МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»
(БГТУ им. В. Г. Шухова)

УТВЕРЖДАЮ
Директор института ЭИТУС
_____ А. В. Белоусов
«20» мая 2021 г.

РАБОЧАЯ ПРОГРАММА

дисциплины (модуля)

Программирование и основы алгоритмизации

Направление подготовки (специальность):
27.03.04 Управление в технических системах

Направленность программы (профиль, специализация):
Управление и информатика в технических системах

Квалификация:
бакалавр

Форма обучения
очная

Институт Энергетики, информационных технологий и управляющих систем

Кафедра Технической кибернетики

Белгород 2021

Рабочая программа составлена на основании требований:

- Федерального государственного образовательного стандарта высшего образования 27.03.04 Управление в технических системах (уровень бакалавриата), утвержденного приказом Министерства науки и высшего образования Российской Федерации № 871 от 31 июля 2020 г.

- учебного плана, утвержденного ученым советом БГТУ им. В. Г. Шухова в 20 21 году.

Составитель (составители):

—		А. В. Крюков
(ученая степень и звание)	(подпись)	(инициалы, фамилия)

Рабочая программа обсуждена на заседании кафедры
«14» 05 20 21 г., протокол № 9

Заведующий кафедрой:

д-р техн. наук, проф.		В. Г. Рубанов
(ученая степень и звание)	(подпись)	(инициалы, фамилия)

Рабочая программа согласована с выпускающей(ими) кафедрой(ами)
Технической кибернетики
(наименование кафедры/кафедр)

Заведующий кафедрой:

д-р техн. наук, проф.		В. Г. Рубанов
(ученая степень и звание)	(подпись)	(инициалы, фамилия)

«14» 05 20 21 г.

Рабочая программа одобрена методической комиссией института
«20» 05 20 21 г., протокол № 9

Председатель:

канд. техн. наук, доц.		А. Н. Семернин
(ученая степень и звание)	(подпись)	(инициалы, фамилия)

1. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

Категория (группа) компетенций	Код и наименование компетенции	Код и наименование индикатора достижения компетенции	Наименование показателя оценивания результата обучения по дисциплине
Использование современных профессиональных технологий в профессиональной деятельности	ОПК-6. Способен разрабатывать и использовать алгоритмы и программы, современные информационные технологии, методы и средства контроля, диагностики и управления, пригодные для практического применения в сфере своей профессиональной деятельности	ОПК-6.1. Разрабатывает и анализирует различные алгоритмы для решения вычислительных, инженерных, экономических и других типов прикладных задач	**Знать:** основные типы алгоритмов и их использование для решения вычислительных, инженерных, экономических и других типов прикладных задач; основные структуры данных, способы их представления и обработки. **Уметь:** читать и отлаживать программы на языке программирования; выбирать и использовать базовые структуры данных для организации сложных управляющих и информационных структур; разрабатывать алгоритмы решения и программировать задачи обработки данных в различных предметной области; разрабатывать проект тестирования программы, выполнять тестирование и отладку программ. **Владеть:** терминологией предмета; основными приемами алгоритмизации и программирования на языках PascalABC.Net, C++.
		ОПК-6.2. Использует технологию структурного программирования при создании программ обработки сложных структур данных	**Знать:** методы и технологии программирования, о методах структурного и модульного программирования; способы описания и представления алгоритмов. **Уметь:** создавать программы на языке программирования по их описанию; решать задачи широкого класса с использованием среды программирования и соответствующих алгоритмов и методов; использовать технологию структурного программирования при создании программ обработки сложных структур данных. **Владеть:** навыками использования компьютерной техники и различных сред программирования в своей профессиональной и учебной деятельности.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

1. Компетенция ОПК-6. Способен разрабатывать и использовать алгоритмы и программы, современные информационные технологии, методы и средства контроля, диагностики и управления, пригодные для практического применения в сфере своей профессиональной деятельности.

Данная компетенция формируется следующими дисциплинами.

Стадия	Наименования дисциплины
1	Программирование и основы алгоритмизации
2	Базы данных
3	Учебная ознакомительная практика
4	Выполнение, подготовка к процедуре защиты и защита выпускной квалификационной работы

3. ОБЪЕМ ДИСЦИПЛИНЫ

Общая трудоемкость дисциплины составляет 7 зач. единиц, 252 часов.
Форма промежуточной аттестации: курсовая работа; экзамен.

Вид учебной работы	Всего часов	Семестр № 3
Общая трудоемкость дисциплины, час	252	252
Контактная работа (аудиторные занятия), в том числе:	**90**	**90**
лекции	34	34
лабораторные	34	34
практические	17	17
групповые консультации в период теоретического обучения и промежуточной аттестации	5	5
Самостоятельная работа студентов, включая индивидуальные и групповые консультации, в том числе:	**162**	**162**
курсовой проект	0	0
курсовая работа	36	36
расчетно-графическое задание	0	0
индивидуальное домашнее задание	0	0
самостоятельная работа на подготовку к аудиторным занятиям (лекции, практические занятия, лабораторные занятия)	90	90
экзамен	36	36

4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1. Наименование тем, их содержание и объем

Курс 2. Семестр 1.

№ п/п	Наименование раздела (краткое содержание)	Объем на тематический раздел по видам учебной нагрузки, час			
		Лекции	Практические занятия	Лабораторные занятия	Самостоятельная работа на подготовку к аудиторным занятиям
1	2	3	4	5	6
1.	**Введение в алгоритмизацию и программирование**				
1.1	**Методологии программирования.** Основные понятия и определения. История и эволюция. Классификация. Этапы решения задач на ЭВМ. Понятие алгоритма. Исполнитель, система команд исполнителя. Свойства алгоритмов. Способы записи алгоритмов. Принципы структурного программирования. Основные алгоритмические структуры и их суперпозиции. **Жизненный цикл программы.** Модели жизненного цикла. Основные этапы разработки программы. Обеспечение сопровождаемости программного средства. Тестирование и отладка программного средства. Принципы и виды отладки.	2	—	—	4
1.2	**Синтаксис и семантика формального языка.** Естественные и формальные языки. Понятия о синтаксисе и семантике формального языка. Инструментарий технологии программирования. Языки и среды программирования – состав, классификация. Транслятор, компилятор.	2	—	—	4
2.	**Структурный подход в программировании**				
2.1	**Основные конструкции алгоритмических языков.** Общие конструкции алгоритмических языков: алфавит, величина (тип, имя и значение). Выражение. Тип выражения. Арифметическое выражение. Символьное выражение. Логическое выражение. Стандартные функции. Структура программы.	2	2	—	4
2.2	**Простые типы языка программирования.** Общая характеристика языка PascalABC.Net, C++. Структуры данных: упорядоченность, однородность, способ доступа. Определение констант. Описание переменных. Стандартные типы данных. Целые типы. Символьный и булевский типы данных. Эквивалентность и совместимость типов. Типы, определяемые программистом: перечисляемый, интервальный. Тип дата-время.	4	—	—	4
2.3	**Основные операторы языка.** Перечень операторов PascalABC.Net, C++. Оператор присваивания. Операторы (процедуры) ввода-вывода. Управление выводом данных в консольном режиме (простейшее формати-	4	4	8	20

