запросов.

1. Имена всех лиц мужского (женского) пола.
2. Имена студентов указанного курса.
3. Имена и должность преподавателей указанной кафедры.
4. Имена служащих со стажем не менее заданного.
5. Имена служащих заданной профессии.
6. Имена рабочих заданного цеха.
7. Имена рабочих заданной профессии.
8. Имена студентов, сдавших все (заданный) экзамены на отлично (хорошо и отлично).
9. Имена студентов, не сдавших все (хотя бы один) экзамен.
10. Имена всех монархов на заданном континенте.
11. Наименование всех деталей (узлов), входящих в заданный узел (механизм).

12. Наименование всех книг в библиотеке (магазине), вышедших не ранее указанного года.

13. Названия всех городов заданной области.

14. Наименование всех товаров в заданном отделе магазина.

15. Количество мужчин (женщин).

16. Количество студентов на указанном курсе.

17. Количество служащих со стажем не менее заданного.

18. Количество рабочих заданной профессии.

19. Количество инженеров в заданном подразделении.

20. Количество товара заданного наименования.

21. Количество студентов, сдавших все экзамены на отлично.

22. Количество студентов, не сдавших хотя бы один экзамен.

23. Количество деталей (узлов), входящих в заданный узел (механизм).

24. Количество указанного транспортного средства в автопарке (на автостоянке).

25. Количество пассажиров во всех вагонах экспресса.

26. Суммарная стоимость товара заданного наименования.

27. Средний балл за сессию заданного студента.

28. Средний балл по предмету для всех студентов.

29. Суммарное количество учебников в библиотеке (магазине).

30. Суммарное количество жителей всех городов в области.

31. Суммарная стоимость продукции заданного наименования по всем накладным.

32. Средняя мощность всех (заданного типа) транспортных средств в организации.

33. Средняя мощность всех дизелей, обслуживаемых заданной фирмой.

34. Средний вес животных заданного вида в зоопарке.

35. Среднее водоизмещение всех парусников на верфи (в порту).

Лабораторная работа № 15
ПОЭТАПНЫЙ КОНТРОЛЬ ИСПОЛНЕНИЯ ПРОЕКТА ПРИ ОБРАБОТКЕ СОБЫТИЙ

Цель. Получить практические навыки разработки объектно-ориентированной программы, управляемой событиями.

Основное содержание работы.

Написать интерактивную программу, выполняющую команды, вводимые пользователем с клавиатуры.

Порядок выполнения работы.

1. Разобрать пример, представленный в приложении. Ответить на следующие вопросы:

а) какова здесь иерархия классов?

б) какова здесь иерархия объектов?

в) как КАЛЬКУЛЯТОРУ передаются аргументы операции? где они хранятся? Каким образом получают к ним доступ устройства СЛОЖЕНИЯ, ВЫЧИТАНИЯ и т.д. ?

г) как обрабатываются события группой?

д) каковы все маршруты события TEvent?

е) как выполняются HandleEvent всех классов?

2. Выбрать группу объектов, которые будут обрабатывать события (это не могут быть объекты, приведенные в приложении).

3.Для выбранной группы объектов определить перечень операций, которые должны выполняться по командам пользователя.

4.Определить вид командной строки <код_операции><параметры>. Решить вопросы:

как кодируются операции? какие передаются параметры?

5. Определить иерархию объектов. Если необходимо, добавить новые объекты (группы объектов).

6. Определить иерархию классов. Если необходимо, добавить новые классы.

7. Определить, какой объект в программе играет роль приложения. В случае необходимости добавить в иерархию классов класс TApp. Решить, в каком классе будет метод Execute, организующий главный цикл обработки событий.

8. Определить и реализовать необходимые для обработки событий методы.

9. Написать основную функцию (main).

Содержание отчета.

1. Титульный лист.
2. Постановка задачи.
3. Схема иерархии классов.
4. Схема иерархии объектов.
5. Описание маршрута, который проходит событие TEvent от формирования до очистки.
6. Определения классов.
7. Реализация методов обработки событий GetEvent, Exicute, EndExec, Valid.
8. Реализация всех методов (для всех классов) HandleEvent.
9. Листинг функции main().

