МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ИНЖЕНЕРНАЯ ШКОЛА

«СОГЛАСОВАНО»	«УТВЕРЖДАЮ»
Руководитель ОП «Прикладная механика»	Заведующий кафедрой Механики и математического моделирования (название кафедры)
(подпись) Озерова Г.П. (Ф.И.О. рук.ОП)	(подпись) Бочарова А.А.. (Ф.И.О. зав. каф.)
«25»июня 2016г.	«25» июня 2016г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ (РПУД)

ОСНОВЫ ПРОГРАММИРОВАНИЯ В КОМПЬЮТЕРНЫХ СИСТЕМАХ

Направление подготовки: 15.03.03 Прикладная механика

Профиль подготовки:

«Математическое и компьютерное моделирование механических систем и процессов»

Форма подготовки очная

курс 2 семестр 3,4
лекции 36 час.
практические занятия - час.
лабораторные работы – 72 час.
в том числе с использованием МАО лек.8 час. /пр. час. /лаб.32 час.
всего часов аудиторной нагрузки 126 час.
в том числе с использованием МАО 40 час.
самостоятельная работа 72 час.
в том числе на подготовку к экзамену 45 час.
контрольные работы -
курсовая работа / курсовой проект -
зачет – 3 семестр
экзамен 4 семестр

Рабочая программа составлена в соответствии с требованиями Федерального государственного образовательного стандарта высшего образования Дальневосточного федерального университета, принятого решением Ученого совета ДВФУ, протокол от 25.02.2016 № 02-16, введенного в действие приказом ректора ДВФУ от 10.03.2016 № 12-13-391

Рабочая программа обсуждена на заседании кафедры механики и математического моделирования, протокол № 9 от «23» июня 2016 г.

Заведующий кафедрой: к.ф.-м.н., доцент Бочарова А.А.
Составитель: к.т.н., доцент Озерова Г.П.

Оборотная сторона титульного листа РПУД

I. Рабочая программа пересмотрена на заседании кафедры:

Протокол от «23» _ июня 2017 г. № 11

Заведующий кафедрой _____ Бочарова А.А.
(подпись) (И.О. Фамилия)

В целях повышения качества знаний студентов в РПУД внесены следующие изменения:

- добавлены расчетно-графические работы (РГР)

II. Рабочая программа пересмотрена на заседании кафедры:

Протокол от «_____» _____ 20___ г. № _____

Заведующий кафедрой _____ _____
(подпись) (И.О. Фамилия)

Аннотация дисциплины
«Основы программирования в компьютерных системах»

Дисциплина «Основы программирования в компьютерных системах» разработана для студентов, обучающихся по направлению подготовки 15.03.03 «Прикладная механика», профиль «Математическое и компьютерное моделирование механических систем и процессов» и является дисциплиной выбора вариативной части Блока 1 «Дисциплины (модули)» учебного плана (Б1.В.ДВ.8.2).

Трудоемкость дисциплины составляет 180 часов (5 зачетных единиц). Учебным планом предусмотрены лекционные занятия (36 часа), лабораторные работы (72 часа) и самостоятельная работа студентов (72 часа из них 45 часов на экзамен). Дисциплина реализуется на 2 курсе в 3,4 семестре. Форма промежуточной аттестации – зачет (3 семестр), экзамен (4 семестр).

Дисциплина «Основы программирования в компьютерных системах» логически связана с дисциплинами «Программирование в инженерных задачах», «Вычислительная механика», «Программные системы инженерного анализа», «Инженерные web-технологии».

Цель дисциплины - формирование теоретических и практических навыков по разработке надежного, качественного программного обеспечения для инженерных расчетов с применением современных технологий программирования, методов и средств коллективной разработки.

Задачи дисциплины:

- Дать целостное представление о возможностях вычислительной техники, современном ее состоянии и тенденциях развития.
- Сформировать умение ставить задачу для решения ее на компьютере, а также реализовать ее средствами имеющейся вычислительной техники.
- Изучить основы структурного программирования, типы данных и конструкции языка высокого уровня

•Сформировать умение реализовывать инженерные вычислительные задачи средствами языка программирования.

•Дать методику, позволяющую свободно изучать и применять новые программные системы.

• Развить логическое и алгоритмическое мышление.

Для успешного изучения дисциплины «Основы программирования в компьютерных системах» у обучающихся должны быть сформированы следующие предварительные компетенции:

• владение навыками работы с различными источниками информации: книгами, учебниками, справочниками, Интернет;

• способность к алгоритмическому мышлению, сформированное в рамках предметов общеобразовательной школы «Алгебра», «Геометрия», «Информатика и ИТК»;

• владение навыками работы с вычислительной техникой;

• способность использовать простейшие текстовые и графические редакторы, офисные программы.

Планируемые результаты обучения по данной дисциплине (знания, умения, владения), соотнесенные с планируемыми результатами освоения образовательной программы, характеризуют этапы формирования следующих компетенций:

Код и формулировка компетенции	**Этапы формирования компетенции**	
ОПК-9 владением методами информационных технологий, соблюдением основных требований информационной безопасности, в том числе защиты государственной тайны	Знает	особенности современных методологий и технологий использования и создания программных средств
	Умеет	проектировать, конструировать и отлаживать программные продукты в соответствии с заданными критериями качества и стандартами
	Владеет	навыками работы с компьютером как средством управления информацией;
ОПК-10 способностью решать стандартные задачи профессиональной	Знает	основы событийного и объектно-ориентированного программирования
	Умеет	разрабатывать алгоритмы и

Код и формулировка компетенции	Этапы формирования компетенции	
деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности		программы для решения задач профессиональной деятельности средствами высокоуровневых языков программирования и визуальных систем программирования.
	Владеет	способностью использовать для решения аналитических, исследовательских и коммуникативных задач современные средства визуальной разработки программ.
ПК-7 умением извлекать актуальную научно-техническую информацию и наукометрическую информацию из электронных ресурсов, в том числе Science Direct, Elsiver Freedom Collection, SCOPUS	Знает	современные электронные научные базы данных для работы с научно-технической и наукометрической информацией при решении поставленных задач
	Умеет	извлекать актуальную научно-техническую информацию и наукометрическую информацию из электронных ресурсов, в том числе Science Direct, Elsiver Freedom Collection, SCOPUS
	Владеет	навыками обработки, анализа и интерпретации результатов исследований, а также подготовки данных для составления отчетов и презентаций, написания докладов, статей и другой научно-технической документации

Для формирования вышеуказанных компетенций в рамках дисциплины «Основы программирования в компьютерных системах» применяются следующие методы активного/ интерактивного обучения: лекция пресс-конференция; лекция «вдвоем»; деловая игра; «мозговой штурм»; групповая консультация.

I. СТРУКТУРА И СОДЕРЖАНИЕ ТЕОРЕТИЧЕСКОЙ ЧАСТИ КУРСА

Раздел I. Основы структурного программирования (12 час.)

Тема 1. Классификация языков программирования (4 час.)

Алгоритм и программа. Поколения языков программирования. Классификация языков программирования. Процедурное, объектно-ориентированное, функциональное и логическое программирование. Системы программирования. Языки программирования высокого уровня; Назначение языков программирования. Этапы разработки программы.

Тема 2. Основы программирования на языке высокого уровня (4 час.)

Синтаксис и семантика программы. Структура программы на языке Паскаль. Понятие синтаксиса и семантики. Представление данных в алгоритмических языках. Простейшие типы данных: INTEGER, REAL, BOOLEAN. Конструкции структурного программирования. Операторы присваивания, ввода, вывода, условные операторы. Этапы разработки программы: проектирование и реализация. Простейшие алгоритмы обработки данных: геометрические и алгебраические вычисления. Особенности обработки целых и вещественных чисел.

Тема 3. Операторы повторения (4 час.)

Назначение операторов цикла. Выделение повторяющихся действий. Различные типы циклов. Цикл со счетчиком. Алгоритмы нахождения минимальных и максимальных значений, сумм и произведений. Цикл с предусловием. Алгоритм разбиения числа на цифры. Цикл с постусловием. Алгоритмы нахождения наибольшего общего делителя и определения простоты числа. Алгоритмы решения вычислительных задач, использующих операторы повторения.

Раздел II. Структурированные типы данных (12 час.)

Тема 1. Массивы (4 час.)

Назначение и структура массивов. Описание одномерного массива. Типовые операции над массивами: ввод, вывод, заполнение случайным образом, простые переборные алгоритмы (поиск максимума/минимума, суммы, среднего арифметического и др.), вставка нового элемента в массив, удаление элемента из массива. Структура программ поиска и сортировки.

Поиск «со сторожем», двоичный поиск. Методы сортировки данных – алгоритмы «пузырька», обмена и вставки.

Описание многомерных массивов. Типовые операции над двумерными массивами: ввод, вывод, заполнение, переборные алгоритмы, удаление и вставка строк и столбцов. Арифметические операции над матрицами и векторами. Сложение матриц и векторов, умножение матриц, умножение матрицы на вектор. Скалярное произведение векторов

Тема 2. Типы данных, описывающие символьную информацию (4 час.)

Понятие символа. Множество символов. Упорядоченность символов. Типовые операции над символами: ввод, вывод, сравнение. Функции преобразования символов. Решение задач, использующих свойство упорядоченности символов: частота вхождения букв в текст, частота вхождения пар букв в текст.

Строка. Описание строк. Сравнение строк. Функции работы со строками. Примеры решения задач: разбиение строк на слова, удаление лишних пробелов, обработка текстов.

Тема 3. Типы данных, определяемые программистом (4 час.)

Описания новых типов данных. Перечисляемый тип. Описание и использование. Операции с перечисляемым типом. Тип диапазон. Описание и использование. Базовый тип. Ограничения на базовые типы данных. Тип множество. Описание и использование. Основные операции над множествами: объединение, пересечение, вычитание. Операции сравнения множеств. Функции, определенные для типа множества. Тип запись. Описание и использование. Обращение к полям. Тип файл. Операции с файловыми типами. Обработка текстовых документов. Примеры решения задач.

Раздел III. Основы модульного программирования (4 час.)

Тема 1. Процедуры и функции (4 час.)

Процедуры и функции, их сущность, назначение и различие. Синтаксис и семантика процедур и функций. Пошаговая разработка программ. Области видимости. Локальные и глобальные переменные. Типы параметров. Формальные и фактические параметры. Способы передачи параметров. Методология разработки программ «сверху вниз» и «снизу вверх».

Раздел IV. Графика и визуализация (8 час.)

Тема 1. Основы растровой графики(4 час.)

Работа в графическом режиме. Графическое окно. Рисование точек. Функции пера. Функции кисти. Функции шрифта. Графические примитивы. Функции канваса. Основы программного рисования. Построение графика функции.

Тема 2. Методы изображения и преобразования графических изображений (4 час.)

Рисование в относительных координатах. Масштабирование. Поворот. Реализация движение «стиранием». Реализация движения через дополнительный буфер. Алгоритмы рисования.

II. СТРУКТУРА И СОДЕРЖАНИЕ ПРАКТИЧЕСКОЙ ЧАСТИ КУРСА

Практические занятия –не предусмотрены учебным планом

Лабораторные работы (72 час.)

Лабораторная работа 1. Изучение PascalABC и BlackBoard.(4 час.)

Лабораторная работа 2. Арифметические выражения. (4 час.)

Лабораторная работа 3. Логические выражения. (4 час.)

Лабораторная работа 4. Операторы повторения. (4 час.)

Лабораторная работа 5. Стандартные алгоритмы: определение простоты числа, вычисление наибольшего общего делителя и пр. (4 час.)

Лабораторная работа 6. Одномерные массивы(4 час.)

Лабораторная работа 7. Алгоритмы сортировки и поиска(4 час.)

Лабораторная работа 8. Двумерные массивы.(4 час.)

Лабораторная работа 9. Операции над матрицами и векторами.(4час.)

Лабораторная работа 10. Символы и строки.(4 час.)

Лабораторная работа 11. Реализация псевдографического редактора.(4 час.)

Лабораторная работа 12. Процедуры и функции. (4час.)

Лабораторная работа 13. Рекурсия . (4час.)

Лабораторная работа 14. Типы данных, определяемые программистом. (4час.)

Лабораторная работа 15. Калькулятор комплексных чисел. (4 час.)

Лабораторная работа 16. Калькулятор смешанных дробей. (4 час.)

Лабораторная работа 17. Реализация программного рисования, масштабирование, движение изображений. (4 час.)

Лабораторная работа 18. Преобразование и движение изображений. (4 час.)

III. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

Учебно-методическое обеспечение самостоятельной работы обучающихся по дисциплине «Основы программирования в компьютерных системах» представлено в Приложении 1 и включает в себя:

– план-график выполнения самостоятельной работы по дисциплине, в том числе примерные нормы времени на выполнение по каждому заданию;

– характеристика заданий для самостоятельной работы обучающихся и методические рекомендации по их выполнению;

– требования к представлению и оформлению результатов самостоятельной работы;

– критерии оценки выполнения самостоятельной работы.