	рование). Условный оператор. Логические выражения. Оператор множественного ветвления. Операторы цикла: с предусловием, с постусловием, с параметром.				
2.4	**Структурированные типы языка программирования высокого уровня.** Массивы. Примеры задач с численными, символьными, булевскими массивами. Строковый тип данных. Записи. Оператор присоединения. Записи с вариантами. Множественный тип. Задание множественного типа и множественной переменной. Операции над множествами. Операции отношения. Примеры задач на множественный тип. Файлы. Понятие логического и физического файлов. Файловые типы. Общие процедуры для работы с файлами. Типизированные файлы. Текстовые файлы. Нетипизированные файлы и процедуры ввода-вывода. Прямой и последовательный доступ к компонентам файлов.	4	4	12	20
2.5	**Алгоритмы поиска и сортировки.** Простой и бинарный поиск. Сортировки: выбором, обменом, вставкой. Анализ сложности алгоритмов на примере сортировок.	4	—	—	6
3.	**Модульное программирование. Проектирование типов данных**				
3.1	Процедуры и функции. Модули. Подпрограммы. Формальные параметры. Параметры-значения, параметры-переменные, параметры-константы. Локальные и глобальные идентификаторы подпрограмм. Процедуры и функции. Рекурсия. Внешние подпрограммы. Модули. Общая структура модуля. Подпрограммы в модулях. Компиляция и использование модулей.	4	2	4	8
3.2	**Организация динамических структур данных (абстрактных типов данных):** стек, очередь, двоичное дерево поиска. Динамические структуры. Динамическое распределение памяти. Виды списков. Примеры использования списков. Организация динамических структур данных: стек, очередь, двоичное дерево поиска.	4	2	4	8
4. Объектно-ориентированное программирование					
4.1	**Введение в объектно-ориентированное программирование.** Введение в объектно-ориентированное программирование (ООП) и проектирование. Инкапсуляция, наследование, полиморфизм. Примеры задач.	2	3	—	5
4.2	**Объектно-событийное и объектно-ориентированное программирование.** Идеология программирования под Windows. Событие и сообщение. Виды событий. События от мыши и клавиатуры. Программирование управления событиями. Обработка исключительных событий. Основы визуального программирования. Компонент. Иерархия компонентов.	2	—	6	5
	ВСЕГО	**34**	**17**	**34**	**90**

4.2. Содержание практических (семинарских) занятий

№ п/п	Наименование раздела дисциплины	Тема практического (семинарского) занятия	Колич. часов	Самостоятельная работа на подготовку к аудиторным занятиям
семестр № 3				
1.	Структурный подход в программировании	Представление алгоритмов в виде блок-схем	2	2
2.	Структурный подход в программировании	Запись математических выражений на алгоритмическом языке.	2	2
3.	Структурный подход в программировании	Основные алгоритмы по обработке числовых данных.	2	2
4.	Структурный подход в программировании	Основные по обработке числовых массивов.	2	2
5.	Структурный подход в программировании	Алгоритмы обработки строкой информации.	2	2
6.	Модульное программирование. Проектирование типов данных	Запись отдельных подпрограмм.	2	2
7.	Модульное программирование. Проектирование типов данных	Описание абстрактных типов данных с использованием динамических списков.	2	2
8.	Объектно-событийное и объектно-ориентированное программирование.	Классы. Объекты. Реализация методов классов.	3	3
		ВСЕГО:	**17**	**17**

4.3. Содержание лабораторных занятий

№ п/п	Наименование раздела дисциплины	Тема лабораторного занятия	Колич. часов	Самостоятельная работа на подготовку к аудиторным занятиям
семестр № 3				
1.	Структурный подход в программировании	Обработка числовых массивов.	4	4
2.	Модульное программирование. Проектирование типов данных	Подпрограммы. Рекурсивные алгоритмы.	4	4
3.	Структурный подход в программировании	Обработка строкового типа.	4	4
4.	Структурный подход в программировании	Применение типа «множество» к решению задач.	4	4
5.	Структурный подход в программировании	Комбинированный тип. Тип запись.	4	4
6.	Структурный подход в программировании	Обработка информации из файлов.	4	4
7.	Модульное программирование. Проектирование типов данных	Динамические переменные, указатели. Обработка списков при помощи указателей.	4	4
8.	Объектно-ориентированное программирование	ООП при решении геометрических задач.	6	6
		ВСЕГО:	**34**	**34**

4.4. Содержание курсового проекта/работы

Целью курсовой работы является выработка у студентов практических навыков по проектированию программ, их отладке и документированию.

Выполнение курсовой работы начинается с разработки технического задания и завершается составлением отчета, в котором должно содержаться описание всей проделанной работы.

Данные цели проявляются через следующие конкретные задачи курсовой работы:

- систематизация, закрепление, углубление и расширение теоретических знаний, полученных при изучении данной дисциплины, а также приобретение практических навыков решения комплексных задач;
- привитие навыков самостоятельной работы по подбору литературы, работы с научной литературой и иными информационными источниками;
- умение самостоятельно систематизировать и излагать знания, полученные в процессе самостоятельного изучения литературы;
- привитие навыков научно-исследовательской работы, использование анализа и самостоятельных выводов.

В результате выполнения курсовой работы студент должен научиться:

- создавать программу в соответствии с основными этапами ее разработки;
- строить схему алгоритма работы программы в соответствии с требованиями ГОСТ 19.701-90;
- грамотно тестировать программу;
- анализировать результаты работы программы и делать соответствующие выводы.

Процесс выполнения работы состоит из следующих этапов:

- выбор темы и беседа с руководителем;
- сбор материала, поиск литературы по теме, подготовка библиографии, составление личного рабочего плана;
- подготовка первого варианта;
- сдача первого варианта курсовой работы руководителю;
- доработка текста по замечаниям, окончательное оформление;
- представление работы на кафедре.

Тематика курсовых работ разрабатывается преподавателем, ежегодно дополняется и уточняется. Темы курсовых работ рассматриваются и утверждаются на заседании комиссии. Студенты выбирают тему курсовой работы самостоятельно, из предложенного списка. Незначительное изменение темы разрешается только по согласованию с преподавателем.

На выполнение курсовой работы предусмотрено 36 часов самостоятельной работы студента.

Перечень тем курсовых работ:

Вариант 1 Разработать программу для поиска пути в лабиринте.
Вариант 2 Разработать программу для разбиения текста, находящегося в файле, на строки длиной около 80 символов с переносом слов. Исходный и обработанный тексты хранятся в файле. Для разбиения слова на части для переноса использовать следующие правила: – две идущие подряд гласные можно разделить, первой из них предшествует согласная, а за второй идет хотя бы одна буква (буква й при этом рассматривается вместе с предшествующей гласной как единое целое); – две идущие подряд согласные можно разделить, если первой из них предшествует гласная, а той части слова, которая идет за второй согласной, имеется хотя бы одна гласная (буквы ь, ъ вместе с предшествующей согласной рассматриваются как единое целое); – если не удается применить указанные выше два пункта, то следует попытаться разбить слово так, чтобы первая часть содержала более чем одну букву и оканчивалась на гласную, а вторая содержала хотя бы одну гласную. Вероятность правильного разбиения увеличивается, если предварительно воспользоваться хотя бы неполным списком приставок, содержащих гласные, и попытаться, прежде всего, выделить из слова такую приставку.
Вариант 3 Разработать программу-калькулятор. Калькулятор позволяет использовать четыре арифметических действия, для выполнения расчетов можно использовать скобки, выполнять расчеты в десятичной и двоичной системах счисления, переводить числа из двоичной системы счисления в десятичную.
Вариант 4 Разработать программу для работы с двоичными деревьями. Реализовать следующие функции: загрузку дерева из файла, сохранение дерева в файле, добавление вершины с проверкой на дублирование, удаление вершины, все виды обхода дерева, просмотр дерева в традиционном представлении (корень вверху, литьевые вершины внизу.)
Вариант 5 Разработать программу для игры «Угадывание слова». Слова для угадывания хранятся в файле. Предусмотреть подсчет числа попыток и ограничить их количество. Программа должна работать в режиме «человек-машина» и «машина-человек».
Вариант 6 Разработать программу для игры «Жизнь». Игра моделирует жизнь поколений гипотетической колонии живых клеток, которые выживают, размножаются или погибают в соответствии со следующими правилами. Клетка выживает, если и только если она имеет двух или трех соседей из восьми возможных. Если у клетки только один сосед или вовсе ни одного, она погибает в изоляции. Если клетка имеет четырех или более соседей, она погибает от перенаселения. В любой пустой позиции, у которой ровно три соседа, в следующем поколении появляется новая клетка. Предусмотреть задание размеров поля и случайную или ручную расстановку клеток.
Вариант 7 Разработать программу для игры в крестики-нолики. Для игрового поля можно задавать произвольные размеры.
Вариант 8 Разработать справочно-информационную систему «Справочная система авиакомпании». Система должна содержать базу данных со следующей информацией: номер рейса, пункт отправления, пункт назначения, тип самолета, время отправления, время прибытия, дни выполнения рейсов, цена билета. Система должна подбирать рейсы (с учетом стыковок рейсов в течение одних суток) по минимальной стоимости билетов. Разработать средства для работы с базой данных: загрузка базы, редактирование записей в базе, добавление (с проверкой на дублирование) и удаление записей, сохранение базы.