Лабораторная работа № 16
ПЛАНИРОВАНИЕ РАБОТЫ ПРИ ПЕРЕГРУЗКЕ ОПЕРАЦИЙ И ВЫВОДЕ ТАБЛИЦЫ СРЕДСТВАМИ MS WORD И ДИАГРАММЫ MS EXCEL

Цель. Получить практические навыки работы в среде BC++5.02 и создания EasyWin-программы. Получить практические навыки создания абстрактных типов данных и перегрузки операций в языке C++.

Основное содержание работы.

Определить и реализовать класс – абстрактный тип данных. Определить и реализовать операции над данными этого класса. Написать и выполнить Easy Win-программу полного тестирования этого класса.

Порядок выполнения работы.

1. Выбрать класс АТД в соответствии с вариантом.
2. Определить и реализовать в классе конструкторы, деструктор, функции Input (ввод с клавиатуры) и Print (вывод на экран), перегрузить операцию присваивания.
3. Написать программу тестирования класса и выполнить тестирование.
4. Дополнить определение класса заданными перегруженными операциями (в соответствии с вариантом).
5. Реализовать эти операции. Выполнить тестирование.

Содержание отчета.

1. Титульный лист.
2. Конкретное задание с указанием номера варианта, реализуемого класса и операций.
3. Определение класса.
4. Обоснование включения в класс нескольких конструкторов, деструктора и операции присваивания.
5. Объяснить выбранное представление памяти для объектов реализуемого класса.

6. Реализация перегруженных операций с обоснованием выбранного способа (функция – член класса, внешняя функция, внешняя дружественная функция).

7. Тестовые данные и результаты тестирования.

Вопросы для самоконтроля.

1. Что такое абстрактный тип данных?
2. Приведите примеры абстрактных типов данных.
3. Каковы синтаксис/семантика “операции-функции”?
4. Как можно вызвать операцию-функцию?
5. Нужно ли перегружать операцию присваивания относительно определенного пользователем типа данных, например класса? Почему?
6. Можно ли изменить приоритет перегруженной операции?
7. Можно ли изменить количество операндов перегруженной операции?
8. Можно ли изменить ассоциативность перегруженной операции?
9. Можно ли, используя дружественную функцию, перегрузить оператор присваивания?
10. Все ли операторы языка C++ могут быть перегружены?
11. Какими двумя разными способами определяются перегруженные операции?
12. Все ли операции можно перегрузить с помощью глобальной дружественной функции?
13. В каких случаях операцию можно перегрузить только глобальной функцией?
14. В каких случаях глобальная операция-функция должна быть дружественной?
15. Обязателен ли в функции operator параметр типа “класс” или “ссылка на класс”?
16. Наследуются ли перегруженные операции?
17. Можно ли повторно перегрузить в производном классе операцию, перегруженную в базовом классе?
18. В чем отличие синтаксиса операции-функции унарной и бинарной операции?
19. Приведите примеры перегрузки операций для стандартных типов.
20. Перегрузите операцию “+” для класса “комплексное число”.
21. Перегрузите операции “<”,”>”,”==” для класса “строка символов”.

Типовые варианты заданий

1. АТД – множество с элементами типа char. Дополнительно перегрузить следующие операции:

+ – добавить элемент в множество(типа char + set);

+ – объединение множеств;

= = – проверка множеств на равенство.

2. АТД – множество с элементами типа char. Дополнительно перегрузить следующие операции:

- – удалить элемент из множества (типа set-char);

* – пересечение множеств;

< – сравнение множеств.

3. АТД – множество с элементами типа char. Дополнительно перегрузить следующие операции:

- – удалить элемент из множества (типа set-char);

> – проверка на подмножество;

!= – проверка множеств на неравенство.