IV. КОНТРОЛЬ ДОСТИЖЕНИЯ ЦЕЛЕЙ КУРСА

№	Контролиру	Коды и этапы формирования компетенций	Оценочные средства

п/п	емые разделы / темы дисциплины			текущий контроль	промежуточная аттестация
1	Основы структурного программирования	ОПК-9, ОПК-10	Знает: особенности современных методологий и технологий использования и создания программных средств	Тест (ПР-1)	Вопросы 1-8
			Умеет: проектировать, конструировать и отлаживать программные продукты в соответствии с заданными критериями качества и стандартами	Расчетно-графическая работа (ПР-12) Лабораторная работа (ПР-6)	Задачи тип I
			Владеет: навыками работы с компьютером как средством управления информацией;	Расчетно-графическая работа (ПР-12)	Задачи тип II
2	Структурированные типы данных	ОПК-9,	Знает: особенности современных методологий и технологий использования и создания программных средств	Собеседование (УО-1)	Вопросы 9-35
			Умеет: проектировать, конструировать и отлаживать программные продукты в соответствии с заданными критериями качества и стандартами	Расчетно-графическая работа (ПР-12) Лабораторная работа (ПР-6)	Задачи тип I
			Владеет: навыками работы с компьютером как средством управления информацией;	Расчетно-графическая работа (ПР-12)	Задачи тип II

№ п/п	Контролируемые разделы / темы дисциплины	Коды и этапы формирования компетенций		Оценочные средства	
				текущий контроль	промежуточная аттестация
3	Основы модульного программирования	ОПК-10, ПК-7	Знает: основы событийного и объектно-ориентированного программирования	Собеседование (УО-1)	Вопросы 36-40
			Умеет: разрабатывать алгоритмы и программы для решения задач профессиональной деятельности средствами высокоуровневых языков программирования и визуальных систем программирования.	Лабораторная работа (ПР-6)	Задачи тип I
			Владеет: способностью использовать для решения аналитических, исследовательских и коммуникативных задач современные средства визуальной разработки программ.	Расчетно-графическая работа (ПР-12)	Задачи тип II
5	Графика и визуализация	ПК-7	Знает: современные электронные научные базы данных для работы с научно-технической и наукометрической информацией при решении поставленных задач	Коллоквиум (УО-2)	Вопросы 41-48
			Умеет: извлекать актуальную научно-техническую информацию и наукометрическую информацию из электронных ресурсов, в том числе Science Direct, Elsiver Freedom Collection, SCOPUS	Лабораторная работа (ПР-6)	Задачи тип I
			Владеет: навыками обработки, анализа и интерпретации результатов исследований, а также подготовки данных для составления отчетов и презентаций, написания докладов, статей и другой научно-технической документации	Расчетно-графическая работа (ПР-12)	Задачи тип II

Типовые контрольные задания, методические материалы, определяющие процедуры оценивания знаний, умений и навыков и (или) опыта

деятельности, а также критерии и показатели, необходимые для оценки знаний, умений, навыков и характеризующие этапы формирования компетенций в процессе освоения образовательной программы, представлены в Приложении 2.

V. СПИСОК УЧЕБНОЙ ЛИТЕРАТУРЫ И ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Основная литература

Трофимов В. В. Информатика. Учебник для вузов. – М: Юрайт. – 2010. –911 с http://lib.dvfu.ru:8080/lib/item?id=chamo:356824&theme=FEFU

2. Озерова, Г.П.Pascal в примерах и задачах [Электронный ресурс] : учебное пособие для вузов / Г. П. Озерова ; Дальневосточный федеральный университет, Инженерная школа. – Владивосток: Изд-во Дальневосточного федерального университета, 2014 , 123с. http://elib.dvfu.ru/vital/access/manager/Repository/fefu:2537

Канцедал С.А. Алгоритмизация и программирование: Учебное пособие / С.А. Канцедал. - М.: ИД ФОРУМ: НИЦ ИНФРА-М, 2013. - 352 с. http://znanium.com/go.php?id=391351

4. Немцова Т. И. Программирование на языке высокого уровня. Программирование на языке Object Pascal: Учебное пособие / Т.И. Немцова; Под ред. Л.Г. Гагариной. - М.: ФОРУМ: ИНФРА-М, 2015. - 496 с. http://znanium.com/bookread.php?book=472870

Дополнительная литература

1. Фаронов В. TurboPascal. Учебное пособие. – М:Кнорус , 2012. – 367 с. http://lib.dvfu.ru:8080/lib/item?id=chamo:664392&theme=FEFU (5 экз)

2. Парфилова Н. И. Программирование. Основы алгоритмизации и программирования : учебник для вузов / Н. И. Парфилова, А. Н. Пылькин, Б. Г. Трусов – Москва : Академия , 2014. - 240 с. http://lib.dvfu.ru:8080/lib/item?id=chamo:790355&theme=FEFU (5 экз)

3. Вирт Н. Алгоритмы и структуры данных. Новая версия для Оберона [Электронный ресурс] : учебное пособие. — Электрон. дан. — М. : ДМК Пресс, 2010. — 272 с. http://e.lanbook.com/books/element.php?pl1_id=1261

Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

1. Сайт свободно распространяемой системы программирования PascalABC. – Режим доступа: http://pascalabc.net/

2. Примеры программ на языке Паскаль. – Режим доступа: http://pascalabc.net/wiki/

3. Справочник по Паскалю. - http://tpdn.ru/

Перечень информационных технологий и программного обеспечения

Необходимое программное обеспечение: свободно распространяемая система программирования PascalABC (http://pascalabc.net/).

Материалы курса размещены в LMS BlackBoard, идентификатор: FU50219-15.03.03-OPvKS-01

VI. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

1. Описание последовательности действий студента («сценарий изучения дисциплины»).

Для успешного изучения дисциплины «Основы программирования в компьютерных системах» студенту необходимо:

1. При подготовке к лекции изучить конспект предыдущей лекции, а также при необходимости теоретический материал, представленный в системе BlackBoard. Далее ознакомиться с материалом следующей лекции (также через систему BlackBoard), вникнуть в суть изучаемой проблемы, подготовить вопросы.

2. На лекционном занятии тщательно конспектировать теоретический материал, участвовать в обсуждении, задавать вопросы.

3. При подготовке к лабораторным занятиям на основе материалов, представленных в системе BlackBoard, сначала понять задание лабораторной

работы, найти теоретический материал, необходимый для работы, изучить алгоритм реализации задания, сформулировать вопросы преподавателю.

4. На лабораторном занятии сначала задать вопросы преподавателю по методике выполнения работы, затем выполнить задание «по образцу». Отправить преподавателю через систему BlackBoard выполненное задание на проверку, ответить на вопросы преподавателя. Прежде, чем приступать к выполнению самостоятельных заданий, обдумать алгоритм их реализации, задать вопросы преподавателю по сути заданий, спланировать их выполнение.

5. Самостоятельные задания лабораторной работы можно выполнять как на аудиторном занятии, так и самостоятельно во внеаудиторное время. При этом результат их реализации необходимо проверять через систему BlackBoard и только после этого отправлять преподавателю на проверку.

6. В течение недели выбрать время для работы со специальной литературой в библиотеке и для занятий на компьютере.

7. Самостоятельную работу организовывать в соответствии с графиком выполнения самостоятельной работы, приведенном в приложении 1.

2. Рекомендации по использованию материалов учебно-методического комплекса. Рекомендуется использовать методические указания и материалы по курсу «Основы программирования в компьютерных системах», размещенные в системе BlackBoard, идентификатор курса FU50219-15.03.03-OPvKS-01.

3. Рекомендации по работе с литературой. Теоретический материал курса становится более понятным, когда дополнительно к прослушиванию лекций, использованию учебно-методического комплекса, представленного в системе BlackBoard, изучаются и книги из списка основной и дополнительной литературы. Литературу по курсу можно изучать в библиотеке, брать книгу на дом или читать ее на компьютере (если это электронный ресурс). Полезно использовать несколько учебников, однако желательно придерживаться рекомендация преподавателя по выбору книг по

каждому разделу. Не рекомендуется «заучивать» материал, желательно добиться понимания изучаемой темы дисциплины, а затем использовать изученный материал для реализации программ и/или сайтов. Кроме того, очень полезно выявить тенденции развития той или иной технологии разработки, выделить для себя направления дальнейшего изучения материала, для достижения более продвинутого уровня изучения дисциплины.

4. Рекомендации по подготовке к экзамену и зачету. Успешная подготовка к экзамену и зачету включает, с одной стороны, добросовестную работу в течение семестра, выполнение всех заданий преподавателя, а с другой – правильная организация процесса непосредственной подготовки. При подготовке к экзамену необходимо освоить теорию: разобрать определения всех понятий, повторить синтаксис и семантику изучаемых языков программирования, типовые алгоритмы решения задач. Затем рассмотреть примеры и самостоятельно реализовать задания из каждой темы. При этом, если задания формулируются студентом самостоятельно, – достигается более продвинутый уровень изучения дисциплины.

VII. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Аудиторные занятия по дисциплине включают лекции, практические занятия и лабораторные работы.

Для проведения лекционных и практических занятий необходима аудитория со следующим оборудованием:

– Акустическая система для потолочного монтажа с низким профилем, Extron SI 3CT LP (пара)

– Акустическая система для потолочного монтажа с низким профилем, Extron SI 3CT LP (пара)

– Акустическая система для потолочного монтажа с низким профилем, Extron SI 3CT LP (пара)

– Врезной интерфейс с системой автоматического втягивания кабелей TLS TAM 201 Standart III

– Документ-камера Avervision CP355AF

– ЖК-панель 47", Full HD, LG M4716CCBA

– Комплект удлинителей DVI по витой паре (передатчик/приёмник), Extron DVI 201 Tx/Rx

– Матричный коммутатор DVI 4x4. Extron DXP 44 DVI PRO

– Микрофонная петличная радиосистема УВЧ диапазона Sennheiser EW 122 G3 в составе рэкового приёмника EM 100 G3, передатчика SK 100 G3, петличного микрофон ME 4 с ветрозащитой и антенн (2 шт.)

– Мультимедийный проектор, Mitsubishi EW330U, 3000 ANSI Lumen, 1280x800

– Расширение для контроллера управления Extron IPL T CR48

– Сетевая видеокамера Multipix MP-HD718

– Сетевой контроллер управления Extron IPL T S4

– Стойка металлическая для ЖК-дисплея У SMS Flatscreen FH T1450

– Усилитель мощности, Extron XPA 2001-100V

– Усилитель-распределитель DVI сигнала, Extron DVI DA2

– Цифровой аудиопроцессор, Extron DMP 44 LC

– Шкаф настенный 19" 7U, Abacom VSP-W960SG60

– Экран проекционный ScreenLine Trim White Ice, 50 см черная кайма сверху, размер рабочей области 236x147 см

Лабораторные работы проводятся в компьютерном классе, в котором

– Моноблок Lenovo C306G-i34164G500UDK (20 шт),

– Акустическая система для потолочного монтажа с низким профилем, Extron SI 3CT LP (пара)

– Акустическая система для потолочного монтажа с низким профилем, Extron SI 3CT LP (пара)

– Акустическая система для потолочного монтажа с низким профилем, Extron SI 3CT LP (пара)

– Врезной интерфейс с системой автоматического втягивания кабелей TLS TAM 201 Standart III

– Документ-камера Avervision CP355AF

– ЖК-панель 47", Full HD, LG M4716CCBA

– Комплект удлинителей DVI по витой паре (передатчик/приёмник), Extron DVI 201 Tx/Rx

– Матричный коммутатор DVI 4x4. Extron DXP 44 DVI PRO

– Микрофонная петличная радиосистема УВЧ диапазона Sennheiser EW 122 G3 в составе рэкового приёмника EM 100 G3, передатчика SK 100 G3, петличного микрофон ME 4 с ветрозащитой и антенн (2 шт.)

– Мультимедийный проектор, Mitsubishi EW330U, 3000 ANSI Lumen, 1280x800

– Расширение для контроллера управления Extron IPL T CR48

– Сетевая видеокамера Multipix MP-HD718

– Сетевой контроллер управления Extron IPL T S4

– Стойка металлическая для ЖК-дисплея У SMS Flatscreen FH T1450

– Усилитель мощности, Extron XPA 2001-100V

– Цифровой аудиопроцессор, Extron DMP 44 LC

– Шкаф настенный 19" 7U, Abacom VSP-W960SG60

– Экран проекционный ScreenLine Trim White Ice, 50 см черная кайма сверху, размер рабочей области 236x147 см

Для проведения лабораторных работ на компьютерах должно быть установлено следующее программное обеспечение:

- операционная система Windows;

-браузер Chrome;

- свободно-распространяемое программное обеспечение ABSPascal.