Вариант 9 Разработать программу для игры в морской бой. Игровое поле – 10х10 позиций. Корабли на игровом поле: четыре одноклеточных, три двухклеточных, два трехклеточных и один четырехклеточный. Корабли расставляются случайным образом. Программа должна работать в режиме «человек-машина».
Вариант 10 Разработать программу для игры «Быки и коровы». Требуется угадать случайное четырехзначное число. Называются пробные цифры. Если цифра по значению и позиции совпала с цифрой в исходном числе, то эта цифра – «корова». Если же цифра совпала по значению, но не совпала по позиции с цифрой в исходном числе, то эта цифра – «бык». Программа должна работать в режиме «человек-машина» и «машина-человек». Например, если загадано число 1294, а названо число 1429, то это одна «корова» и три «быка». Программа должна работать в режиме «человек-машина» и «машина-человек».
Вариант 11 Разработать справочно-информационную систему «Справочная система железнодорожной компании». Система должна содержать базу данных со следующей информацией: номер рейса, пункт отправления, пункт назначения, тип вагона, время отправления, время прибытия, дни выполнения рейсов, цена билета. Система должна подбирать рейсы (с учетом пересадок в течение одних суток) по минимальному времени в пути. Разработать средства для работы с базой данных: загрузка базы, редактирование записей в базе, добавление (с проверкой на дублирование) и удаление записей, сохранение базы.
Вариант 12 Разработать программу для нахождения пути коня на шахматной доске, начинающегося на заданном поле шахматной доски и заканчивающегося на другом. Никакое поле не должно встречаться в маршруте дважды. Представить возможные решения в наглядном виде.
Вариант 13 Игра в слова Составить программу, позволяющую компьютеру и человеку играть в слова. Предварительно программа объясняет правила игры и позволяет уточнить их в любой момент. Тематикой игры могут быть по выбору города, животные, растения и т.д. Тематику из предложенных компьютером (не менее 5) выбирает человек. Для игры компьютер использует собственную базу данных (для каждой тематики свою), хранящуюся в виде текстового файла. Если названное человеком слово отсутствует в базе, уточняется, правильно ли оно названо, и в случае правильности заносится в базу, иначе уточняется. Правила игры: называется слово, и другой игрок должен предложить другое, начинающееся с той буквы, на которую оканчивается названное.

4.5. Содержание расчетно-графического задания, индивидуальных домашних заданий

Выполнение индивидуальных домашних заданий и расчетно-графических заданий не предусмотрено учебным планом дисциплины.

5. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕГО КОНТРОЛЯ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

5.1. Реализация компетенций

1. Компетенция ОПК-6. Способен разрабатывать и использовать алгоритмы и программы, современные информационные технологии, методы и средства контроля, диагностики и управления, пригодные для практического применения в сфере своей профессиональной деятельности.

Наименование индикатора достижения компетенции	Используемые средства оценивания
ОПК-6.1. Разрабатывает и анализирует различные алгоритмы для решения вычислительных, инженерных, экономических и других типов прикладных задач	защита лабораторных работ; тестирование; курсовая работа; экзамен
ОПК-6.2. Использует технологию структурного программирования при создании программ обработки сложных структур данных	защита лабораторных работ; тестирование; курсовая работа; экзамен

5.2. Типовые контрольные задания для промежуточной аттестации

5.2.1. Перечень контрольных вопросов (типовых заданий) для экзамена / дифференцированного зачета / зачета

Промежуточная аттестация осуществляется в конце семестра после завершения изучения дисциплины в форме экзамена.

Экзамен включает 2 теоретических вопроса и 3 практических заданий. Для подготовки к ответу на вопросы и задания билета, который студент вытаскивает случайным образом, отводится время в пределах 120 минут. После ответа на теоретические вопросы билета, преподаватель задает дополнительные вопросы.

Распределение вопросов и заданий по билетам находится в закрытом для студентов доступе. Ежегодно по дисциплине на заседании кафедры утверждается комплект билетов для проведения экзамена по дисциплине. Экзамен является наиболее значимым оценочным средством и решающим в итоговой отметке учебных достижений студента.

Перечень контрольных вопросов (типовых заданий) для экзамена:

1. **Типы данных, их структура:**
   - структура данных; классификация структур данных;
   - физическая и логическая структура данных;
   - основные типы данных в языке программирования Pascal, C++, их особенности;
   - понятие совместимости по присвоению. Проблемы совместимости;
   - логический тип. Основные процедуры и функции при работе с ним;
   - символьный тип. Основные процедуры и функции при работе с ним.
2. **Числовой тип данных:**
   - основные целочисленные форматы. Диапазоны представления чисел в них;

- основные форматы вещественных чисел. Представление переменных вещественного типа в памяти;
- погрешность округления и вычислительная погрешность. Проблема сравнения вещественных чисел;
- виды округлений вещественных чисел. Реализация различных алгоритмов округления;
- арифметические выражения. Операции в арифметических выражениях. Операции *div* и *mod*. Стандартные арифметические функции и процедуры;
- побитовые операции над целыми числами:

- общий принцип работы побитовых операций (*and, or, not, xor, shl, shr*);
- применение побитовых операции для обработки числовой информации (выделение отдельных бит, установка/сброс отдельных бит в структуре байта, обмен двух целочисленных переменных местами, проверка числа на четность).

3. Разветвляющиеся структуры.

- условный оператор полной и неполной структуры;
- вложенный условный оператор;
- запись сложных логических выражений. Логические операции;
- понятие «условного» и «безусловного» перехода. Оператор безусловного перехода goto. Операторы *break, exit, halt, continue*;
- оператор выбора (особенности и примеры работы с ним). Тип выражения-переключателя в операторе выбора варианта;
- решение задача поиска *max(a, b, c)* разными способами.

4. Циклические структуры:

- общая структура цикла;
- виды циклов. Понятие «детерминированного», «итерационного» цикла, «цикла с предусловием», «цикла с постусловием». Моделирование цикла *repeat* с помощью цикла *while*;
- вложенные циклы. Метод окаймления и метод последовательной детализации. Примеры использования;
- цикл *foreach*. Примеры использования;
- циклы, управляемые флагами;
- зацикливание, бесконечные циклы;
- оператор принудительного завершения цикла (*break, continue*). Эквиваленты таких циклических структур без использования этих операторов;
- применение оператора *goto* для реализации различных циклических структур;
- переборные задачи;
- примеры использования циклов для реализации стандартных алгоритмов (вычисления $n!$, $n!!$, a^n, табулирование функции, вычисление суммы цифр целого числа, определение простоты числа, разложение целого числа на простые сомножители).

5. Статические массивы.

- основные термины и особенности при работе с массивами;
- тип индексов массива. Обращение к элементу по индексу. Выход за границы диапазона;
- хранение статических массивов в памяти;

- реализация основных алгоритмических структур при работе с массивами:

- заполнение массива и вывод его на экран;
- вывод содержимого двумерного массива в виде таблицы;
- задание многомерных массивов как констант;
- вычисление суммы и произведения элементов массива;
- поиск максимального (минимального) элемента;
- поиск элемента в массиве методом деления пополам;
- вставка новых элементов в массив;
- удаление элементов из массива;
- перестановка элементов в одномерном и двумерном массиве (например, изменение на обратный порядок, поменять местами строки или столбцы);
- сдвиг влево/вправо элементов массива, циклический сдвиг;
- слияние двух упорядоченных массивов в один упорядоченный;

- методы сортировки массивов: описание метода и фрагмент программного кода. Рекурсивный метод сортировки массива.

6. Множества.