Лабораторная работа № 17

ПОЭТАПНЫЙ КОНТРОЛЬ ИСПОЛНЕНИЯ ПРОЕКТА ПРИ РЕАЛИЗАЦИИ ШАБЛОНОВ ФУНКЦИЙ И КЛАССОВ

Цель. Получить практические навыки создания шаблонов и использования их в программах C++.

Основное содержание работы.

Создать шаблон заданного класса и использовать его для данных различных типов.

Порядок выполнения работы.

1. Создать шаблон заданного класса. Определить конструкторы, деструктор, перегруженную операцию присваивания (“=”) и операции, заданные в варианте задания.
2. Написать программу тестирования, в которой проверяется использование шаблона для стандартных типов данных.
3. Выполнить тестирование.
4. Определить пользовательский класс, который будет использоваться в качестве параметра шаблона. Определить в классе необходимые функции и перегруженные операции.
5. Написать программу тестирования, в которой проверяется использование шаблона для пользовательского типа.
6. Выполнить тестирование.

Содержание отчета.

1. Титульный лист: название дисциплины, номер и наименование работы, фамилия, имя, отчество студента, дата выполнения.
2. Постановка задачи.

Следует дать конкретную постановку, т.е указать шаблон какого класса должен быть создан, какие должны быть в нем конструкторы, компоненты-функции, перегруженные операции и т.д.

То же самое следует указать для пользовательского класса.

3. Определение шаблона класса с комментариями.
4. Определение пользовательского класса с комментариями.
5. Реализация конструкторов, деструктора, операции присваивания и операций, которые заданы в варианте задания.
6. То же самое для пользовательского класса.
7. Результаты тестирования. Следует указать для каких типов и какие операции проверены и какие выявлены ошибки (или не выявлены)

Вопросы для самоконтроля.

1. В чем смысл использования шаблонов?
2. Каковы синтаксис/семантика шаблонов функций?
3. Каковы синтаксис/семантика шаблонов классов?
4. Напишите параметризованную функцию сортировки массива методом обмена.
5. Определите шаблон класса “вектор” – одномерный массив.
6. Что такое параметры шаблона функции?
7. Перечислите основные свойства параметров шаблона функции.
8. Как записывать параметр шаблона?
9. Можно ли перегружать параметризованные функции?

10. Перечислите основные свойства параметризованных классов.

11. Может ли быть пустым список параметров шаблона? Объясните.

12. Как вызвать параметризованную функцию без параметров?

13. Все ли компонентные функции параметризованного класса являются параметризованными?

14. Являются ли дружественные функции, описанные в параметризованном классе, параметризованными?

15. Могут ли шаблоны классов содержать виртуальные компонентные функции?

16. Как определяются компонентные функции параметризованных классов вне определения шаблона класса?

Типовые варианты заданий

1.Класс – одномерный массив. Дополнительно перегрузить следующие операции:

* – умножение массивов;

[] – доступ по индексу.

2. Класс – одномерный массив. Дополнительно перегрузить следующие операции:

int() – размер массива;

[] – доступ по индексу.

3. Класс – одномерный массив. Дополнительно перегрузить следующие операции:

[] – доступ по индексу;

= = – проверка на равенство;

!= – проверка на неравенство.

Лабораторная работа № 18
ПОЭТАПНЫЙ КОНТРОЛЬ ИСПОЛНЕНИЯ ПРОЕКТА ПРИ РАЗРАБОТКЕ, ОТЛАДКЕ И МОДИФИКАЦИИ ПРОГРАММНОГО ПРОДУКТА

Цель. Освоить технологию обобщенного программирования с использованием библиотеки стандартных шаблонов (STL) языка C++.

Основное содержание работы.

Написать три программы с использованием STL. Первая и вторая программы должны демонстрировать работу с контейнерами STL, третья – использование алгоритмов STL.

Порядок выполнения работы.

Написать и отладить три программы. Первая программа демонстрирует использование контейнерных классов для хранения встроенных типов данных.

Вторая программа демонстрирует использование контейнерных классов для хранения пользовательских типов данных.

Третья программа демонстрирует использование алгоритмов STL.