–

Приложение 1

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ИНЖЕНЕРНАЯ ШКОЛА

УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

по дисциплине «Основы программирования в компьютерных системах»

Направление подготовки: 15.03.03 Прикладная механика

Профиль подготовки: «Математическое и компьютерное моделирование механических систем и процессов»

Форма подготовки очная

Владивосток
2016

План-график выполнения самостоятельной работы по дисциплине

№ п/п	Дата/сроки выполнения	Вид самостоятельной работы	Примерные нормы времени на выполнение	Форма контроля
1	1-5 недели семестра	Решение задач по теме «Выражения и ветвления»	3 час.	ПР-12
2	6-10 недели семестра	Решение задач по теме «Операторы повторения»	3 час.	ПР-12
3	11-12 недели семестра	Подготовка к тесту по разделу «Основы структурного программирования»	3 час.	ПР-1
4	17 -18 недели семестра	Решение задач по теме «Одномерные массивы»	3 час.	ПР-12
5	Экзаменационная сессия	Подготовка к зачету за 3 семестр	6 час.	зачет
8	8-9 недели семестра	Подготовка к устному опросу по разделу «Структурированные типы данных»	3 час.	УО-1
10	12-13 недели семестра	Подготовка к устному опросу по разделу «Основы модульного программирования»	3 час.	УО-1
12	18 неделя семестра	Подготовка к устному опросу по разделу «Графика и визуализация»	3 час.	УО-2
13	Экзаменационная сессия	Подготовка к экзамену, второй семестр	45 часов	Экзамен
		Итого	**72 час.**	

Характеристика заданий для самостоятельной работы обучающихся и методические рекомендации по их выполнению

Расчетно-графические задания

Каждая работа состоит из нескольких заданий, в каждом задании требуется решить задачу определенного типа. Варианты заданий приведены в приложении 2. Все задачи реализуются на языке Паскаль в системе PascalABC. Перед сдачей преподавателю задания, его необходимо проверить с помощью специального инструмента, реализованного в виде web-приложения и размещенного в курсе «Основы программирования в компьютерных системах» LMS BlackBoard. Далее приводится образцы решения каждого расчетно-графического задания.

Расчетно-графическая работа по разделу «Основы структурного программирования», тема «Выражение и ветвление», базовый уровень

Задача 1. Дан треугольник, заданный координатами трех его вершин. Определить, лежат ли хотя бы две его вершины в одной четверти координатной плоскости.

Решение:

```
VAR
  x1, y1, x2, y2, x3, y3, r1, r2, r3 : REAL;
 BEGIN
  WRITELN('введите координаты трех точек');
  READLN (x1, y1, x2, y2, x3, y3);
 { проверить, может ли треугольник быть задан этими точками, по
правилу треугольника}
  r1 :=sqrt( sqr(x1 - x2) + sqr(y1 - y2));
  r2:=sqrt( sqr(x2 - x3) + sqr(y2 - y3));
  r3:=sqrt( sqr(x3 - x1) + sqr(y3 - y1));
  IF (r1 >= r2 + r3) OR (r2 >= r1 + r3) OR (r3 >= r2 + r1)
  THEN
    WRITELN('точки не могут быть вершинами треугольника')
  ELSE
  BEGIN
    { две точки лежат в одной четверти, если произведение их
     первых и вторых координат положительно }
    IF ((x1*x2 >= 0) AND (y1*y2 >= 0)) OR
    ((x1*x3 >= 0) AND (y1*y3 >= 0)) OR
    ((x3*x2 >= 0) AND (y3*y2 >= 0))
    THEN
      WRITELN('две вершины лежат в одной четверти')
    ELSE
      WRITELN('все вершины лежат в разных плоскостях')
    END
 END.
```

Задача 2. Определить, равна ли сумма двух первых цифр заданного четырехзначного числа сумме двух его последних цифр.

Решение:

```
VAR
  num, a1, a2, a3, a4: INTEGER;
 BEGIN
  WRITELN('введите четырехзначное число');
  READLN(num);
  IF (num < 1000) OR (num > 9999) THEN
    WRITELN('введено неверное значение')
  ELSE
  BEGIN
    a1 := num MOD 10;  {выделение последней цифры}
    num := num DIV 10; {уменьшение разряда числа}
    a2 := num MOD 10;
    num := num DIV 10;
```

```
    a3 := num MOD 10;
    num := num DIV 10;
    a4 := num MOD 10;
    IF (a1 + a2)=(a3 + a4) THEN  WRITELN('сумма цифр равна')
    ELSE WRITELN('сумма цифр не равна');
  END
END.
```

Задача 3. Выяснить, являются ли поля (k,l) и (m,n) шахматной доски полями одного цвета.

Решение:

```
VAR
  k, l, m, n : INTEGER;
 BEGIN
  WRITELN('введите координаты двух шахматных полей');
  READ(k, l ,m, n);
  IF (k<1) OR (k>8) OR (l<1) OR (l>8) OR (m<1)
     OR (m>8) OR (n<1) OR (n>8)
  THEN WRITE ('введены неправильные координаты')
  ELSE
   { поля имеют одинаковый цвет, если сумма их координат
    является для обоих полей либо четной, либо нечетной }
    IF  ODD(k+l) = ODD(n+m)
    THEN WRITELN('поля имеют одинаковый цвет')
    ELSE WRITELN('поля имеют разный цвет')
 END.
```

Расчетно-графическая работа по разделу «Основы структурного программирования», тема «Операторы повторения», продвинутый уровень

Задача 1. Дана последовательность из **n** элементов. Найти максимальный элемент последовательности и его номер.

Решение:

```
VAR
  num, max: REAL;
  n,i, nMax: INTEGER;
BEGIN
  WRITE('n='); READ(n);
  IF n>0 THEN
  BEGIN
    WRITE('num=');READ(num);
    max:=num; {считаем максимальным первое введенное число}
    nMax:= 1; {запоминаем его номер}
    FOR i:= 2 TO n DO
    BEGIN
      WRITE('num=');READ(num);
      IF num > max THEN {найдено число, больше максимального}
```

```
      BEGIN
       {запоминаем новый максимум и его номер}
        max:= num;
        nMax:= i;
      END;
    END;
    WRITELN('max=', max, ' nMax=', nMax);
  END
  ELSE WRITELN('ошибочный ввод');
 END.
```

Задача 2. Вычислить сумму ряда для заданного числа членов **n**:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots + (-1)^n \frac{x^{2n+1}}{(2n+1)!}$$

Решение:

```
VAR
  n, i:INTEGER;
  sum, r, x: REAL;
BEGIN
  WRITE('n='); READ(n);
  WRITE('x='); READ(x);
  IF n > 0 THEN
  BEGIN
    r:= x;
    sum:= r;
    FOR i:=1 TO n-1 DO
    BEGIN
     { используем предыдущее значение r }
       r:=r*(-1)*sqr(x)/(2*i*(2*i+1));
       sum:=sum + r;
    END;
    WRITELN('sum=',sum, 'sin(x)= ',sin(x));
  END
  ELSE WRITE('ошибочный ввод');
END.
```

Задача 3. Подобрать всевозможные варианты размена произвольной >7 руб. суммы с помощью 3-х и 5-ти рублевых купюр.

Решение:

```
VAR
  x, y, sum :INTEGER;
BEGIN
  WRITE('sum='); READ(sum);
  IF sum > 7 THEN
  BEGIN
    FOR x :=0 TO sum div 3 DO
      FOR y :=0 TO sum div 5 DO
        IF x*3 + y*5 = sum
        THEN WRITELN('3*',x, '+5*',y, '=',sum);
```

```
  END
  ELSE WRITE('ошибочный ввод');
 END.
```

Расчетно-графическая работа по разделу «Структурированные типы данных», тема «Одномерные массивы», базовый уровень

Задача 1. Найти максимальный элемент в массиве A и его место в массиве.

Решение:

```
const
  n1=20;
var
  a:array [1..n1] of real;
  n, i, k:integer;
  max:real;
begin
  write('n='); readln(n);
  if (n<=0) or (n>n1) then writeln('error ', n)
  else
    begin
      for i:=1 to n do
        begin
          write('a[',i, ']= '); readln(a[i]);
        end;
    {предполагаем, что первый элемент - максимальный}
      max:= a[1]; k:=1;
      for i:=2 to n do
        if max<a[i] then
         begin
          {запоминаем новый максимум и его номер}
          max:=a[i]; k:=i;
         end;
      writeln('max=',max, ' k=',k);
    end;
  readln;
end.
```

Задача 2. Удалить из массива **A** элемент с номером **k**.

Решение:

```
const
  n1=20;
var
  a:array [1..n1] of real;
  n, i, k, j:integer;
begin
  write('n='); readln(n);
  {проверку n выполните самостоятельно}
```

```
  for i:=1 to n do
    begin
      write('a[',i, ']= '); readln(a[i]);
    end;
  write('k='); readln(k);
  if (k<=0) or (k>n) then writeln('error ',k)
  else
    begin
  {сдвигаем элементы массива на одну позицию влево,
    начиная с k-го элемента}
      for j:=k to n-1 do
        a[j]:=a[j+1];
      n:=n-1;
     {вывод элементов массива}
      writeln('new A');
      for i:=1 to n do
        write(a[i],' ');
    end;
  readln;
end.
```

Задача 3. Определить количество различных элементов массива.

Решение:

```
const n1=20;
var
k, j, i: integer;
a, b: array [1..n1] of integer;
begin
  write('n= '); readln(n);
  for i:=1 to n do
    begin
      write('a[',i, ']= '); readln(a[i]);
    end;

  k:=0;
  for i:=1 to n do
    begin
    {заносим очередной элемент в конец массива B}
      b[k+1]:=a[i];
      j:=1;
      while b[j]<>a[i] do j:=j+1;
      if j=(k+1) {такого элемента в массиве не было}
       then k:=k+1;{сдвигаем указатель конца массива B}
    end;

  writeln('count= ',m);
  write('different ');
  for j:=1 to k do
    write(a[j], ' ');
    end;
  readln;
end.
```

Расчетно-графическая работа по разделу «Структурированные типы данных», тема «Двумерные массивы», базовый уровень

Задача 1. Заполнить массив размером n на n следующим образом:

1	0	1	0	1	0
0	1	0	1	0	1
1	0	1	0	1	0
0	1	0	1	0	1
1	0	1	0	1	0

Решение:

```
CONST
  n1=15;  {количество строк и столбцов}
VAR
  a: ARRAY [1..n1,1..n1] OF INTEGER;
  n,i,j:Integer;
Begin
  Write ('n='); Readln(n);
  If (n<1) or (n>n1) then Write ('Неправильный размер массива')
  Else
  Begin
  {заполнение массива}
    FOR i:=1 TO n DO
      FOR j:=1 TO n DO
        IF ODD(i+j) THEN a[i,j]:=0  ELSE a[i,j]:=1;
  {вывод массива}
     writeln('array A');
     for i:=1 to n do
     begin
       for j:=1 to n do
         write(a[i,j], ' ');
       writeln;
     end;
  end;
end.
```

Задача 2. Вывести все элементы, которые больше чем сумма остальных в строке.

Решение:

```
const
  n1=20;
Var
  a:array[1..n1,1..n1] of real;
  sum:array[1..n1] of real;{массив для хранения суммы каждой
строки}
  n,i,j:Integer;
  fl: boolean; {признак того, есть ли искомые документы}

Begin
  Write ('n='); Readln(n);
```

```
  If (n<1) or (n>n1) then Write ('Неправильный размер массива')
  Else
  Begin
  {ввод массива}
    For i:=1 to n do
     for j:=1 to n do
     Begin
       Write ('a[',i,',',j,']=');
       Readln (a[i,j]);
     end;
  {вывод массива}
     writeln('array A');
     for i:=1 to n do
     begin
       for j:=1 to n do
         write(a[i,j], ' ');
       writeln;
     end;
  {вычисляем сумму каждой строки и заносим значение в
   соответствующий элемент массива Sum}
    For j:=1 to n do
    begin
      sum[j]:=0;
      for i:=1 to n do
        sum[j]:=sum[j]+a[i,j];
    end;
  {предполагаем, элементов, больших суммы остальных в строке
нет}
    fl:= false;
    for i:=1 to n do
      for j:=1 to n do
        if a[i,j]>(sum[j]-a[i,j]) then
          begin
            writeln('a[',i,',',j,']=',a[i,j]);
            fl:= true; {такие элементы есть}
          end;
     if not fl
     then writeln('элементов, больших остальных в своей строке,
нет');
  end;
  readln;
end.
```

Задача 3. Найти номера строк квадратной целочисленной матрицы, элементы в каждой из которых одинаковы.

Решение:

```
const
  n1=20; m1=20;
Var
  a:array[1..n1,1..n1] of real;
  n, i, j:Integer;
  p: real;
```

```
  fl, {признак того, что в текущей строке все элементы
одинаковы}
  fl1: boolean; {признак того, что есть строки с одинаковыми
                 элементами}
Begin
  Write ('n='); Readln(n);
  If (n<1) or (n>n1)  then Write ('Неправильный размер массива')
    Else
    Begin
      For i:=1 to n do
        For j:=1 to n do
        Begin
          Write ('a[',i,',',j,']=');
          Readln (a[i,j])
        end;
    writeln('массив A');
    For i:=1 to n do
    Begin
      For j:=1 to n do
        Write(a[i,j] , ' ');
      Writeln;
     end;
     fl1:=false;{предполагаем, что строк с одинаковыми элементами
нет}
     For i:=1 to n do
     begin
       fl:=true;{предполагаем, что в i-ой строке все элементы
                 одинаковы}
       for j:=2 to n do
         if a[i,1]<>a[i,j]
         then fl:=false; {нашелся неодинаковый элемент}
       if fl then
       begin
         writeln('в строке ', i, ' все элементы одинаковы');
         fl1:=true; { строки с одинаковыми элементами есть}
       end;
    end;
    if not fl1 then writeln('Строк с одинаковыми элементами
нет');
  end;
  readln;
end.
```

Расчетно-графическая работа по разделу «Структурированные типы данных», тема «Типы данных, определяемые программистом», продвинутый уровень

Задача 1. Дана строка текста, состоящая из слов, разделенных пробелом. Получить новую строку с обратным порядком слов, например “я пишу программу” – “программу пишу я”.