- основные термины и особенности при работе со множествами;
- отличия множества от массива данных;
- примеры программ по обработки множеств;
- хранение множества в памяти;
- реализация основных алгоритмических структур при работе с множествами:

- заполнение множества и вывод его элементов на экран;
- задание константы типа множества;
- проверка на принадлежность элемента множеству;
- проверка на вхождения одного множества во второе;

- создание пользовательского типа множества с использованием массивов со структурой вида: *set_arr1: array[1..n] of boolean*. Реализация основных операций при работе с такой структурой (занесение элемента во множество, определение принадлежности элемента множеству, объединение и пересечение множеств);
- создание пользовательского типа множества с использованием массивов со структурой вида: *set_arr: array[0..n] of <тип элемента>*. Реализация основных операций при работе с такой структурой (занесение элемента во множество, определение принадлежности элемента множеству, объединение и пересечение множеств);
- создание пользовательского типа множества с использованием массивов со структурой вида: *set_arr: array[0..n] of byte*, где один элемент массива (1 байт) хранит информацию о принадлежности 8 элементов. Реализация основных операций при работе с такой структурой (занесение элемента во множество, определение принадлежности элемента множеству, объединение и пересечение множеств).

7. Строковый тип:

- основные процедуры и функции при работе со строками;
- отличия массива символов (*array of char*) от строкового типа (*string*);

- примеры программ по обработки переменных строкового типа - формирование строки ‘ABCDEF…Z’, сумма всех цифр строки; удаление всех вхождений подстроки в строку.

8. Комбинированный тип данных – записи:

- особенности декларирования и работы с переменными типа «запись»;
- поля записей. Инициализация записей;
- вложенное описание записей. Доступ к полям записи. Оператор *with*;
- совместное описание информационных полей и методов внутри записи;
- примеры программ по сортировке массива записей. Индексная сортировка (основная идея, реализация метода).

9. Подпрограммы:

- назначение процедур и функций. Сравнение процедур и функций;
- стандартные подпрограммы;
- достоинства подпрограмм;
- оператор *exit*. Переменная *result*;
- обмен информацией между подпрограммами. Формальные и фактические параметры. Вызов по ссылке и по значению;
- локальные и глобальные переменные и подпрограммы. Принцип локальности при описании переменных;
- время жизни и область видимости переменной;
- примеры употребления подпрограмм в решении задач;
- методы разработки программ «сверху вниз» и «снизу вверх». Преимущества и недостатки каждого метода.
- модульная организация программы. Общая структура модуля. Достоинства использования библиотек. Примеры.

10. Построение «гибких» подпрограмм:

- подпрограммы с переменным числом параметров;
- параметры по умолчанию;
- понятие о перезагрузке имен подпрограмм;
- обобщенные подпрограммы (Generic);
- Процедурный тип;

11. Рекурсия:

- понятие рекурсии, рекурсивных и итерационных алгоритмов. Простейшие примеры рекурсии;
- глубина рекурсии. Дерево рекурсивных вызовов;
- преимущества и недостатки в использовании рекурсивных подпрограмм; примеры перевода итерационного алгоритма в рекурсивный.
- основные формы рекурсивных подпрограмм;
- понятие косвенной рекурсии;
- переполнение программного стека. Рекурсивное зацикливание;

- примеры рекурсивных алгоритмов (арифметическая и геометрическая прогрессии как частные случаи рекуррентных последовательностей; факториал числа; числа Фибоначчи; степень числа; нахождение минимального элемента в массиве);
- рекурсивный метод сортировки массива.

12. Файлы.

- преимущества файлов. Классификация файлов по типу компонент и по способу доступа;
- процедуры и функции при работе с файлами;
- основные типы ошибок при работе с файлами; их обработка. Операторы *try except* и *try finally* при работе с файлами;
- понятие файловой переменной, файлового указателя;
- типизированные файлы. Процедуры и функции для работы с типизированными файлами; примеры программ по обработке типизированных файлов;
- текстовые файлы, их описание и основные отличия от типизированных файлов. Способы обмена с текстовыми файлами;
- стандартные текстовые файлы *Input* и *Output*;
- примеры задач по обработке текстовых файлов.

13. Динамические переменные.

- статическая и динамическая память; статические и динамические переменные; их основные отличия между собой. Ошибки при неправильной работе с динамической памятью. Достоинства и недостатки при работе с динамическими объектами.
- указатели и их объявление. Типы указателей. Особенности работы с указателем *pointer*. Неявные указатели.
- операторы для работы с указателями;
- бестиповые указатели. Совместимость по присваиванию и приведение типов указателей;
- доступ к памяти, имеющей другое внутреннее представление;
- примеры решения задач с указателями.

14. Динамические структуры данных.

- динамические массивы. Изменение длины динамического массива;
- реализация простейших операций при работе с динамическим массивом;
- виды списков. Сравнение списков и массивов;
- реализация простейших операций при работе с однонаправленным списком через запись – заполнение списка, удаление элементов списка, просмотр списка;
- реализация простейших операций при работе с однонаправленным списком через класс – заполнение списка, удаление элементов списка, просмотр списка;
- основные операции с линейными двусвязными списками: инициализация, вставка элемента в начало и конец, вставка элемента в середину перед и после данного, удаление элемента в начале, середине и конце списка, проход по списку;

15. Классы.

- отличие классов от стандартных записей;
- свойства и методы классов. Понятие конструктора. Переменная *Self*. Шаблоны классов;
- примеры объявления классов.

16. Объектно-ориентированное программирование:

- наследование, примеры. Цели наследования. Замещающие методы. Вызов унаследованного конструктора;
- принцип «Открыт-закрыт» и его роль при проектировании сложных систем. Учет будущих изменений. Пример: очередь с подсчетом элементов;
- инкапсуляция;
- определение полиморфизма. Раннее и позднее связывание;
- позднее связывание и виртуальные методы. Переопределение виртуального метода.

Типовые практические задания для экзамена:

ЗАДАНИЕ 1.

Описать процедуру **SwapArr(A, N)**, которая сортирует массив **mas** размера **N** в порядке убывания по полю **chislo** (**mas** — массив типа **Tmas**, являющийся одновременно входным и выходным параметром).

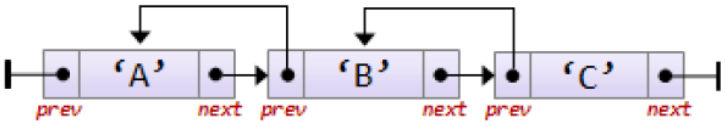
```
Type
  Tmas = record  chislo:byte;  FIO:string[10]  end;
Var
  mas : array[1..30] of Tmas.
```

ЗАДАНИЕ 2.

Написать функцию **find(st:string):word** которая находит номер второго пробела в строке **st** (если второго пробела нет строке, то функция возвращает 0).

ЗАДАНИЕ 3.

Пусть **Head** – указатель на головной элемент двусвязного линейного списка. Написать фрагмент программы через указатели (с описанием переменных) для преобразования данного списка в циклический двунаправленный список.



Критерии оценивания результатов студента на экзамене:

Оценка	Критерии оценивания
5	Студент полностью и правильно ответил на теоретические и практические вопросы билета. Студент владеет теоретическим материалом, отсутствуют ошибки при описании теории, формулирует собственные, самостоятельные, обоснованные, аргументированные суждения. Ответил на все дополнительные вопросы.
4	Студент ответил на теоретический вопрос билета с небольшими неточностями. Не выполнено одно из заданий практической части. Студент владеет теоретическим материалом, отсутствуют ошибки при описании теории. Ответил на большинство дополнительных вопросов.
3	Студент ответил на теоретический вопрос билета с существенными неточностями. Выполнено одно из заданий практической части. Студент владеет теоретическим материалом, присутствуют незначительные ошибки при описании теории. При ответах на дополнительные вопросы было допущено много неточностей.
2	При ответе на теоретический вопрос билета студент продемонстрировал недостаточный уровень знаний. При ответах на дополнительные вопросы было допущено множество неправильных ответов.

5.2.2. Перечень контрольных материалов для защиты курсового проекта / курсовой работы

Условием получения оценки по курсовой работе является не только подготовка разработка программы и отчета по курсовой работе, но и устная защита. К защите допускается готовые работы — окончательный вариант, исправленный на основании замечаний руководителя. Защита проводится как на практических занятиях, так и во внеурочное время. Автор работы выступает с докладом, в котором излагает основные итоги работы над проблемой, выводы и рекомендации. Продолжительность выступления — 7 минут. Защита проводится с использованием наглядного материала и демонстрацией разработанного приложения. Во время защиты необходимо ответить на заданные вопросы. Присутствие руководителя на защите курсовой работы студента обязательно.