В программе № 1 выполнить следующее:

1. Создать объект-контейнер в соответствии с вариантом задания и заполнить его данными, тип которых определяется вариантом задания.

2. Просмотреть контейнер.

3. Изменить контейнер, удалив из него одни элементы и заменив другие.

4. Просмотреть контейнер, используя для доступа к его элементам итераторы.

5. Создать второй контейнер этого же класса и заполнить его данными того же типа, что и первый контейнер.

6. Изменить первый контейнер, удалив из него n элементов после заданного и добавив затем в него все элементы из второго контейнера.

7. Просмотреть первый и второй контейнеры.

В программе № 2 выполнить то же самое, но для данных пользовательского типа.

В программе № 3 выполнить следующее:

1. Создать контейнер, содержащий объекты пользовательского типа. Тип контейнера выбирается в соответствии с вариантом задания.

2. Отсортировать его по убыванию элементов.

3. Просмотреть контейнер.

4. Используя подходящий алгоритм, найти в контейнере элемент, удовлетворяющий заданному условию.

5. Переместить элементы, удовлетворяющие заданному условию в другой (предварительно пустой) контейнер. Тип второго контейнера определяется вариантом задания.

6. Просмотреть второй контейнер.

7. Отсортировать первый и второй контейнеры по возрастанию элементов.

8. Просмотреть их.

9. Получить третий контейнер путем слияния первых двух.

10. Просмотреть третий контейнер.

11 Подсчитать, сколько элементов, удовлетворяющих заданному условию, содержит третий контейнер.

12.Определить, есть ли в третьем контейнере элемент, удовлетворяющий заданному условию.

Содержание отчета.

1. Титульный лист.

2. Постановка задач.

3. Определение пользовательского класса.

4. Определения используемых в программах компонентных функций для работы с контейнером, включая конструкторы.

5. Объяснение этих функций.

6. Объяснение используемых в программах алгоритмов STL.

7. Определения и объяснения, используемых предикатов и функций сравнения.

Тест

Типовые вопросы

ПК-15

1. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: в чем сущность модульного программирования:

Варианты ответа:
A) в разбиении программы на отдельные функционально независимые части;
B) в разбиении программы на отдельные равные части;
C) в разбиение программы на процедуры и функции;

2. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: можно ли сочетать модульное и структурное программирование:

Варианты ответа:
A) да;
B) нет.

3. При управлении сборкой базовых элементов программы может ли модуль включать несколько процедур или функций:

Варианты ответа:
A) да;
B) нет.

4. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: в чем заключается независимость модуля:

Варианты ответа:
A) в написании, отладке и тестировании независимо от остальных модулей;
B) в разработке и написании независимо от других модулей;
C) в независимости от работы основной программы.

5. При управлении сборкой базовых элементов программы сократится ли размер программы, если ее написать в виде набора модулей:

Варианты ответа:
A) нет;
B) да.

6. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: можно ли сочетать структурное программирование с модульным:

Варианты ответа:
A) можно;
B) нельзя;
C) только в особых случаях.

7. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: любую ли программу можно привести к структурированному виду:

Варианты ответа:
A) любую;
B) не все;
C) нельзя.

8. При управлении сборкой базовых элементов программы разрешается ли использование оператора GO TO при структурном программировании:

Варианты ответа:
A) нет;
B) да;
C) иногда.

9. При управлении сборкой базовых элементов программы разрешается ли использование циклов при структурном программировании:

Варианты ответа:
A) да;
B) нет.

10. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: Что такое объект, в объектно-ориентированное программировании:

Варианты ответа:
A) тип данных;
B) структура данных;
C) событие;
D) обработка событий;
E) использование стандартных процедур.

11. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: какое утверждение верно:

Варианты ответа:
A) предки наследуют свойства родителей;
B) родители наследуют свойства потомков;
C) потомки не могут иметь общих предков;
D) потомки наследуют свойства родителей.

12. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: Может ли дочерний элемент иметь двух родителей:

Варианты ответа:
A) да;
B) нет;
C) только для визуальных элементов;
D) если их свойства совпадают.