Решение:

```
VAR
  StrWordO, StrWordN: STRING;
  Word: STRING;
  IndBlank: INTEGER;
BEGIN
  StrWordN:= ''; {пока строка пуста}
  READLN(StrWordO);
  StrWordO:=CONCAT(StrWordO, ' '); {добавляем пробел в конце
строки, чтобы можно было выделить последнее слово}
{будем последовательно удалять слова из исходной строки до тех
пор, пока она не станет пустой}
  WHILE StrWordO<>'' DO
  BEGIN
    IndBlank:=POS(' ', StrWordO);  {определяем позицию первого в
строке пробела, то есть конца очередного слова}
    Word:= COPY(StrWordO, 1, IndBlank); {заносим слово от начала
                            строки вместе с пробелом в
переменную}
    DELETE(StrWordO,1, IndBlank); {удаляем слово из исходной
                                   строки}
    StrWordN:=Word + StrWordN; {вставляем выделенное слово в
                                 начало новой строки}
  END;
  WRITELN(StrWordN);
END.
```

Задача 2. Дана строка, состоящая из слов, разделенных пробелами. Вывести все слова строки в алфавитном порядке .

Решение:

```
CONST
N=40;
VAR
  S, w: STRING;
  sArr: array [1..n] of string;
  k, m, i, j: INTEGER;
BEGIN
  READLN(S);
  S:=s+' '; {добавляем пробел в конце строки}
  m:=0;
{разбиваем строку на слова и заносим слова в массив sArr}
  WHILE S<>'' DO
  BEGIN
    k:=POS(' ', S);
    w:= COPY(S, 1, k-1);
    m:=m+1;
    sArr[m]:=w;
    DELETE(S,1, k)
  END;
{сортируем массив строк}
  For i:=m-1 downto 1 do
    For j:=1 to i do
```

```
      If  sArr[j]>sArr[j+1] then
      Begin
        W:=sArr[j+1]; sArr[j+1]:=sArr[j]; sArr[j]:=w;
      End;

  For i:=1 to m do
    Writeln(sArr[i]);
END.
```

Задача 3. Дан список студентов группы и оценки каждого студента за одну сессию. Подсчитать количество отличников, хорошистов, двоечников и троечников в группе.

Решение:

```
CONST
  n=20;
  m=4;
TYPE
  TMark= ARRAY [1..m] OF 2..5;
  TStudent=RECORD
    fio: string [40];
    grup: string[6];
    val: TMark
  END;
VAR
  Stud: ARRAY [1..n]  OF TStudent;
  Gr_val : array [2..5] OF INTEGER;
  i, j, min: INTEGER;

FUNCTION min_v(a:Tmark):INTEGER;
  VAR k: integer;
      min : integer;
  BEGIN
    min := a[1];
    FOR k:=2 TO m DO
      IF a[k]<min THEN min:=a[k];
    min_v := min;
  END; {min_v}

BEGIN
  WRITELN(' введите информацию о студенте');
  FOR i:=1 TO n DO
  BEGIN
    WRITE(' Группа: '); READLN(Stud[i].grup);
    WRITE(' Фамилия: '); READLN(Stud[i].fio);
    FOR j:=1 TO m DO
    BEGIN
      WRITE( j, '-ая  оценка'  ); READLN(Stud[i].val[j]);
    END;
  END;
```

```
{находим минимальную оценку у каждого студента и увеличиваем
соответствующий элемент массива групп успеваемости}
  FOR i:=1 TO n DO
  BEGIN
    min:=min_v(stud[i].val);
    gr_val[min]:=gr_val[min]+1;
  END;
  WRITELN(' Отличников ', gr_val[5]);
  WRITELN(' Хорошистов ', gr_val[4]);
  WRITELN(' Троечников  ', gr_val[3]);
  WRITELN(' Двоечников  ', gr_val[2]);
END.
```

Задача 4. Дан текстовый файл stud.txt, в который занесена информация о студенте в следующем виде:

Решение:

Фамилия И.О.	Номер группы	оц1	оц2	оц3	оц4
30	6	2	2	2	2
Иванов И.П.	Е-841	5	4	3	4
Петров С.К.	У-641	4	3	3	2
Сидоров Р.Ш.	Е-841	5	5	5	5
Панов Е.Р.	У-841	4	2	2	3

В новый файл занести информацию о студентах заданной группы.

```
CONST
  m=4;
TYPE
  TMark= ARRAY [1..m] OF 2..5;
  TStudent=RECORD
    fio: string [30];
    grup: string[6];
    val: TMark
  END;
VAR
  f,f1: text;
  student: TStudent;
  num_gr: string[6];
BEGIN
  ASSIGN(F,'stud.txt'); {этот оператор связывает файловую
переменную f с файлом  stud.txt}
  ASSIGN(F1,'gr.txt');
  RESET(f); {открывает файл stud.txt для чтения и устанавливает
указатель файловой переменной  f на первую строку файла}
  REWRITE(f1); { создает пустой файл и устанавливает указатель
файловой переменной f1 на начало файла }
  write('Введите название группы'); readln(num_gr);
  WHILE not EOF(f) DO {пока не конец исходного файла}
  WITH Student DO
  BEGIN
    {читаем строку из исходного файла}
    readln (f, fio, grup, val[1], val[2], val[3], val[4]);
```

```
    { В результате указатель файловой переменной перемещается на
следующую строку в файле. Если указатель был установлен на
последнюю строку, то после ее чтения, указатель устанавливается
на специальный символ, обозначающий конец файла. }
    IF grup = num_gr
    THEN writeln (f1, fio, grup, ' ', val[1],' ', val[2],' ',
val[3],' ', val[4]);
    { В выходной файл заносится новая строка, в то место, куда
установлен указатель. После вывода очередной строки, указатель
перемещается на пустую (несуществующую) строку, куда будет
осуществлен следующий вывод. }
  END;
  CLOSE(f);
  CLOSE(f1); {оба файла закрываются}
END.
```

Расчетно-графическая работа по разделу «Основы модульного программирования»

Задача 1. Вывести все простые числа, меньшие N.

Решение:

```
VAR
  i, n : integer;
{ Опишем функцию для определения, является ли число x,
передаваемое в качестве параметра, простым или составным. Если
число простое, функция возвратит значение true, в противном
случае (число является составным, false}
FUNCTION Simple(x: integer):boolean;
  VAR count : integer;
  BEGIN
    count:=1;
 { простое число делится только на себя и на 1}
    REPEAT
      count:=count+1
    UNTIL (x MOD count)=0;
    IF x = count THEN Simple := true
    else Simple := false;
  END;
BEGIN
  write('N=');
  readln(n);
{ Перебираем все числа от 2 до n-1}
  FOR i:=2 TO n-1 DO
{ если число простое (значение функции true), то печатаем его}
    IF Simple(i) THEN write(i,' ');
END.
```

Задача 2. Посчитать периметр N- угольника, заданного координатами своих вершин.

Решение:

```
CONST
  n1=10;
VAR
  x, y : array [1..n1] of real;
  i, n : integer;
  sumLength : real;
{ функция вычисляет расстояние между точками (x1,y1) и (x2,y2)}
  FUNCTION Len(x1,y1,x2,y2 : real):real;
  BEGIN
    len:=sqrt(sqr(x2-x1)+sqr(y2-y1));
  END;

BEGIN
  write('n=');readln(n);
  FOR i:=1 TO n DO
  BEGIN
    write('x[',i,']=');
    readln(x[i]);
    write('y[',i,']=');
    readln(y[i]);
  END;
{ вычисляем длину стороны, образованную 1 и последней вершиной и
заносим ее в переменную, для вычисления периметра}
  SumLength := Len (x[1], y[1], x[n], y[n]);
  FOR i:= 1 TO n-1 do
{ вычисляем длину стороны, образованную i и i+1 вершиной и
добавляем ее в переменную, для вычисления периметра}
    SumLength := SumLength + Len(x[i], y[i], x[i+1], y[i+1]);
  writeln(' SumLength=', SumLength);
END.
```

Расчетно-графическая работа по разделу «Графика и визуализация»

Задача 1. Нарисовать фигуру «кораблик» и реализовать ее движение по горизонтали, так, чтобы фигура не выходила за границу окна.

Решение:

```
uses GraphABC;
var
  i, x, y, h, kx, ky: integer;
  procedure drawShip(x,y: integer; kx,ky: real);
  begin
    { корпус корабля}
    setPenColor(clBlack);
    MoveTo(round(x+1*kx),round(y +18*ky));
    LineTo(round(x+21*kx),round(y+14*ky));
    LineTo(round(x+18*kx),round(y+20*ky));
    LineTo(round(x+1*kx),round(y+20*ky));
    LineTo(round(x+1*kx),round(y +18*ky));
   //закрашиваем область
    FloodFill(round(x+5*kx),round(y +18*ky),clBlue);
    { парус}
```

```
    MoveTo(round(x+5*kx),round(y+17*ky));
    LineTo(round(x+10*kx),round(y+3*ky));
    LineTo(round(x+20*kx),round(y+14*ky));
    LineTo(round(x+5*kx),round(y+17*ky));
    FloodFill(round(x+7*kx),round(y +15*ky),clRed);
    { флаг }
    setBrushStyle(bsHatch);
    setBrushColor(clGreen);
    rectangle( round(x+10*kx),round(y+3*ky),
               round(x+16*kx),round(y+1*ky));
    {иллюминаторы}
    setBrushStyle(bsSolid);
    setBrushColor(clWhite);
    circle(round(x+16*kx),round(y+17*ky),round(kx*1));
    circle(round(x+12*kx),round(y+17.5*ky),round(kx*1));
    circle(round(x+8*kx),round(y+18*ky),round(kx*1));
  end; { DrawShip }
```

```
begin
  {SetWindowSize(700,500);}
  x:=20;  y:=20;  kx:=5;  ky:=5;  h:=2;
  LockDrawing;
  While x<= (windowWidth - 22*kx) do
  begin
    ClearWindow;
    {рисуем кораблик}
    drawShip(x,y,kx,ky);
    Redraw;
    Sleep(1);
    x := x+h;
  end;
end.
```

Задача 2. Построить график функции $f(x) = x \cdot \sin(x) \cdot \cos(x)$ на интервале $[a,b]$.

Решение:

```
uses GraphABC;
var x, h, MinF, MaxF, y, a, b: real;
    i, w, hp: integer;

  function F(x: real):real;
  BEGIN
    f:=x*sin(x)*cos(x);
  END; {F}

  function P_x(x: real; {переводимая координата}
               a,b:real;  {границы исходного интервала}
               MaxX:integer ):integer; {ширина графического поля}
  BEGIN
    P_x:=round(MaxX/(b-a)*(x-a));
  END; {P_x}
```

```
{Для преобразования y в координату экрана }
  function P_y( y: real;
                MinF,MaxF:real; {границы исходного интервала}
                MaxY:integer):integer;  {высота графического поля}
  BEGIN
    P_y:=round((MaxY-3)-(MaxY-3)/(MaxF-MinF)*(y-MinF));
  END;
 begin
  SetWindowSize(700,500);//создание графического окна
  w:=WindowWidth;
  hp:=WindowHeight;
  a:=-5;  b:=5;
  x:=a; minF:=f(x); MaxF:=F(x);  h:=(b-a)/500;
  for i:=1 to 500 do
   begin
     if maxF <= f(x) then maxF:=f(x);
     if minF >= f(x) then minF:=f(x);
     x:=x+h;
   end;
   {Вывод осей координат:
   ось OX (y=0, x=a..b) }
  Line( P_x(a,a,b,w),P_y(0,MinF,MaxF,hp),
       P_x(b,a,b,w), P_y(0,MinF,MaxF,hp));
   {ось OY (x=0,y=MinF..MaxF)}
  Line(P_x(0,a,b,w),P_y(MinF,MinF,MaxF,hp),
       P_x(0,a,b,w),P_y(MaxF,MinF,MaxF,hp));
  SetPenColor(clBlue);
  SetPenWidth(3);
   {Построение графика функции}
  x:=a;  y:=F(a);{ начальная точка}
  {курсор в начальную точку на экране}
  MoveTo(P_x(x,a,b,w), P_y(y,MinF,MaxF,hp));
  For i:=1 to 500 do
  begin
    {вычисляем X  и значение функции в этой точке}
    x:=x+h;  y:=f(x);
    {Проводим линию от текущей позиции в точку с координатами:}
    LineTo(P_x(x,a,b,w),P_y(y,MinF,MaxF,hp))
  end;
end.
```

Устные опросы и коллоквиум

Устные опросы и коллоквиум проводятся преподавателем по завершению изучения каждого раздела. Вопросы и задания приведены в приложении 2. Для подготовки используется основная и дополнительная литература по дисциплине «Основы программирования в компьютерных системах», а также информация, размещенная в LMS BlackBoard.