Защита курсовой работы — это выступление студента перед экзаменационной комиссией, в ходе которого учащийся раскрывает тему исследования, обозначает основные моменты своей работы. Защита курсового проекта позволяет понять, насколько глубоко проработана тема исследования и насколько хорошо студент в ней разбирается.

Как правило, защита представляет собой демонстрацию презентации курсовой работы, синхронно сопровождающуюся докладом. Экзаменационная комиссия состоит из старшего преподавателя и научных руководителей защищающихся студентов.

Критерии оценивания выполнения курсовой работы

Оценка	Критерии оценивания		
	Знать	Уметь	Владеть
5	Студент знает теоретический материал, отсутствуют ошибки при описании теории и практической реализации, студент формулирует собственные, самостоятельные, обоснованные, аргументированные суждения, представляет полные и развернутые ответы на дополнительные вопросы.	Студент умеет самостоятельно разрабатывать алгоритмы для реализации поставленных задач на ЭВМ, обосновывать использование выбранных методов.	Курсовой проект выполнен полностью, студент владеет навыками самостоятельного использования компьютерной техники и среды программирования в своей профессиональной и учебной деятельности; методиками проверки правильности и точности получаемых численных решений
4	Студент знает теоретический материал, присутствуют незначительные ошибки при описании теории и алгоритмов, студент	Студент умеет самостоятельно разрабатывать алгоритмы для реализации поставленных задач на ЭВМ, но без обоснования	Курсовой проект выполнен полностью, студент владеет навыками самостоятельного использования

Оценка	Критерии оценивания		
	Знать	Уметь	Владеть
	формулирует собственные, самостоятельные, обоснованные, аргументированные суждения, допуская незначительные ошибки на дополнительные вопросы.	выбранных методов.	компьютерной техники и среды программирования в своей профессиональной и учебной деятельности; методиками проверки правильности и точности получаемых численных решений
3	Студент знает теоретический материал на минимально допустимом уровне, присутствуют незначительные ошибки при описании теории и практической реализации, испытывает затруднения в формулировке собственных обоснованных и аргументированных суждений, допуская незначительные ошибки на дополнительные вопросы.	Студент умеет разрабатывать алгоритмы с дополнительной помощью преподавателя для реализации поставленных задач на ЭВМ, но без обоснования выбранных методов.	Курсовой проект выполнен полностью, однако в нем присутствуют ряд недочетов, связанных с описанием методов и алгоритмов при решении задачи проектирования.
2	Студент практически не знает теоретический материал, допуская ошибки по сущности рассматриваемых (обсуждаемых) вопросов, испытывает затруднения в формулировке собственных обоснованных и аргументированных суждений, допускает ошибки при ответе на дополнительные вопросы.	Студент не умеет разрабатывать алгоритмы для реализации поставленных задач на ЭВМ.	Курсовой проект выполнен частично и содержит ряд существенных недочетов, студент не владеет навыками самостоятельного использования компьютерной техники и среды программирования в своей профессиональной и учебной деятельности; методиками проверки правильности и точности получаемых численных решений

5.3. Типовые контрольные задания (материалы) для текущего контроля в семестре

В лабораторном практикуме по дисциплине представлен перечень работ, обозначены цель и задачи, необходимые теоретические и методические указания к работе, перечень контрольных вопросов.

Защита лабораторных работ возможна после проверки правильности выполнения задания, оформления отчета. Защита проводится в форме собеседования преподавателя со студентом по теме работы. Примерный перечень контрольных вопросов для защиты практических работ представлен в таблице.

Тема лабораторной работы	Контрольные вопросы / задания (материалы)
1. Обработка числовых массивов.	1. Что такое массивы? Почему массив является структурированным типом данных? Что называют элементом массива? 2. Что такое индекс массива? Каким требованиям он должен удовлетворять? 3. Всегда ли индекс элемента массива совпадает с порядковым его номером? Приведите примеры. 4. Почему при описании массивов предпочтительнее употреблять константы, а не указывать размеры массива в явном виде? 5. Для чего в программах используют одномерные / двумерные массивы? 6. Общие и отличительные черты одномерных, двумерных и *n*-мерных массивов. 7. Могут ли существовать массивы массивов? Если могут, приведите пример такого описания и пример работы с такой структурой. 8. Верно ли, что одномерный массив соответствует понятию линейной таблицы (вектору)? 9. Сколько элементов может содержать массив? Сколько индексов может быть у одного элемента массива? Существуют ли ограничения на размерность массива? Может ли массив быть элементом массива? 10. Элементы какого типа может содержать массив? Какие типы данных допустимы для индексов элементов массива? 11. Может ли левая граница индексов массива быть меньше правой? Может ли левая граница индексов массива быть равной нулю / отрицательной? 12. Обязательно ли количество элементов массива должно быть фиксированным, то есть определяться при трансляции программы? Чем это объясняется? 13. Какие действия определены над массивом как единым объектом? 14. Как можно заполнить одномерный / двумерный массив? Приведите примеры. 15. Как можно сымитировать работу с массивом переменной длины? 16. Опишите порядок действий при вычислении суммы элементов массива. 17. Опишите порядок действий при определении максимального элемента в массиве. 18. Целесообразно ли использовать вложенные циклы, если совершается обход только главной диагонали квадратной матрицы? одной строки матрицы? одного столбца матрицы? 19. Опишите порядок действий при поиске элемента в неупорядоченном / упорядоченном массиве. 20. Опишите последовательность действий при создании алгоритма для поиска элемента в массиве методом деления пополам. 21. Какие способы объявления многомерных массивов вы знаете? Какая конструкция применяется для обработки *n*-мерного массива.

	22. Как обратиться к элементу многомерного массива? Можно ли выполнять обход двумерного массива, организовав внешний цикл по столбцам, а внутренний — по строкам? 23. Как располагаются в памяти ЭВМ элементы многомерных массивов? 24. Как подсчитать количество памяти, отведенное под массив? 25. В каких случаях допускается обращение к многомерному массиву целиком? 26. Какая конструкция применяется для обработки *n*-мерного массива. 27. Что называется сортировкой массива? Какие методы сортировки вы знаете, опишите их существенные отличия.
2. Подпрограммы. Рекурсивные алгоритмы.	1. Дайте определение термину «подпрограммы». Какие существуют виды подпрограмм? 2. Расскажите о методе последовательной детализации при разработке программ. Перечислите преимущества использования подпрограмм. 3. Что представляют собой вложенные подпрограммы? 4. В каких случаях целесообразно использовать именно функции? Какого типа может быть значение функции? 5. Чем синтаксически отличается описание процедуры от описания функции? 6. Как определить тип значения, возвращаемое функцией? Существует ли ограничения на тип возвращаемого функцией значения? Приведите соответствующие примеры. 7. В чем различие между стандартными и определенными пользователем подпрограммами? Приведите примеры. 8. Существуют ли подпрограммы без параметров? Приведите соответствующие примеры. Дайте определение термину «параметр». 9. Каким образом осуществляется обмен данными между основной программой и подпрограммой без параметров? Приведите соответствующие примеры. 10. Могут ли фактические параметры быть выражениями / именами переменных / именами других процедур или функций? Приведите соответствующие примеры. 11. Перечислите правила обращения к процедурам / функциям. 12. В чем разница между параметрами-переменными, параметрами-константами и параметрами-значениями? 13. Какое соответствие должно соблюдаться между формальными и фактическими параметрами? 14. Опишите последовательность событий при вызове процедуры или функции. 15. Какие переменные называются локальными? Чем глобальные переменные отличаются от локальных? Что произойдет при совпадении имен глобальных и локальных переменных? 16. Что такое время жизни переменной? Что такое область видимости переменной? 17. Поясните назначение служебных слов ***exit***, ***forward***. Приведите примеры их использования. 18. Может ли имя локальной переменной совпадать с именем глобальной? Можно ли утверждать, что одноименные глобальные и локальные переменные – это разные переменные?