13. При управлении сборкой базовых элементов программы могут ли два визуальных компонента иметь общего предка:

Варианты ответа:
A) да;
B) нет;
C) если их свойства совпадают;
D) если их методы совпадают.

14. При управлении сборкой базовых элементов программы приводит ли изменение свойств к изменению поведения экземпляра:

Варианты ответа:
A) нет;
B) только для визуальных;
C) только НЕ для визуальных;
D) да.

15. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: можно ли свойствам присваивать значения:

Варианты ответа:
A) да (всегда);
B) не всегда;
C) нет.

16. При управлении сборкой базовых элементов программы могут ли могут ли два различных объекта реагировать на событие по-разному:

Варианты ответа:
A) да;
B) нет.

17. При управлении сборкой базовых элементов программы могут ли два экземпляра одного объекта реагировать на событие по-разному:

Варианты ответа:
A) да;
B) нет.

18. Анализируя особенности перспективных проектов в области применения информационных технологий, ответить на вопрос: какие этапы проектирования можно объединять:

Варианты ответа:
A) технический и рабочий;
B) эскизный и рабочий;
C) технический и эскизный.

ПК-16

19. Планируя работы в проекте при модульном программировании, желательно, чтобы модуль имел:

Варианты ответа:
A) большой размер;
B) небольшой размер;
C) фиксированный размер;
D) любой размер.

20. При планировании работы в проекте достоинством модульного программирования является:

Варианты ответа:
A) создание программы по частям в произвольном порядке;
B) не требует компоновки;
C) всегда дает эффективные программы;
D) снижает количество ошибок.

21. При планировании работы в проекте недостатком модульного программирования является:

Варианты ответа:
A) увеличивает трудоемкость программирования;
B) усложняет процедуру комплексного тестирования;
C) снижает быстродействие программы;
D) не позволяет выполнять оптимизацию программы.

22. Планируя работы в проекте при структурном программировании, задача выполняется:
Варианты ответа:
A) поэтапным разбиением на более легкие задачи;
B) без участия программиста;
C) объединением отдельных модулей программы.

23. Инкапсуляция это:
Варианты ответа:
A) определение новых типов данных;
B) определение новых структур данных;
C) объединение переменных, процедур и функций в одно целое;
D) разделение переменных, процедур и функций;

24. Наследование это:
Варианты ответа:
A) передача свойств экземплярам;
B) передача свойств предкам;
C) передача свойств потомкам;
D) передача событий потомкам.

25. Полиморфизм это:
Варианты ответа:
A) изменение поведения потомков, имеющих общих предков;
B) передача свойств по наследству;
C) изменение поведения потомков на разные события;
D) изменение поведения экземпляров, имеющих общих предков;

26. При планировании работы в проекте есть ли различие в поведении объекта и экземпляра того же типа:
Варианты ответа:
A) да;
B) если у них есть общий предок;
C) нет;
D) если у них нет общего предков.

27. При планировании работы в проекте процесс преобразования постановки задачи в план алгоритмического или вычислительного решения это:
Варианты ответа:
A) проектирование;
B) анализ требований;
C) программирование;
D) тестирование.

28. Этап разработки программы, на котором дается характеристика области применения программы:
Варианты ответа:
A) техническое задание;
B) эскизный проект;
C) технический проект;
D) внедрение;
E) рабочий проект.

29. Планируя работы в проекте, укажите правильную последовательность создания программы:

Варианты ответа:
A) формулирование задачи, анализ требований, проектирование, программирование;
B) анализ требований, проектирование, программирование, тестирование, отладка;
C) анализ требований, программирование, проектирование, тестирование;
D) анализ требований, проектирование, программирование, модификация, трассировка;
E) формулирование задачи, анализ требований, программирование, проектирование, отладка.

30. Согласовав план управления изменениями с заинтересованными сторонами проекта, уточнение структуры входных и выходных данных, разработка алгоритмов, определение элементов интерфейса проводятся на уровне:

Варианты ответа:
A) технического проекта;
B) рабочего проекта;
C) эскизного проекта.