Вопросы, возникающие в процессе подготовки, студент может задать преподавателю любо на консультациях, либо через специальное средство LMS BlackBoard.

Тестирование

Тестирование осуществляется на занятии через систему BlackBoard. Для подготовки тестов используются пробные тесты, размещенные в системе BlackBoard. Эти тесты не оцениваются преподавателем и служат элементом самоконтроля. Оба типа теста формируются на основе одной базы вопросов.

Требования к представлению и оформлению результатов самостоятельной работы

Результатом самостоятельной работы студентов являются:

1. Программы для решения заданий расчетно-графической работы, реализованные в системе PascalABC и оформленные в соответствии с правилами оформления программ. Эти программы обязательно отсылаются преподавателю через систему BlackBoard. Защита расчетно-графического задания осуществляется только после того, как задача проверена и оценена преподавателем в системе BlackBoard.

2. Тесты самоконтроля, выполненные в системе BlackBoard. Все попытки, выполненные пользователем, сохраняются. К основному тесту студент допускается только после прохождения на зачетном уровне данного вида тестов.

Правила оформления программ

I. Блоки

1.1. Каждый оператор языка программирования записывается на отдельной строке.

1.2. Содержимое конструкций begin-end, repeat-until, case-end, record-end выделяется отступом в виде некоторого количества дополнительных пробелов в начале строки или табуляции.

1.3. Содержимое конструкций if-then, if-then-else, while-do, with-do допускается писать в одну строку, если суммарная длина строки остается

приемлемой. В случае, если содержимое перенести необходимо, то его можно вынести в отдельную строку, дополнив отступом.

1.4. Конструкцию if-then-else можно писать в две части, выделяя действие в случае невыполнения условия (else) на отдельную строку, но тогда else должен выделяться дополнительным отступом. Также по предыдущему пункту можно писать всю конструкцию в одну строку или в четыре, но никак иначе.

1.5. Если в любую конструкцию (кроме if-then-else) вложен begin-end, то begin-end можно писать на том же уровне вложенности, что и конструкцию, в которую он вложен. Более того, допускается не переносить begin на отдельную строку.

1.6. Содержимое секций uses, type, const, var должно выделяться отступом. Сами названия секций при этом должны идти без отступа в отдельной строке.

1.7. Выбранный вид отступа должен быть единым для всей программы: не допускается менять ширину отступа в пробелах в середине программы, а также менять табуляцию на пробелы и наоборот.

II. Пробелы внутри строк

2.1. Любая бинарная операция (<, >, <=, >=, <>, =, +, -, *, div, mod, /, and, or, xor, и т.д.) должна выделяться пробелом как слева, так и справа от себя. В том числе это относится и к знакуприсваивания :=.

2.2. Однострочный комментарий может идти в той же строке, что и код, но при этом он обязан

отделяться от кода двумя пробелами. Во всех иных случаях комментарий не может идти в той же строке, что и код.

2.3. Перед символами «:», «,» и «;» никогда не должен идти пробел, и всегда должен идти после (если это не конец строки).

2.4. Запрещается использовать в строке более одного пробела подряд, если эти пробелы не составляют отступ и не служат для отделения комментария от кода.

2.5. Ключевые слова array, case, do, downto, else, file, if, in, of, then, to, while, with, until всегда должны содержать по обе стороны от себя хотя бы один пробельный символ (допускается перевод строки).

2.6. Запрещено отделять знак «..» (две точки) от соседних лексем слева и справа при объявлении дипапазонов.

2.7. При обращении к элементу массива пробелы между квадратными скобками и их содержимым не ставятся. Также не ставится пробел между именем переменной и открывающейся квадратной скобкой.

III. Идентификаторы

3.1. В программе запрещается использование кириллицы (даже в комментариях).

3.2. В программе запрещается использование транслита.

3.3. В качестве имен простых переменных/констант, смысл которых достаточно очевиден, разрешается использовать короткие название из одной-двух букв. При этом, по сложившейся традиции, имена счетчиков в циклах обычно называют i и j. Для нетривиальных понятий лучше использовать английские слова.

3.4. Не стоит злоупотреблять длинными именами переменных и констант. 10-15 символов должно быть достаточно для передачи смысла переменной.

3.5. Имена переменных и констант должны содержать только строчные латинские буквы, цифры и нижние подчеркивания для разделения слов в названии.

3.6. Имена функций и процедур именуются с использованием CamelCase, то есть содержат только латинские буквы и цифры, каждое слово, кроме первого, пишется с заглавной буквы.

3.7. Все ключевые слова языка должны записываться только строчными латинскими буквами.

4.2. Каждая процедура и функция должна содержать перед и после себя ровно одну пустую строку.

4.3. В случае, если в функцию или процедуру передается переменная сложного типа (массив, множество и т.п.), она обязательно должна выделяться одним из модификаторов const или var.

4.4. Между списком аргументов и именем функции/процедуры никогда не ставится пробел.

V. Другие правила

5.1. Запрещается использование конструкции goto.

5.2. В программе запрещается использование нетривиальных констант без их явного описания в разделе const. Обычно нетривиальными считаются любые константы, кроме, быть может, -1, 0, 1, 2, 10.

5.3. Каждая переменная перед использованием обязана быть явно инициализирована присваиванием.

5.4. Запрещается сравнивать переменные типа boolean с true или false.

5.5. Внутри цикла for запрещается явное изменение значения счетчика цикла.

5.6. Запрещено использовать строки длиннее 120 символов. Строки можно разбивать на части в длинных выражениях и списках аргументов. Строки 2, 3, 4, ... одного оператора считается вложенными в первую строку и выделяются отступом.

Критерии оценки выполнения самостоятельной работы

Самостоятельная работа студентов включает расчетно-графические задания, подготовку к устным опросам, тестирование. Критерии оценки каждого вида работы приведены в приложении 2.

Приложение 2

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ИНЖЕНЕРНАЯ ШКОЛА

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине «Основы программирования в компьютерных системах»
Направление подготовки: 15.03.03 Прикладная механика
Профиль подготовки: «Математическое и компьютерное моделирование механических систем и процессов»
Форма подготовки очная

Владивосток
2016

Паспорт ФОС

Код и формулировка компетенции	Этапы формирования компетенции	
ОПК-9 владением методами информационных технологий, соблюдением основных требований информационной безопасности, в том числе защиты государственной тайны	Знает	особенности современных методологий и технологий использования и создания программных средств
	Умеет	проектировать, конструировать и отлаживать программные продукты в соответствии с заданными критериями качества и стандартами
	Владеет	навыками работы с компьютером как средством управления информацией;
ОПК-10 способностью решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности	Знает	основы событийного и объектно-ориентированного программирования
	Умеет	разрабатывать алгоритмы и программы для решения задач профессиональной деятельности средствами высокоуровневых языков программирования и визуальных систем программирования.
	Владеет	способностью использовать для решения аналитических, исследовательских и коммуникативных задач современные средства визуальной разработки программ.
ПК-7 умением извлекать актуальную научно-техническую информацию и наукометрическую информацию из электронных ресурсов, в том числе Science Direct, Elsiver Freedom Collection, SCOPUS	Знает	современные электронные научные базы данных для работы с научно-технической и наукометрической информацией при решении поставленных задач
	Умеет	извлекать актуальную научно-техническую информацию и наукометрическую информацию из электронных ресурсов, в том числе Science Direct, Elsiver Freedom Collection, SCOPUS
	Владеет	навыками обработки, анализа и интерпретации результатов исследований, а также подготовки данных для составления отчетов и презентаций, написания докладов, статей и другой научно-технической документации

№ п/п	Контролируемые разделы / темы дисциплины	Коды и этапы формирования компетенций		Оценочные средства	
				текущий контроль	промежуточная аттестация
1	Основы структурного программирования	ОПК-9, ОПК-10	Знает: особенности современных методологий и технологий использования и создания программных средств	Тест (ПР-1)	Вопросы 1-8
			Умеет: проектировать, конструировать и отлаживать программные продукты в соответствии с заданными критериями качества и стандартами	Расчетно-графическая работа (ПР-12) Лабораторная работа (ПР-6)	Задачи тип I
			Владеет: навыками работы с компьютером как средством управления информацией;	Расчетно-графическая работа (ПР-12)	Задачи тип II
2	Структурированные типы данных	ОПК-9,	Знает: особенности современных методологий и технологий использования и создания программных средств	Собеседование (УО-1)	Вопросы 9-35
			Умеет: проектировать, конструировать и отлаживать программные продукты в соответствии с заданными критериями качества и стандартами	Расчетно-графическая работа (ПР-12) Лабораторная работа (ПР-6)	Задачи тип I
			Владеет: навыками работы с компьютером как средством управления информацией;	Расчетно-графическая работа	Задачи тип II

№ п/п	Контролируемые разделы / темы дисциплины	Коды и этапы формирования компетенций		Оценочные средства	
				текущий контроль	промежуточная аттестация
				(ПР-12)	
3	Основы модульного программирования	ОПК-10, ПК-7	Знает: основы событийного и объектно-ориентированного программирования	Собеседование (УО-1)	Вопросы 36-40
			Умеет: разрабатывать алгоритмы и программы для решения задач профессиональной деятельности средствами высокоуровневых языков программирования и визуальных систем программирования.	Лабораторная работа (ПР-6)	Задачи тип I
			Владеет: способностью использовать для решения аналитических, исследовательских и коммуникативных задач современные средства визуальной разработки программ.	Расчетно-графическая работа (ПР-12)	Задачи тип II
5	Графика и визуализация	ПК-7	Знает: современные электронные научные базы данных для работы с научно-технической и наукометрической информацией при решении поставленных задач	Коллоквиум (УО-2)	Вопросы 41-48
			Умеет: извлекать актуальную научно-техническую информацию и наукометрическую информацию из электронных ресурсов, в том числе Science Direct, Elsiver Freedom Collection, SCOPUS	Лабораторная работа (ПР-6)	Задачи тип I
			Владеет: навыками обработки, анализа и интерпретации результатов исследований, а также подготовки данных для составления отчетов и презентаций, написания докладов, статей и другой научно-технической документации	Расчетно-графическая работа (ПР-12)	Задачи тип II

Шкала оценивания уровня сформированности компетенций

Код и формулировка компетенции	Этапы формирования компетенции		критерии	показатели
ОПК-9 владением методами информационных технологий, соблюдением основных требований информационной безопасности, в том числе защиты государственной тайны	Знает	особенности современных методологий и технологий использования и создания программных средств	- знание определений основных понятий методов информационных технологий; - знание основных этапов создания программных продуктов; - знание требований по информационной безопасности	- способность дать определения основных понятий информационных технологий; - способность перечислить и раскрыть суть методов информационных технологий, которые изучил и освоил обучающийся; - способность перечислить и раскрыть последовательность и содержание этапов создания программных продуктов; - способность сформулировать и раскрыть суть требований к информационной безопасности
	Умеет	проектировать, конструировать и отлаживать программные продукты в соответствии с заданными критериями качества и стандартами	- умение проектировать, конструирует программные продукты, используя стандартные методы и эталонные образцы -умение отлаживать и тестировать программные продукты для выбранной предметной области	- способность проектировать и конструировать программные продукты; - способность адаптировать стандартный алгоритм для решения конкретной задачи; - способность проводить исчерпывающее тестирование и отладку программы
	Владеет	навыками работы с компьютером как средством управления информацией;	- владение терминологией информационных и компьютерных технологий; - владение способностью сформулировать техническое задание на разработку программного продукта; - владение компьютером как средством обработки информации	- способность бегло и точно применять терминологический аппарат информационных и компьютерных технологий в устных ответах на вопросы и в письменных работах, - способность сформулировать задание техническое задание на разработку программного продукта; - способность корректно представлять знания в алгоритмической форме. - способность свободно применять стандартные программные продукты