	19. Может ли быть функция или процедура параметром подпрограммы? Приведите соответствующие примеры.
3. Обработка строкового типа.	1. Что представляет собой выражение строкового типа? К каким типам данных относятся строки? 2. Перечислите операции, определенные над данными строкового типа в языке Pascal, C++. Какие текстовые функции определены в Excel? 3. Как описываются переменные символьного типа? Какие операции применимы к символьным данным? Каково множество значений литерного типа? 4. Какой объем памяти требуется для хранения переменной символьного типа / строкового типа? Как хранится в памяти переменная типа string? 5. Как можно объявить величину строкового типа? Как можно обратиться к элементам строки? Какова максимально возможная длина строки? 6. Чем отличается массив символов (array of char) от строкового типа (string)? 7. Можно ли к данным символьного типа применять операции отношения? Как происходит сравнение данных строкового типа? 8. Какой объем памяти требуется для хранения переменной символьного типа? Какова максимально возможная длина строки? 9. С величиной какого типа данных совместим по присваиванию отдельный символ строки? Как осуществляется доступ к отдельному символу строки? 10. Верно ли то, что каждому символу ставится в соответствие целое число в диапазоне *0..255*? 11. Выделите основные стандартные процедуры и функции для работы со строками. 12. Может ли в процессе выполнения программы изменяться фактическая длина строки? Всегда ли фактическая длина строки равна объявленной в описании? 13. Может ли в процессе выполнения программы фактическая длина строки стать больше, чем объявлено в описании? Что произойдет в этом случае? Какова максимально возможная длина строки?
4. Применение типа «множество» к решению задач.	1. Что такое множество? Каким элементам должны удовлетворять все элементы множества? 2. Почему множество является структурированным типом данных? 3. Как хранится множество в памяти ЭВМ? Какой максимальный объем оперативной памяти может быть отведен под хранение одного множества? 4. Существуют ли ограничения по количеству элементов, входящих во множество? 5. Как вывести элементы множества? Как подсчитать количество элементов в множестве? 6. В чём сходство и различия множества и массива? Преимущества использования типа множество? 7. Приведите примеры задач, где использование множеств, упростит реализацию программы. 8. Какие операции над множествами существуют? Каков тип результатов выражений 9. с применением операций над множествами?

	10. В каком порядке выполняются операции в выражениях множественного типа? 11. Для чего применяются по отношению ко множествам операции ">=", "<="? В чем их отличие? 12. Для чего применяется операция in? Особенности ее применения. 13. Как записываются операции над множествами на языке Pascal? 14. Что такое конструктор множества и когда он применяется. 15. Как объявляются (декларируются) переменные типа множество? 16. Что такое пустое множество и как оно задаётся? 17. Какое значение у выражений: а) x in [x]; б) [ ] <= [x,y,z]; в) [x]<>[x,x,x]?
5. Комбинированный тип. Тип запись.	1. В каких случаях целесообразно использовать переменные типа запись? Как описываются переменные комбинированного типа? 2. К каким типам данных относится запись? Чем запись отличается она от массива? 3. Что такое поле записи? Какого типа могут быть поля записи? Могут ли совпадать имена полей у разных записей? Приведите соответствующие примеры. 4. Каков максимально допустимый уровень вложенности записей? 5. Какие операции определены над записями? Как осуществляется ввод-вывод записей? Всегда ли работа с записями сводится к работе с ее компонентами? Обос¬нуйте свой ответ. 6. Как осуществляется ссылка на компоненты записи? Что такое оператор присоединения? Каков его формат? 7. Как получить доступ к полю записи в массиве из записей? Как получить доступ к элементу массива, являющегося полем записи? 8. Что такое фиксированные записи? Вариантные записи? Когда применяется запись с вариантами? 9. Из каких частей состоит запись с вариантами? Приведите пример объявления и работу с переменной типа «запись с выбором». 10. Назначение и пример использования оператора With. 11. Как определить объем памяти под статическую запись / запись с вариантами? 12. Как заполнить в общем случае массив записей? Привести пример.
6. Обработка информации из файлов.	1. Дайте понятие файла, файловой переменной. 2. Перечислите виды файлов в языке Pascal/C. Как они описываются? Дайте понятие текстового режима доступа к файлу. 3. Назовите отличия файлового типа от типа массив. 4. Как определяется длина файла? Может ли файл не содержать ни одной записи? Если может, то как об этом узнать? 5. Прокомментируйте назначение процедур открытия и закрытия файлов. 6. Сравните текстовые и типизованные файлы по способу доступа к записям и по способу хранения данных. Приведите соответствующие примеры. 7. Как осуществляется чтение из текстовых и типизованных файлов? Приведите соответствующие примеры. 8. Как производится запись в текстовые и типизованные файлы? 9. Перечислите библиотечные процедуры и функции Pascal/C для работы с файлами. Приведите соответствующие примеры. 10. Какое максимальное количество файлов может быть одновременно открыто? Можно ли это количество увеличить?

	11. Поясните назначение стандартных текстовых файлов Input и Output. Приведите соответствующие примеры. 12. В каких случаях может наступить «неудачное открытие файла». Как обработать ошибки такого рода? 13. Как получить доступ к элементу файла с заданным номером? Приведите соответствующие примеры. 14. Поясните понятие «файл прямого доступа» и «файл последовательного доступа». 15. Можно ли какой-либо файл открыть одновременно для чтения и для записи? 16. Почему в текстовых файлах используется признак конца строки, а в типизированных – нет? 17. Напишите фрагмент программы для решения следующей подзадачи: «Дан текстовый файл. Подсчитать количество строк в нем». 18. Напишите фрагмент программы для решения следующей подзадачи: «Дан текстовый файл. Вывести третий символ второй строки». 19. Напишите фрагмент программы для решения следующей подзадачи: «Дан одномерный числовой массив. Записать все его элементы в текстовый файл». 20. Предложите несколько вариантов решения для удаления определенного элемента в каком-либо файле. 21. Предложите несколько вариантов программного решения для копирования содержимого файла в новый файл.
7. Динамические переменные, указатели. Обработка списков при помощи указателей.	1. Что является значением ссылочного типа? 2. В каких случаях используются динамические переменные и динамические структуры? 3. Какие операции определены над указателями? В каких трех состояниях может находиться указательная переменная? 4. Дайте расшифровку термина «куча». Какие параметры характеризуют текущее состояние «кучи». Как происходит их изменение при работе с динамическими переменными? 5. В каких случаях в Pascal возможно использование идентификатора до его описания? 6. Назначение константы *Nil*. Является ли значение *Nil* совместимо с любым ссылочным типом? 7. Могут ли использоваться операции "=" и "<>" для сравнения операндов типа указатель. В каком случае два указателя являются равными? 8. В чем отличие и особенности работы процедуры new, операции @ и функции *Ptr* при работе с переменной-указателем? 9. Какие функции используются для сбора информации об адресах различных объектов? Привидите примеры работы с ними. 10. Как подсчитать объем памяти, необходимый для хранения статических переменных. Приведите примеры для различных статических типов данных. Как создать и организовать работу с массивом, занимающим размер более 64Кб в Turbo Pascal 7.0? 11. Дайте определения всем основным динамическим структурам – списка, очереди, стека, дерева, кольца. Поясните отличия стека и очереди от простого списка. 12. Приведите примеры описания типа данных для работы со структурой: стек, очередь, дерево, кольцевой двунаправленный список.

	13. Дайте геометрическую интерпретацию динамическим структурам список, кольцо. Опишите графически как происходят изменения списка при добавлении или удалении элемента из него. 14. Опишите понятия «двоичное дерево», «степень дерева», «лист дерева», «идеально сбалансированное дерево». 15. Какие операции требуется выполнить для вставки в список его элемента? 16. Опишите отдельные шаги реализации алгоритма, как из входного списка получить упорядоченный список. 17. Опишите, как происходит работа со стеком при считывании и удалении значения из вершины стека. 18. Чем отличаются структуры односвязные линейные списки, односвязные циклические списки, двусвязные линейные списки, двусвязные циклические списки при работе с ними и их описании? 19. Можно ли для построения списка обойтись одной переменной? Сколько полей может содержать элемент списка? От чего зависит количество полей? Приведите примеры. 20. Сколько элементов может содержать список? Как заканчивается список? 21. Обязательно ли применять процедуру освобождения памяти, занятой элементом, когда мы избавляемся от этого элемента в списке? Каким образом это влияет на работу программы? 22. Можно ли ссылку одного элемента направить сразу на несколько других элемента? Как необходимо поменять тип указателя, чтобы решить эту проблему? 23. Может ли элемент списка быть такого типа, чтобы содержать несколько полей типа указателя? Если – да, то приведите пример, для чего это может быть нужно. 24. Можно ли последовательно "связать" два списка разного типа и почему? 25. Можно ли одновременно работать с несколькими списками сразу? 26. Как Вы считаете, на что нужно обращать особое внимание при работе со списками? 27. Опишите, как происходит работа при занесении нового элемента в очередь и его последующее извлечение из нее? 28. Перечислите и опишите способы обхода дерева и его вывода. 29. Опишите, как происходит занесение элемента в кольцо, извлечение элемента из кольца и обход кольца.
8. ООП при решении геометрических задач.	1. Объясните, в чем заключается преимущество и недостатки ООП. 2. Какие основные принципы лежат в основе объектно-ориентированного программирования? 3. Дайте определения терминам: класса, объекта, наследования, полиморфизма, виртуальных методов. Что в общем случае может содержать объект? 4. Чем отличаются скрытые и общедоступные поля и методы? Как изменить такое представление? 5. Верно ли, что каждый объект может иметь любое количество потомков и предков? 6. Приведите пример описания метода, описания объектного типа с методами и не содержащего методы. 7. Укажите программную конструкцию ООП, в которой используются следующие специальные слова: *constructor, virtual, public,*