Код и формулировка компетенции	Этапы формирования компетенции		критерии	показатели
ОПК-10 способностью решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности	Знает	основы событийного и объектно-ориентированного программирования	- знание определений основных понятий и определений структурного и модульного программирования; - знание основных способов представления реальных данных с помощью типов данных языков высокого уровня; - знание основных приемов разработки программ	- способность дать определения основных понятий и определений модульного и структурного программирования - способность представить реальные данные с помощью типов данных языков высокого уровня; - способность сформулировать и раскрыть суть основных приемы разработки программ;
	Умеет	разрабатывать алгоритмы и программы для решения задач профессиональной деятельности средствами высокоуровневых языков программирования и визуальных систем программирования.	- умение разрабатывать алгоритмы и программы для решения задач профессиональной деятельности средствами языков программирования, используя стандартные методы и эталонные образцы	- способность разрабатывать собственные программные продукты для решения стандартных задач в области профессиональной деятельности
	Владеет	способностью использовать для решения аналитических, исследовательских и коммуникативных задач современные средства визуальной разработки программ.	- владение технологиями разработки программ для решения аналитических, исследовательских и коммуникативных задач профессиональной деятельности, в том числе и в нестандартных ситуациях - владение методами командной работы;	- способность проектировать и реализовать нестандартные задачи в области профессиональной деятельности; - способность работать в проектной команде по разработке программных продуктов
ПК-7 умением извлекать актуальную научно-техническую информацию и наукометрическую информацию из электронных ресурсов, в том числе Science Direct, Elsiver Freedom Collection, SCOPUS	Знает	современные электронные научные базы данных для работы с научно-технической и наукометрической информацией при решении поставленных задач	- знание основных понятий, используемых в компьютерной графике; - знание способов изображения и трансформации растровых изображений;	- способность дать определения основных понятий, используемых в компьютерной графике; - способность сформулировать и раскрыть суть основных способов изображения и трансформации растровых изображений ;
	Умеет	извлекать актуальную научно-техническую информацию и наукометрическую информацию из электронных ресурсов, в том числе Science Direct, Elsiver Freedom	- умение использовать стандартные алгоритмы растровой графики; - умение использовать графические редакторы и другие системы визуализации данных в стандартных задачах профессиональной деятельности;	- способность использовать стандартные алгоритмы растровой графики при разработки программ; - способность использовать графические редакторы и другие графические пакеты в стандартных

Код и формулировка компетенции	Этапы формирования компетенции		критерии	показатели
		Collection, SCOPUS		задачах профессиональной деятельности;
	Владеет	навыками обработки, анализа и интерпретации результатов исследований, а также подготовки данных для составления отчетов и презентаций, написания докладов, статей и другой научно-технической документации	- владения методами проектирования и реализации программ для визуализации изображений и данных различной природы; - свободное владение пакетами графики и визуализации;	- способность проектировать и реализовать программы для визуализации изображений и данных различной природы; - способность свободно владеть пакетами графики и визуализации;

Методические рекомендации, определяющие процедуры оценивания результатов освоения дисциплины

Оценочные средства для промежуточной аттестации

Перечень типовых вопросов к зачету

1. Основные парадигмы программирования.
2. Классификация языков программирования.
3. Понятие алгоритма и программы.
4. Структура программы на языке Паскаль.
5. Целые и вещественные типы данных. Операции над ними. Правила преобразования и совместимости типов.
6. Операторы присваивания, ввода и вывода.
7. Логический тип данных.
8. Условный оператор.
9. Операторы повторения в языке Паскаль.
10. Цикл с параметрами.
11. Цикл с предусловием.
12. Цикл с постусловием.
13. Разбиение числа на цифры.

14. Алгоритм Эвклида нахождения наибольшего общего делителя двух чисел.

15. Алгоритмы вычисления сумм и произведений.

16. Алгоритмы статистической обработки данных.

17. Вычисление суммы ряда для заданного числа членов и с заданной точностью.

18. Структурированные типы данных: одномерные массивы. Операции ввода и вывода. Заполнения элементов массива случайным образом.

19. Операции над массивами: перебор, перестановка элементов, копирование массива целиком.

20. Операции над массивами: удаление и вставка элементов массива.

21. Поиск первого встретившегося и последнего встретившегося элемента в неупорядоченном массиве.

22. Сортировка пузырьком.

23. Сортировка обменом.

24. Сортировка вставками.

25. Поиск элемента в упорядоченном массиве (двоичный поиск).

26. Структурированные типы данных: двумерные массивы. Типовые операции над массивами.

27. Двумерные массивы. Арифметические операции над матрицами и векторами.

28. Типы данных, описывающие символьную информацию (тип CHAR). Операции над ними. Функции преобразования.

Перечень типовых экзаменационных вопросов

1. Тип данных для описания символьных строк. Операции, допустимые с переменными этого типа.

2. Тип данных для описания символьных строк. Встроенные операции над строками.

3. Типы данных, определяемый программистом: перечислимый тип. Операции над переменными этого типа.

4. Типы данных, определяемые программистом: интервальный тип. Операции над переменными этого типа.

5. Типы данных, определяемые программистом: тип множество. Операции над переменными этого типа.

6. Типы данных, определяемые программистом: тип запись. Операции над переменными этого типа.

7. Файлы в языке Паскаль. Операции с файлами.

8. Процедуры и функции в языке Паскаль.

9. Область видимости. Понятие локальности и глобальности.

10. Использование формальных и фактических параметров в процедурах и функциях.

11. Понятие модульности

12. Проектирование программы «сверху вниз» и «снизу вверх»

13. Типы графических изображений

14. Графические возможности языка Паскаль.

15. Основные понятия графического режима: перо, холст, канвас.

16. Процедуры реализации графических примитивов

17. Алгоритм построения графика функции

18. Относительные и абсолютные координаты для рисования изображения

19. Преобразование изображений: масштабирование и поворот.

20. Программная реализация движения.

Перечень типовых экзаменационных задач

На экзамен по каждой теме выносится два вида задач: для оценки «продвинутого уровня» студентов предлагается составить алгоритм некоторой задачи (не обязательно реализовывать полную программу), для

проверки «высокого уровня» - предлагается реализовать программу на языке Паскаль (в полном объеме).

Типовые задачи на составление алгоритма (тип I)

Раздел «Основы модульного программирования»

1. Дано натуральное число. Составить алгоритм, который позволяет получить самую большую цифру этого числа.

2. Составить алгоритм решения системы двух уравнений.

Раздел «Структурированные типы данных»

1. Написать алгоритм, который в последовательности $\mathbf{a_1,a_2,...,a_n}$ меняет местами наибольший и наименьший элементы.

2. Седловая точка – это элемент матрицы, который является наименьшим в своей строке и одновременно наибольшим в своем столбце или наоборот. Составить алгоритм поиска седловых точек матрицы **A(n,m)**.

3. Дана строка, состоящая из слов и чисел, разделенных пробелами. Написать алгоритм, который находит сумму всех отрицательных чисел из этой строки.

4. Дан текстовый файл, в который занесена информация о сдаче сессии студентами факультета по группам. Написать программу, которая вычисляет сколько пятерок было получено каждой группой по каждому предмету.

Раздел «Основы модульного программирования»

1. Реализовать функция проверки числа на простоту.

2. Реализовать функцию, вычисляющую произведение делителей натурального числа.

Раздел «Графика и визуализация»

1. Составить алгоритм построения графиков, заданных в параметрическом виде. Построить график следующей функции:

$$\begin{cases} x(t) = sin^3(t) \\ y(t) = cos^3(t) \end{cases}$$

2. Составить алгоритм построения графиков, заданных в полярной системе координат. Построить график следующей функции:

$$r(t) = 1 + 3 \cdot t$$

Типовые задачи на реализацию программы на языке Паскаль(тип II)

Раздел «Основы модульного программирования»

1. Даны координаты трех вершин треугольника найти. Написать программу, которая определяет тип этого треугольника (прямоугольный, остроугольный или тупоугольный).

2. Написать программу, которая вводит с клавиатуры последовательность из N целых чисел и найти сумму положительных и произведение отрицательных чисел этой последовательности.

Раздел «Структурированные типы данных»

1. Даны две последовательности чисел, размерности **n** и **m**. Написать программу, которая определяет являются ли элементы первой последовательность подмножеством второй.

2. Составьте программу, меняющую местами элементы квадратной матрицы симметрично побочной диагонали.

3. Дана строка, состоящая из слов, разделенных пробелами. Написать программу формирования новой строки, в которую включены все слова-перевертыши.

4. Дан текстовый файл, в который занесена информация о сдаче сессии студентами факультета по группам. Написать программу, которая вычисляет сколько пятерок было получено каждой группой по каждому предмету.

Определить наибольший общий делитель 6 натур. чисел.

Раздел «Основы модульного программирования»

1. Написать программу, которая находит числа из промежутка **[a,b]**, с максимальным числом делителей. (Предварительно реализовать функцию, вычисляющую количество делителей числа)

2. Описать функцию, определяющую, является ли число **k** степенью числа 5. С ее помощью найти количество степеней числа 5 в массиве, состоящем из **n** натуральных чисел.

Раздел «Графика и визуализация»

1. Нарисовать произвольную картинку, масштабируемую по вертикали и горизонтали и реализовать ее движение по периметру окна.

Образец экзаменационного билета

1. Классификация языков программирования.

2. Операции над массивами: перебор, перестановка элементов, копирование массива целиком

3. Составить алгоритм вычисления суммы ряда:

$$(1+x)^p = 1 + \frac{p}{1!}x + \frac{p(p-1)}{2!}x^2 + \ldots + \frac{p(p-1)(p-2)\ldots(p-n+1)}{n!}x^n + \ldots$$

4. Даны координаты центров двух окружностей (x_1,y_1) и (x_2,y_2), а также их радиусы r_1 и r_2 $(r_1>r_2)$. Написать программу, которая определяет взаимное расположение этих окружностей.

Принцип составления экзаменационного билета

Первые два вопроса являются теоретическими и предназначены для оценивания порогового уровня освоения дисциплины. Третий вопрос, на составление алгоритма, предназначен для оценки продвинутого уровня. Последний вопрос – написание программы на компьютере – для высокого уровня освоения. Таблица для составления экзаменационных билетов для двух семестров по фонду оценочных средств:

Номер вопроса	I семестр	II семестр
1	вопросы 1 – 9	вопросы 26 – 35
2	вопросы 10 – 25	вопросы 35 – 48
3	задачи I типа	задачи I типа
4	задачи II типа	задачи II типа

Критерии выставления оценки студенту на зачете по дисциплине

«Основы программирования в компьютерных системах»

Баллы (рейтинговой оценки)	**Оценка экзамена** (стандартная)	**Требования к сформированным компетенциям**
61-100	«зачтено»	Оценка «зачтено» выставляется студенту, если он глубоко и прочно усвоил программный материал по основам программирования, исчерпывающе, последовательно, четко и логически стройно его излагает, умеет составлять алгоритм и программу, а затем выполнять ее документирование, свободно справляется с задачами, вопросами и другими видами применения знаний, причем не затрудняется с ответом при видоизменении заданий, свободно использует компьютер для сбора и анализа данных, выбирает эффективный алгоритм обработки информации, правильно обосновывает принятое решение, владеет разносторонними навыками и приемами выполнения практических задач, связанных с проектированием и реализацией программ в области профессиональной деятельности.
0-60	*«незачтено»*	Оценка «незачтено» выставляется студенту, который не знает значительной части программного материала по основам программирования, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические работы, связанные с написанием программ и алгоритмов. Оценка «неудовлетворительно» ставится студентам, которые не могут продолжить обучение без дополнительных занятий по соответствующей дисциплине

Критерии выставления оценки студенту на экзамене по дисциплине

«Основы программирования в компьютерных системах»

Баллы (рейтинговой оценки)	**Оценка экзамена** (стандартная)	**Требования к сформированным компетенциям**
86-100	«отлично»	Оценка «отлично» выставляется студенту, если он глубоко и прочно усвоил программный материал по основам программирования, исчерпывающе, последовательно, четко и логически стройно его излагает, умеет составлять алгоритм и программу, а затем выполнять ее документирование, свободно справляется с задачами, вопросами и другими видами применения знаний, причем не затрудняется с ответом при видоизменении заданий, свободно использует компьютер для сбора и анализа данных, выбирает эффективный алгоритм обработки информации, правильно обосновывает принятое решение, владеет разносторонними навыками и приемами

		выполнения практических задач, связанных с проектированием и реализацией программ в области профессиональной деятельности.
76-85	*«хорошо»*	Оценка «хорошо» выставляется студенту, если он твердо знает материал по основам программирования, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопрос, правильно применяет теоретические положения при решении практических вопросов и задач, связанных с проектированием и реализацией программ в области профессиональной деятельности, владеет необходимыми навыками и приемами их выполнения с использованием информационно-коммуникационных средств.
61-75	*«удовлетвори тельно»*	Оценка «удовлетворительно» выставляется студенту, если он имеет знания только основного материала в области программирования, но не усвоил его деталей, допускает неточности, недостаточно правильные формулировки, нарушения логической последовательности в изложении программного материала, испытывает затруднения при выполнении практических работ, связанным с написанием программ и применением стандартных пакетов в области своей профессиональной деятельности.
0-60	*«неудовлетвор ительно»*	Оценка «неудовлетворительно» выставляется студенту, который не знает значительной части программного материала по основам программирования, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические работы, связанные с написанием программ и алгоритмов. Оценка «неудовлетворительно» ставится студентам, которые не могут продолжить обучение без дополнительных занятий по соответствующей дисциплине

.Оценочные средства для текущей аттестации

Вопросы для собеседований и коллоквиума по дисциплине «Основы программирования в компьютерных системах»

Раздел «Структурированные типы данных»

1. Дать определение массива.
2. Чем отличаются структурированные типы данных от простых?
3. Какие типы данных не допустимы для компонентов массива? Почему?
4. В каких разделах программы допустимо описание массива?
5. Как определяется общее число элементов массива?
6. Можно ли использовать при описании массива тип диапазон?

7. Дайте определение индекса. Какие ограничения накладываются при описании границ изменения индекса?

8. Что называется базовым типом?

9. Приведите описание двух способов объявления массива.

10. Можно ли в качестве границ индексов массива использовать переменные? Объяснить почему.

11. Как осуществляется доступ к каждому элементу массива?

12. Объявить массив, состоящий из 4 строк и 5 столбцов, элементами которого являются вещественные числа. Для этого массива реализовать:

- ввод, заполнение случайным образом (числами в интервале от 5.5 до 6.7), вывод массива;

- поиск максимального элемента в массиве, в каждом столбце и каждой строке;

- вставку строки после 3-ей и удаление 5 столбца

13. Дать определение массива.

14. Чем отличаются структурированные типы данных от простых?

15. Какие типы данных не допустимы для компонентов массива? Почему?

16. В каких разделах программы допустимо описание массива?

17. Как определяется общее число элементов массива?

18. Можно ли использовать при описании массива тип диапазон?

19. Дайте определение индекса. Какие ограничения накладываются при описании границ изменения индекса?

20. Что называется базовым типом?

21. Приведите описание двух способов объявления двухмерного массива.

22. Можно ли в качестве границ индексов массива использовать переменные? Объяснить почему.

23. Как осуществляется доступ к каждому элементу массива?

24. Какие математические объекты можно описать средствами двумерных массивов?

25. Объявить вектор, размерностью n, матрицу, размерностью n на n. Для них реализовать:

- заполнение случайным образом числами в интервале от 5.5 до 6.7;

- сложение двух векторов, сложение двух матриц, умножение матрицы на вектор, умножение двух матриц;

26. Как задается описание переменных символьного типа?

27. Каково множество значений символьного типа?

28. Приведите примеры символьных констант.

29. Что такое код символа? Какое значение он может принимать?

30. Можно ли к данным символьного типа применять операции отношения?

31. Функции ORD и CHR. Как они работают?

32. Верны ли утверждения:

ORD(‘0’)=0

‘a’=’A’

CHR(ORD(c))=c, где c – переменная символьного типа

CRH(ORD(i))=i, где i – переменная целого типа

33. Дана переменная i, описанная как диапазон 0..9. Преобразовать ее в символьную переменную, представляющую соответствующую цифру. Например, 3 – ‘3’.

34. Какие операции применимы к символьным данным? Описать их и привести примеры.

35. Могут ли переменные символьного типа использоваться как индексы массива? Если да, то в каких задачах это можно использовать?

36. Является ли тип **string** структурированным типом данных.

37. Чем отличается строковый тип от массива символов?

38. Какое максимально возможное количество символов может содержать строка?

39. Можно ли при описании типа **string** определить ее максимально возможную длину?

40. Всегда ли фактическая длина строки равна длине, указанной при описании?

41. Каким образом в переменную типа **string** поместить какое-то конкретное значение?

42. Как можно обратиться к отдельному символу строки?

43. Как происходит сравнение строк?

44. Какое значение получат следующие выражения:

‘3’>’23’

‘Aa’<’aA’

‘09’>’07’

‘Да’ > ‘Дача’

45. Каким оператором или операторами осуществляется ввод строки?

46. Перечислите основные процедуры и функции для обработки строк.

47. Верно ли, что запись относится к структурированным типам данных?

48. Чем отличается запись от массива?

49. Что такое поле записи? Верно ли, что все поля записи должны иметь один и тот же тип?

50. Как осуществляется ссылка на компоненты записи?

51. Может ли типом поля записи быть массив?

52. Могут ли записи иметь вложенную структуру?

53. Можно ли значения полей записи использовать в выражениях?

54. Какой оператор используется для упрощения доступа к полям записи?

55. Объявить тип-запись для определения следующих понятий:

- цена в рублях и копейках;

- время в часах, минутах и секундах;

- адрес (город, улица, дом, квартира);

56. Для типа, описывающего цену в рублях и копейках, реализовать ввод, вывод переменной и массива этого типа.

57. Дайте определение множеству.

58. Что называется базовым типом множества?

59. Может ли базовый тип множества быть вещественным числом? Почему?

60. Может ли множество не содержать ни одного элемента?

61. Объявить множество, базовый тип которого буквы a,b,c. Сколько и каких значений может принимать переменная данного типа?

62. Может ли множество содержать элементы разных типов?

63. Какие операции допустимы над множествами?

64. Какие множества считаются равными? Имеет ли значение порядок следования элементов в множестве?

65. Что такое объединение, пересечение и разность множеств?

66. Вычислите значения выражений:

[6] <> [6,6,6] [1, 8] = [8,1]

[‘n’,’m’] = [‘n’..’m’] [1,7] <= [1,9]

[ ]<[1, 9] trunc(6,7) in [1,5,6]

67. Описать множество, базовым типом которого является цифры. Реализовать следующие операции для этого множества:

- ввести с клавиатуры 5 цифр и занести их в множество;

- сформировать новое множество, куда входят цифры, отсутствующие в исходном множестве;

- вывести элементы сформированного множества.

68. Что такое файл?

69. Какого типа могут быть компоненты файла?

70. Сколько компонент может содержать файл?

71. Нужно ли при определении файла указывать его длину?

72. Сколько компонент файла могут быть доступны в одновременно?

73. Куда помещается при записи очередной компонент файла?

74. Какая запись будет прочитаны при выполнении операции чтения?

75. Какие действия выполняются системой при операциях **reset** и **rewrite**?

76. Можно ли открыть файл одновременно для чтения и записи?

77. Что произойдет при попытке чтения файла после того, как он был прочитан до конца?

78. Может ли файл не содержать ни одной записи? Если может, то как об этом узнать?

79. Каково назначение операции **close**?

80. Пусть файл открыт для записи. Можно ли по состоянию признака конца файла определить, сделана ли хотя бы одна запись в этот файл?

81. Приведите описание и фрагмент программы для формирования файла, состоящего из строк и последующего его считывания.

Раздел «Основы модульного программирования»

1. Дать определение процедуры и функции.
2. Описать синтаксис и семантику процедуры.
3. Описать синтаксис и семантику функции.
4. Дать определение параметра.
5. Существуют ли подпрограммы без параметров?
6. Какие параметры называются фактическими, а какие формальными?
7. Каково соответствие между формальными и фактическими параметрами?
8. Какое количество значений возвращает функция?
9. Как задать тип значения, возвращаемого функцией?
10. Каким образом осуществляется обмен данными между основной программой и процедурой без параметров?
11. Какие переменные называются локальными, а какие глобальными? В чем их отличие?
12. Может ли имя глобальной переменной совпадать с именем локальной?

13. Какие существуют способы передачи параметров? В чем их отличие?

14. При каком способе передачи изменение соответствующего формального параметра внутри процедуры изменяет и фактический параметр?

15. Можно ли функцию описать в виде процедуры? Если да, то каким образом?

16. Можно ли процедуру описать в виде функции? Если да, то при каком условии и каким образом?

17. Необходимость и преимущества использования модульного программирования.

18. Описать алгоритм разработки по программы по принципу «снизу вверх».

19. Описать алгоритм разработки программы «сверху вниз»

Раздел «Графика и визуализация»

1. Какие графические примитивы содержит модуль GraphABC?
2. Какими процедурами задают цвет и толщину пера?
3. Какими процедурами задают цвет кисти?
4. Какие процедуры используют при выводе и форматировании текста в графическом окне
5. Какие размеры имеет экран в графическом режиме Паскаль?
6. Где находится точка начала координат?
7. Как изменяются координаты x и y в Паскале?
8. Какого типа должны быть координаты объектов?
9. Какие команды рисования объектов вы знаете?
10. Как установить графический режим?
11. Движется ли что-то физически на экране при изображении движения?
12. Как организовать изменение координат при движении по вертикали?

13. Написать программу, которая демонстрирует движение прямоугольника по диагоналям: уз левого верхнего угла – в правый нижний, а затем из правого верхнего угла – в левый нижний.

14. Написать программу, которая спрашивает у пользователя какую фигуру нужно нарисовать:1 – эллипс, 2 – круг, 3 – дугу и какого цвета; и выводит на экран соответствующую фигуру

15. Изобразить на экране прямоугольник, вращающийся в плоскости экрана вокруг своей середины.

16. Составить программу, изображающую бегущую строку с каким-нибудь текстом.

17. Составить программу, которая демонстрирует движение увеличивающегося прямоугольника по диагонали.

18. Написать программу, которая спрашивает пользователя, как должен вести себя шар: 1 центра до границ экрана или 2 экрана к центру и его цвет; и показывать соответствующее изменение шарика.

19. Изобразить на экране отрезок, вращающийся в плоскости экрана вокруг точки, делящей отрезок в отношении 1:3.

20. Составить программу, изображающую бегущую строку (снизу вверх) с каким

21. Составить программу, которая показывает упругий удар двух шаров, которые движутся навстречу друг другу, сталкиваются и продолжают движение в разные стороны. Шары должны быть разного цвета.

22. Написать программу, которая спрашивает у пользователя какую фигуру нужно нарисовать:1 – прямоугольник, 2 – круг, 3 – линию и какого цвета; и выводит на экран соответствующую фигуру.

23. Изобразить на экране отрезок, вращающийся в плоскости экрана вокруг своей середины.

24. Составить программу, изображающую бегущую строку (справа налево) с каким-нибудь текстом.

Критерии оценки устных ответов на собеседовании и коллоквиуме:

✓ 100-85 баллов выставляется студенту, если его ответ показывает прочные знания основных положений изучаемого раздела программирования, отличается глубиной и полнотой раскрытия темы; владение терминологическим аппаратом; умение объяснять синтаксис и семантику основных конструкций программирования и структур данных, делать выводы и обобщения, давать аргументированные ответы, приводить примеры алгоритмов; свободное владение монологической речью, логичность и последовательность ответа; умение алгоритмически описывать проблему из выбранной предметной области.

✓ 85-76 баллов выставляется студенту, если его ответ, обнаруживающий прочные знания основных положений изучаемого раздела программирования, отличается глубиной и полнотой раскрытия темы; владение терминологическим аппаратом; умение объяснять синтаксис и семантику основных конструкций программирования и структур данных, делать выводы и обобщения, давать аргументированные ответы, приводить примеры алгоритмов; свободное владение монологической речью, логичность и последовательность ответа; умение алгоритмически описывать проблему из выбранной предметной области. Однако допускается одна - две неточности в ответе.

✓ 75-61 балл выставляется студенту, если его ответ, свидетельствующий в основном о знании основных положений изучаемого раздела программирования, отличающийся недостаточной глубиной и полнотой раскрытия темы; знанием основных вопросов теории; слабо сформированными навыками объяснения синтаксис и семантику основных конструкций программирования и структур данных, недостаточным умением давать аргументированные ответы и приводить примеры алгоритмов; недостаточно свободным владением монологической речью, логичностью и последовательностью ответа. Допускается несколько ошибок в содержании ответа; неумение алгоритмически описывать проблему из выбранной предметной области.

✓ 60-50 баллов выставляется студенту, если его ответ, обнаруживающий незнание процессов основных положений изучаемого раздела программирования, отличающийся неглубоким раскрытием темы; незнанием основных вопросов теории, несформированными навыками анализа синтаксиса и семантики основных конструкций программирования и структур данных; неумением давать аргументированные ответы, слабым владением монологической речью, отсутствием логичности и последовательности. Допускаются серьезные ошибки в содержании ответа; неумение алгоритмически описывать проблему из выбранной предметной области

Тестовые задания по дисциплине «Основы программирования в компьютерных системах»

Каждому студенту формируется индивидуальный тест, в который входит 20 вопросов, выбранных случайным образом из списка заданий. В тесте реализованы следующее типы тестовых заданий:

- выбор одного варианта ответа;
- выбор нескольких вариантов ответа;
- ввод числового значения;
- выбор правильного порядка следования ответов.

Типовых заданий теста по разделу «Основы структурного программирования»

1. Какие из перечисленных идентификаторов являются допустимыми в языке Pascal?

a) Count_Num

b) Num_1

c) F_and_S

d) 1_Num

e) First N

2. Какое значение получат целочисленные переменные x и y после выполнения следующих операторов:

Num1=16; Num2=5;

x := Num1 DIV 3 - Num2*6 MOD 4

y := (Num2- Num1) MOD 6 + 11

a) x = 3; y= 16

b) x = 5; y= 16

c) x = 3; y= 12

d) x = 12; y= 5

3. Переменная s в результате выполнения команд

```
s:=0;
FOR k:=1 TO 4 DO
  s:=s+k;
```

получит значение…

4. Установите в правильной последовательности команды, позволяющие вывести таблицу значений функции y=sin x для x от 0 до 2 с шагом 0.2

a) x:=0;

b) REPEAT

c) y:=sin(x);

d) writeln('x=', x, ' y=',y);

e) x:=x+0.2;

f) UNTIL x>2;

Критерии оценки теста, состоящего из 20 вопросов

- ✓ 14-20 баллов – считается, что тест пройден.
- ✓ 0-13 баллов – тест не засчитывается

Комплекты заданий для выполнения расчетно-графических работ по дисциплине «Основы программирования в компьютерных системах»

Расчетно-графическая работа по разделу «Основы структурного программирования», тема «Выражение и ветвление» базовый уровень

Расчетно-графическая работа состоит из трех заданий. В каждом задании требуется реализовать программу на языке Паскаль для некоторой

задачи. При этом необходимо использовать следующие операторы: ввод, вывод, присваивание и условный.