	private, object, destructor. 8. Верно ли, что потомок автоматически наследует все поля и методы своего предка? 9. В каком случае необходим вызов конструктора? Верно ли, что конструктор обычно представляет собой метод, задающий для объекта некоторые начальные значения (т.е. выполняющий его инициализацию)? 10. Можно ли переопределять поля объекта-предка? Приведите пример. 11. Как регулируется область доступности (видимости) компонентов объекта? 12. Инициализация графического режима. Основные параметры. 13. Система координат экрана и управление графическим курсором. 14. Что такое графический примитив? Перечислите графические примитивы и параметры для их построения. Привести соответствующие примеры. 15. Подпрограммы для вывода отрезков; их параметры. Привести соответствующие примеры. 16. Построение прямоугольников и ломаных; окружностей, эллипсов, дуг. Привести соответствующие примеры. 17. Управление стилем линий. Привести соответствующие примеры. 18. Перечислите основные принципы, используемые при создании движущихся изображений. 19. Какими способами может быть выполнено стирание элемента изображения на экране?

Критерии оценивания лабораторной работы.

Оценка	Критерии оценивания
5 (отл.)	Работа выполнена полностью. Студент владеет теоретическим материалом, отсутствуют ошибки при описании теории, формулирует собственные, самостоятельные, обоснованные, аргументированные суждения, представляет полные и развернутые ответы на дополнительные вопросы.
4 (хор.)	Работа выполнена полностью. Студент владеет теоретическим материалом, отсутствуют ошибки при описании теории, формулирует собственные, самостоятельные, обоснованные, аргументированные суждения, допуская незначительные ошибки на дополнительные вопросы.
3 (удовл.)	Работа выполнена полностью. Студент владеет теоретическим материалом на минимально допустимом уровне, присутствуют незначительные ошибки при описании теории, испытывает затруднения в формулировке собственных обоснованных и аргументированных суждений, допуская незначительные ошибки на дополнительные вопросы.
2 (неудовл.)	Работа выполнена не полностью. Студент практически не владеет теоретическим материалом, допуская ошибки по сущности рассматриваемых (обсуждаемых) вопросов, испытывает затруднения в формулировке собственных обоснованных и аргументированных суждений, допускает ошибки при ответе на дополнительные вопросы.

5.4. Описание критериев оценивания компетенций и шкалы оценивания

При промежуточной аттестации в форме экзамена, дифференцированного зачета, дифференцированного зачета при защите курсового проекта/работы используется следующая шкала оценивания: 2 – неудовлетворительно, 3 – удовлетворительно, 4 – хорошо, 5 – отлично.

Критериями оценивания достижений показателей являются:

Наименование показателя оценивания результата обучения по дисциплине	Критерий оценивания
Знания	Знание терминов, классификаций, основных принципов
	Объем освоенного материала
	Полнота ответов на вопросы
	Четкость изложения и интерпретации знаний
Умения	Умение проводить анализ алгоритмов решения задач
	Умение разрабатывать программы на различных языках программирования
Навыки	Владеть навыками самостоятельной работы с учебной и научной литературой
	Владеет навыками обработки информации с использованием различных языков программирования

Оценка преподавателем выставляется интегрально с учётом всех показателей и критериев оценивания.

Оценка сформированности компетенций по показателю Знания.

Критерий	Уровень освоения и оценка			
	2	3	4	5
Знание терминов, классификаций, основных принципов	Не знает терминов классификаций, основных принципов.	Знает термины классификации, основные принципы, но допускает неточности формулировок.	Знает термины классификации, основные принципы.	Знает термины классификации, основные принципы, может корректно сформулировать их самостоятельно.
Объем освоенного материала	Не знает значительной части материала дисциплины.	Знает только основной материал дисциплины, не усвоил его деталей.	Знает материал дисциплины в достаточном объеме.	Обладает твердым и полным знанием материала дисциплины, владеет дополнительными знаниями.
Полнота ответов на вопросы	Не дает ответы на большинство вопросов.	Дает неполные ответы на все вопросы.	Дает ответы на вопросы, но не все – полные.	Дает полные, развернутые ответы на поставленные вопросы.
Четкость изложения и интерпретации знаний	Излагает знания без логической последовательности.	Излагает знания с нарушениями в логической последовательности.	Излагает знания без нарушений в логической последовательности.	Излагает знания в логической последовательности, самостоя-

				тельно их интерпретируя и анализируя.
	Не иллюстрирует изложение поясняющими схемами, рисунками и примерами	Выполняет поясняющие схемы и рисунки небрежно и с ошибками	Выполняет поясняющие рисунки и схемы корректно и понятно.	Выполняет поясняющие рисунки и схемы точно и аккуратно, раскрывая полноту усвоенных знаний.
	Неверно излагает и интерпретирует знания.	Допускает неточности в изложении и интерпретации знаний.	Грамотно и по существу излагает знания.	Грамотно и точно излагает знания, делает самостоятельные выводы.

Оценка сформированности компетенций по показателю Умения.

Критерий	Уровень освоения и оценка			
	2	3	4	5
Умение проводить анализ алгоритмов решения задач	Не умеет проводить анализ алгоритмов решения задач.	Умеет проводить анализ алгоритмов решения задач с подсказками преподавателя.	Умеет проводить анализ алгоритмов при решении типовых задач.	Умеет самостоятельно проводить анализ алгоритмов при решении нетиповых задач.
Умение разрабатывать программы на различных языках программирования	Не умеет разрабатывать программы на различных языках программирования.	Умеет разрабатывать простейшие программы на различных языках программирования, содержащие последовательные инструкций, условные переходы и циклы.	Умеет разрабатывать несложные программы на различных языках программирования, реализующие стандартные алгоритмы.	Умеет разрабатывать программы на различных языках программирования, реализующие алгоритмы повышенной сложности .

Оценка сформированности компетенций по показателю Навыки.

Критерий	Уровень освоения и оценка			
	2	3	4	5
Владеть навыками самостоятельной работы с учебной и научной литературой	Не использует учебную и научную литературу для подготовки к занятиям.	Имеются навыки самостоятельной работы с учебной и научной литературой, но недостаточные для полноценной подготовки.	Владеет навыками самостоятельной работы с учебной и научной литературой при подготовке к занятиям.	Использует учебную и научную литературу для самостоятельного приобретения новых знаний.
Владеет навыками обработки информации с использованием различных	В принципе не понимает, как обрабатывать информацию с использованием различных языков программирования.	Имеет лишь представление об обработки информации с использованием различных языков программирования.	Имеет представление об обработке информации лишь конкретного вида (текстовой или числовой и т.д.).	Владеет навыками обработки разнообразной информации с использованием различных языков

языков программирования				программирования.