Первое задание – вычислительная геометрическая задача.

Второе задание – задача на разбиение числа на цифры и выполнения некоторых операций с полученными цифрами

Третье задание – текстовая задача.

Типовое задание

1. Найти уравнение прямой, проходящей через две различные точки, заданные своими координатами.
2. Определить, равна ли сумма двух первых цифр заданного четырехзначного натурального числа сумме двух его последних цифр.
3. На поле **(k, p)** шахматной доски расположен ферзь. Угрожает ли он полю **(m, n)**?

Расчетно-графическая работа по разделу «Основы структурного программирования», тема «Операторы повторения», продвинутый уровень

Расчетно-графическая работа состоит из трех заданий. В каждом задании требуется реализовать программу на языке Паскаль для некоторой задачи. При этом необходимо использовать следующие операторы: ввод, вывод, присваивание, условный, циклы.

Первое задание – переборная задача для последовательности вводимых чисел (поиск максимума, суммирование и пр.).

Второе задание – вычисление суммы ряда либо для заданного числа членов, либо с заданной точностью.

Третье задание – вычислительная задача, в которой используются простые или вложенные циклы.

Типовое задание

1. Ввести с клавиатуры последовательность из **n** целых чисел и найти максимальное число среди элементов, стоящих на четных местах.
2. Вычислить сумму ряда для заданного числа членов **n** (**n** и **x** вводятся с клавиатуры):

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{5!} + \cdots + (-1)^n \frac{x^{2n}}{(2n)!}$$

3. Определить, является ли натуральное число **k** степенью **3.**

Расчетно-графическая работа по разделу «Структурированные типы данных», тема «Одномерные массивы», базовый уровень

Расчетно-графическая работа состоит из трех заданий. В каждом задании требуется реализовать программу на языке Паскаль для некоторой задачи, в которой обрабатывается одномерный массив.

Первое задание – переборная задача для одномерного массива (поиск максимума, суммирование и пр.).

Второе задание – удаление или вставка элементов массива.

Третье задание – вычислительная задача, в которой используются одномерные массивы.

Типовое задание

1. Дана последовательность из **n** целых чисел. Найти количество чисел этой последовательности, расположенных между первым максимальным и последним минимальным элементами.
2. Дана последовательность из **n** целых чисел. Если сумма ее элементов отрицательна, то вставить перед первым элементов новый, равный по модулю этой сумме.
3. Даны две последовательности по **n** целых чисел в каждой. Найти наименьшее среди тех чисел первой последовательности, которые не входят во вторую.

Расчетно-графическая работа по разделу «Структурированные типы данных», тема «Двумерные массивы», базовый уровень

Расчетно-графическая работа состоит из трех заданий. В каждом задании требуется реализовать программу на языке Паскаль для некоторой задачи, в которой обрабатывается двумерный массив.

Первое задание – заполнение квадратного массива по некоторому правилу

Второе задание – переборные задачи для двумерных массивов.

Третье задание – вычислительная задача, в которой используются одномерные и двумерные массивы.

Типовое задание

1. Заполнить матрицу **n** на **n** следующим образом (пустые клетки заполняются **0**):

1	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	1	**1**	**1**	**1**	**1**	**1**	
		1	**1**	**1**	**1**		
			1	**1**			

2. Определить **k**-количество “особых” элементов массива **A**, считая элемент “особым” если он больше суммы остальных элементов своего столбца.
3. Дана вещественная матрица **A(n,m).** Упорядочить ее строки по не убыванию их первых элементов.

Расчетно-графическая работа по разделу «Структурированные типы данных», тема «Типы данных, определяемые программистом», продвинутый уровень

Расчетно-графическая работа состоит из четырех заданий. В каждом задании требуется реализовать программу на языке Паскаль для некоторой задачи, в которой используются различные типы данных.

Первое задание – обработка символьной строки.

Второе задание – обработка массива строк.

Третье задание – задачи, в которых используются пользовательские типы данных (интервальный тип, тип диапазон, запись, множество).

Четвертое задание – обработка файлов.

Типовое задание

1. Проверить, правильно ли в строку входят круглые скобки.
2. Дана последовательность строк, состоящих из слов и чисел, разделенных пробелами. Признак конца последовательности - строка ‘STOP’. Вывести в алфавитном порядке все различные слова в тексте.
3. Написать программу, включающую в себя следующие компоненты:

Объявить:

- константу **n**, равную 10;
- тип данных **TPoint,** для описания точки на плоскости с координатами **(x,y);**
- переменную **Point**, описывающую точку и массив **Point_arr**, описывающий **n** точек на плоскости.

Реализовать:

а) функцию **Len_line**, вычисляющую расстояние между двумя точками **p1** и **p2**, точки **p1** и **p2** передать в качестве параметров.

б) основную программу, в которой выполняется следующая последовательность действий:

- заполнение случайным образом массива **Point_arr**;
- вывод на экран массива точек;
- вычисление длины ломаной, построенной последовательно по точкам массива (первый отрезок ломанной составляют первая и вторая точки массива, второй отрезок ломаной – вторая и третья точка, **n-1** отрезок – точки **n-1** и **n**);
- вывод длины ломаной на экран.

4. Дан текстовый файл, в который занесена информация о сдаче сессии студентами университета по группам. Количество предметов 4. Найти

средний бал сдачи экзаменов по каждой группе и средний бал по университету в целом.

Расчетно-графическая работа по разделу «Основы модульного программирования»

Расчетно-графическая работа состоит из двух заданий. В каждом задании требуется реализовать программу на языке Паскаль для некоторой задачи, в которой реализуются процедуры или функции.

Первое задание – реализация и использование процедур и функций.

Второе задание – реализация и использование процедур и функций, усложненный вариант

Типовое задание

1. Дано **n** точек, заданных своими координатами. Найти пару точек, расстояние между которыми наибольшее. (Реализовать функцию вычисления расстояния между точками)

2. Для каждой из точек **1,2,3,...,k** напечатать значение **f(i)** количества целых чисел взаимно простых с **i** меньших **k**. Если наибольший общий делитель двух чисел равен 1, то числа называют взаимно простыми. (Реализовать функцию вычисления наибольшего общего делителя)

Расчетно-графическая работа по разделу «Графика и визуализация»

Расчетно-графическая работа состоит из двух заданий. В каждом задании требуется реализовать программу на языке Паскаль для некоторой задачи, в которой используются графический режим работы.

Первое задание – масштабирование и движение изображения.

Второе задание – визуализация графика функции или области.

Типовое задание

1. Нарисовать произвольную картинку, масштабируемую по вертикали и горизонтали и реализовать ее движение по горизонтали с ускорением, так чтобы картинка не выходила за границы окна.

2. Реализовать построение графиков, заданных в полярной системе координат. Построить график следующей функции:

$r(t) = 1 + 3 \cdot t$

Критерии оценки расчетно-графической работы

✓ 10-8 баллов выставляется студенту, если студент выполнил все пункты расчётно-графического задания. Фактических ошибок, связанных с пониманием проблемы, нет; семантических и синтаксических ошибок в программах нет. При защите студент отвечает на все вопросы преподавателя.

✓ 7-6 баллов – работа выполнена полностью; одна задача реализована не для всех исходных данных или одна программа оформлена не в соответствии со стандартами. При защите студент отвечает на все вопросы преподавателя.

✓ 5-4 балла – работа выполнена полностью. Две задачи реализованы не для всех входных данных или две программы оформлены не в соответствии со стандартами. При защите студент не отвечает на 1-2 вопроса преподавателя.

✓ 1-3 балла – работа выполнена не полностью. Три задачи реализованы не для всех входных данных или три программы оформлены не в соответствии со стандартами. При защите студент не отвечает на более, чем на 2 вопроса преподавателя.

Лабораторные работы

Структура каждой лабораторной работы следующая:

1. Реализация нескольких задач на языке Паскаль в системе PascalABC по предложенному образцу.
2. Задания для самостоятельной работы.

Все лабораторные работы можно проверить через специальное средство, реализованное в системе BlackBoard. Выполненные лабораторные работы необходимо защитить, но предварительно они должны быть отправлены преподавателю на проверку через систему BlackBoard.

Типовая лабораторная работа
на тему «Операторы повторения»

Задание 1. Распечатать все четные числа, принадлежащие интервалу **[a, b]**. Числа вывести по 15 элементов в строке.

```
program zl;
var
  a, b, i, k: integer;
begin
  write('a='); readln(a);
  write('b='); readln(b);
  if a > b
    then  writeln('Значение a должно быть меньше или равно b')
  else
  begin
   k:=0;// для того, чтобы считать сколько элементов выведено
   for i:= a to b do
     if not odd(i) then
     begin
       k:= k+1;
       write(i:4);
       if k mod 15 =0 then writeln;
     end;
  end;
end.
```

Тесты

1) a = -10, b =23 2) a= -60, b = 78 3) a= 1, b = 14 4) a = 0, b = -20

Задания для самостоятельной работы:

2. Распечатать все числа, делящиеся на 3, принадлежащие интервалу **[a, b]**. Числа вывести по 10 элементов в строке. Напечатать их количество.
3. Вывести **n** элементов арифметической прогрессии с шагом **h** и первым членом **a1**. Элементы прогрессии вывести в строку через запятую.

Посчитать сумму этих элементов. Формула для вычисления очередного элемента: $\mathbf{a_{i+1} = a_i + h}$.

4. Ввести с клавиатуры последовательность целых чисел, заканчивающуюся 0. Найти среднее арифметическое этой последовательности.

Задание 5. Напечатать таблицу значений функции на интервале **[a,b]**. Таблица значений должна содержать **n** элементов.

$$F(x) = \frac{\sin(x) \cdot \cos(x)}{1 + \sin^2(x)}$$

```
program z5;
var
   n: integer;
   f, x, h, a, b : real;
begin
  write('n='); readln(n);
  write('a='); readln(a);
  write('b='); readln(b);
  if (a > b) or (n<0)
    then  writeln('Ошибочные входные данные')
  else
  begin
   h := (b-a)/(n-1); // вычисляем шаг изменения аргумента
функции
   x := a;
   writeln('----------------------');
   writeln('|     x    |     F(x)   |');
   writeln('----------------------');
   while x <= b do
   begin
     f:= sin(x)*cos(x)/(1+sqr(sin(x)));
     writeln('| ',x:6:2,' | ',f:8:3,' |');
     x := x + h; // переходим к следующему x
   end;
   writeln('----------------------');
  end;
end.
```

Тесты

1) n=10, a = -7, b =3 2) n=15, a = 0, b =3 3) n=5, a= -10, b = 10 4) n=20, a = 5, b =1

Задания для самостоятельной работы:

6. Исправить задачу 2 так, чтобы после таблицы значений функции выводились следующие характеристики:

   - максимальное и минимальное значения функции;

- количество положительных, отрицательных и нулевых значений функции.

Задание 7. Составить все двухэлементные подможества множества целых чисел в интервале **[a,b].**

```
program z7;
var
   i, j, a, b : integer;
begin
  write('a='); readln(a);
  write('b='); readln(b);
  if (a > b)
    then  writeln('Значение b должно быть больше или равно a')
  else
  begin
   for i := a to b do
   begin
     for  j := i+1 to b do
        write ('(',i,',',j,') ');
     writeln;
   end;
  if (a = b) then writeln('двухэлементных подмножеств множество не
имеет')
  end;
end.
```

Тесты

1) a = 7, b =12 2) a= 0, b = 5 3) a= 4, b = 2 4) a = 10, b = 10

Задания для самостоятельной работы:

8. Составить все трехэлементные подможества множества целых чисел в интервале **[a,b]** и посчитать их количество.
9. Решить в целых числах уравнение: $x^2 - x \cdot y - 2y^2 = 7$

Критерии оценки лабораторной работы

✓ 10-8 баллов выставляется студенту, если студент выполнил все задания лабораторной работы, в том числе и самостоятельные. Фактических ошибок, связанных с пониманием проблемы, нет; семантических и синтаксических ошибок в программах нет. При защите студент отвечает на все вопросы преподавателя.

✓ 7-6 баллов – работа выполнена полностью; студент выполнил все предложенные в лабораторной работе задания, одно самостоятельное задание

реализована не для всех исходных данных или одна программа оформлена не в соответствии со стандартами. При защите студент отвечает на все вопросы преподавателя.

- ✓ 5-4 балла – работа выполнена полностью. Два самостоятельных задания реализованы не для всех входных данных или две программы оформлены не в соответствии со стандартами. При защите студент не отвечает на 1-2 вопроса преподавателя.
- ✓ 1-3 балла – работа выполнена не полностью. Все самостоятельные задания реализованы не для всех входных данных или все программы, которые необходимо выполнить самостоятельно, оформлены не в соответствии со стандартами. При защите студент не отвечает более, чем на 2 вопроса преподавателя.