6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ И УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ

6.1. Материально-техническое обеспечение

№	Наименование специальных помещений и помещений для самостоятельной работы	Оснащенность специальных помещений и помещений для самостоятельной работы
1	Специализированный компьютерный класс для проведения лабораторных занятий УК 4, № 229	15 персональных компьютеров, подключенных к сети «Интернет» и имеющих доступ в электронно-информационную образовательную среду, проектор, 10 комплектов оборудования для моделирования систем NI Elvis II
2	Учебная аудитория для проведения лекционных и практических занятий УК 4, № 323	Мультимедийный проектор, экран, ноутбук; специализированная мебель
3	Читальный зал библиотеки для самостоятельной работы	Компьютерная техника, подключенная к сети «Интернет» и имеющая доступ в электронно-информационную образовательную среду; специализированная мебель

6.2. Лицензионное и свободно распространяемое программное обеспечение

№	Перечень лицензионного программного обеспечения	Реквизиты подтверждающего документа
1	Microsoft Windows Professional 8.1	Соглашение Microsoft Open Value Subscription V9221014 от 2020-11-01 до 2023-10-31
2	Microsoft Office Professional Plus 2016	Соглашение Microsoft Open Value Subscription V6328633. Соглашение действительно с 02.10.2017 по 31.10.2023
3	Microsoft Windows 10 Корпоративная	Соглашение Microsoft Open Value Subscription V6328633. Соглашение действительно с 02.10.2017 по 31.10.2023). Договор поставки ПО 0326100004117000038-0003147-01 от 06.10.2017
4	Kaspersky Endpoint Security «Стандартный Russian Edition»	Сублицензионный договор № 102 от 24.05.2018. Срок действия лицензии до 19.08.2020 Гражданско-правовой Договор (Контракт) № 27782 «Поставка продления права пользования (лицензии) Kaspersky Endpoint Security от 03.06.2020. Срок действия лицензии 19.08.2022г.
5	Google Chrome	Свободно распространяемое ПО согласно условиям лицензионного соглашения
6	Mozilla Firefox	Свободно распространяемое ПО согласно условиям лицензионного соглашения Mozilla Public License 2.0 MPL
7	Система программирования PascalABC.NET	Свободно распространяемое ПО. Разрабатывается под свободной лицензией LGPLv3 как язык программирования для сферы образования и научных исследований.
8	Система адаптивного электронного тестирования	Персональный сайт кафедры. Доступ по ссылке http://aseo.tk-bstu.ru

6.3. Перечень учебных изданий и учебно-методических материалов

Печатные издания

1. Программирование, численные методы, оптимизация: учебное пособие / сост. А. В. Крюков, И. А. Рыбин, В. А. Порхало. — Белгород: Изд-во БГТУ, 2017. — 30 с.
2. Программирование и алгоритмизация : метод. указания к выполнению лаб. работ для студентов, обучающихся по направлениям 15.03.04 - Автоматизация технол. процессов и пр-в, 15.03.06 Мехатроника и робототехника, 27.03.04 Упр. в техн. системах. Ч. 1 / БГТУ им. В. Г. Шухова, каф. техн. кибернетики ; сост.: А. В. Крюков. — Белгород : Изд-во БГТУ им. В. Г. Шухова, 2014. — 160 с. : граф., табл.
3. Программирование и алгоритмизация : метод. указания к выполнению лаб. работ для студентов, обучающихся по направлениям 15.03.04 — Автоматизация технол. процессов и пр-в, 15.03.06 Мехатроника и робототехника, 27.03.04 Упр. в техн. системах. Ч. 2 / БГТУ им. В. Г. Шухова, каф. техн. кибернетики ; сост.: А. В. Крюков. — Белгород : Изд-во БГТУ им. В. Г. Шухова, 2014. — 111 с. : граф., табл.
4. Стативко Р.У. Языки программирования: учеб. пособие для студентов очной формы обучения / Р. У. Стативко, Е. А. Лазебная. — Белгород : Изд-во БГТУ им. В. Г. Шухова, 2015.
5. Чернова С.Б. Информатика. Программирование в среде PascalABC.NET: лаб. практикум: учеб. пособие для студентов всех направлений бакалавриата / С. Б. Чернова, Д. Н. Старченко. — Белгород: Изд-во БГТУ им. В. Г. Шухова, 2015.
6. Борисенко В.В. Основы программирования / В. В. Борисенко. - Москва: Интернет-университет информационных технологий, 2005. — 314 с.
7. Непейвода, Н. Н. Стили и методы программирования : учеб. пособие / Н. Н. Непейвода. — Москва: Интернет-Университет Информационных Технологий, 2005. — 315 с.

Электронные издания

1. Журавлева М.Г. Основы программирования. Введение в язык Си. Ч.1: учебное пособие по курсам «Программирование», «Основы алгоритмизации и программирования» / Журавлева М.Г., Алексеев В.А., Домашнев П.А.. — Липецк: Липецкий государственный технический университет, ЭБС АСВ, 2019. — 99 с. — ISBN 978-5-00175-001-7. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/101463.html.

2. Окулов С.М. Основы программирования / Окулов С.М.. — Москва: Лаборатория знаний, 2020. — 337 с. — ISBN 978-5-00101-759-2. — Текст: электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/6449.html.
3. Давыдова Н.А. Программирование: учебное пособие / Давыдова Н.А., Боровская Е.В.. — Москва: Лаборатория знаний, 2020. — 239 с. — ISBN 978-5-00101-788-2. — Текст: электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: http://www.iprbookshop.ru/6485.html.
4. Программирование и алгоритмизация [Электронный ресурс]: метод. указания к выполнению лаб. работ для студентов, обучающихся по направлениям 15.03.04 Автоматизация технол. процессов и пр-в, 15.03.06 Мехатроника и робототехника, 27.03.04 Упр. в техн. системах. Ч. 1 / БГТУ им. В. Г. Шухова, каф. техн. кибернетики ; сост.: А. В. Крюков. — Электрон. текстовые дан. — Белгород : Изд-во БГТУ им. В. Г. Шухова, 2014. Режим доступа: https://elib.bstu.ru/Reader/Book/2015021810360534600000659652.
5. Программирование и алгоритмизация [Электронный ресурс]: метод. указания к выполнению лаб. работ для студентов, обучающихся по направлениям 15.03.04 Автоматизация технол. процессов и пр-в, 15.03.06 Мехатроника и робототехника, 27.03.04 Упр. в техн. системах. Ч. 2 / БГТУ им. В. Г. Шухова, каф. техн. кибернетики ; сост.: А. В. Крюков. — Электрон. текстовые дан. — Белгород : Изд-во БГТУ им. В. Г. Шухова, 2014. Режим доступа: https://elib.bstu.ru/Reader/Book/2015021718090690900000657435.
6. Баженова И.Ю. Введение в программирование [Электронный ресурс]: учеб. пособие / И. Ю. Баженова, В. А. Сухомлин. — Электрон. текстовые дан. — Москва : Интернет-Университет Информационных Технологий : БИНОМ. Лаборатория знаний, 2007. Режим доступа: https://elib.bstu.ru/Reader/Book/9084.
7. Алексеев Е.Р. Free Pascal и Lazarus [Электронный ресурс]: учеб. по программированию / Е. Р. Алексеев, О. В. Чеснокова, Т. В. Кучер. — Электрон. текстовые дан. — Москва: ДМК Пресс, 2010. Режим доступа: https://elib.bstu.ru/Reader/Book/7255.

6.4. Перечень интернет ресурсов, профессиональных баз данных, информационно-справочных систем

1. Комплект Федеральных цифровых информационно-образовательных ресурсов (ФЦИОР) [Электронный ресурс]. – Режим доступа: http://www.fcior.edu.ru.
2. Информационно-коммуникационные технологии в образовании [Электронный ресурс]. – Режим доступа: http://www.ict.edu.ru.
3. Библиотека реализованных алгоритмов обработки информации [Электронный ресурс]. – Режим доступа: http://alglib.sources.ru.

4. Интернет-ресурс, посвященный алгоритмам машинного обучения [Электронный ресурс]. – Режим доступа: http://machinelearning.ru.

7. УТВЕРЖДЕНИЕ РАБОЧЕЙ ПРОГРАММЫ

Рабочая программа утверждена на 20_____ / 20_____ учебный год без изменений.

Протокол № _____ заседания кафедры от «____» _____ 20____ г.

Заведующий кафедрой _____ _____В. Г. Рубанов_____
подпись ФИО

Директор института _____ _____А. В. Белоусов_____
подпись ФИО