

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКАЯ ГОСУДАРСТВЕННАЯ АКАДЕМИЯ»

«УТВЕРЖДАЮ»

Проректор по учебной работе

« 31 » марта 2021 г.

Г.Ю. Нагорная



РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Информатика и программирование

Уровень образовательной программы _____ бакалавриат _____

Направление подготовки _____ 09.03.03 Прикладная информатика _____

Направленность (профиль) _____ Прикладная информатика в экономике _____

Форма обучения _____ очная _____

Срок освоения ОП _____ 4 года _____

Кафедра разработчик РПД _____ Общая информатика _____

Выпускающая кафедра _____ Прикладная информатика _____

Начальник
учебно-методического управления

Семенова Л.У.

Директор института

Тебуев Д.Б.

Заведующий выпускающей кафедрой

Хапаева Л.Х.

г. Черкесск, 2021 г.

СОДЕРЖАНИЕ

| | |
|--|-----|
| 1. Цели освоения дисциплины | 4 |
| 2. Место дисциплины в структуре образовательной программы | 4 |
| 3. Планируемые результаты обучения по дисциплине | 5 |
| 4. Структура и содержание дисциплины | 6 |
| 4.1. Объем дисциплины и виды учебной работы..... | 6 |
| 4.2. Содержание дисциплины | 7 |
| 4.2.1. Разделы (темы) дисциплины, виды учебной деятельности и формы контроля..... | 7 |
| 4.2.2. Лекционный курс | 9 |
| 4.2.3. Лабораторный практикум | 12 |
| 4.2.4. Практические занятия | 13 |
| 4.3. Самостоятельная работа обучающегося..... | 14 |
| 5. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине | 16 |
| 6. Образовательные технологии | 21 |
| 7. Учебно-методическое и информационное обеспечение дисциплины | 21 |
| 7.1. Перечень основной и дополнительной учебной литературы..... | 21 |
| 7.2. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»..... | 23 |
| 7.3. Информационные технологии, лицензионное программное обеспечение | 24 |
| 8. Материально-техническое обеспечение дисциплины | 25 |
| 8.1. Требования к аудиториям (помещениям, местам) для проведения занятий | 25 |
| 8.2. Требования к оборудованию рабочих мест преподавателя и обучающихся | 26 |
| 8.3. Требования к специализированному оборудованию..... | 26 |
| 9. Особенности реализации дисциплины для инвалидов и лиц с ограниченными возможностями здоровья | 27 |
| Приложение 1. Фонд оценочных средств | 28 |
| Рецензия на рабочую программу | 108 |
| Лист переутверждения рабочей программы дисциплины | 109 |
| Приложение 2. Аннотация рабочей программы | 110 |

1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Целью освоения дисциплины «Информатика и программирование» является приобретение обучающимися фундаментальных теоретических и практических знаний в области информатики и программирования, формирование основ для её профессионального использования, а также способность разрабатывать алгоритмы и программы, пригодные для практического применения

Задачами дисциплины являются:

- формирование у обучающихся информационной культуры на основе разъяснения роли информатики и вычислительной техники в развитии общества и ускорении научно-технического прогресса;
- ознакомление с основными понятиями информатики, теории кодирования, теории алгоритмов, формирование и развитие на этой основе логического и алгоритмического мышления обучающихся, развитие их творческого потенциала;
- систематизация приёмов и методов работы с аппаратными и программными средствами вычислительной техники;
- ознакомление с современными технологиями программирования, основными понятиями, методами и принципами разработки программ, языками программирования высокого уровня, перспективными направлениями развития программного обеспечения;
- формирование и развитие у обучающихся устойчивых навыков программирования задач, их решения на ЭВМ, формирование практических навыков работы с системным, инструментальным и прикладным программным обеспечением.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

2.1. Дисциплина «Информатика и программирование» относится к обязательной части, Блока 1. Дисциплины (модули), имеет тесную связь с другими дисциплинами.

2.2. В таблице приведены предшествующие и последующие дисциплины, направленные на формирование компетенций дисциплины в соответствии с матрицей компетенций ОП.

Предшествующие дисциплины, направленные на формирование компетенций

| № п/п | Предшествующие дисциплины | Последующие дисциплины |
|-------|--|--|
| 1 | Иностранный язык Методика преподавания информатики и математики в школе Введение в специальность | Вычислительные системы, сети и телекоммуникации Информационные технологии Базы данных Компьютерная графика Разработка и стандартизация программных средств и информационных технологий Интеллектуальные информационные системы Высокоуровневые методы информатики и программирования |

3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Планируемые результаты освоения образовательной программы (ОП) – компетенции обучающихся определяются требованиями стандарта по направлению подготовки 09.03.03 Прикладная информатика и формируются в соответствии с матрицей компетенций ОП

| № п/п | Номер/индекс компетенции | Наименование компетенции (или ее части) | В результате изучения дисциплины обучающиеся должны: |
|--------------|---------------------------------|--|--|
| 1 | 2 | 3 | 4 |
| 1. | ОПК-7 | Способен разрабатывать алгоритмы и программы, пригодные для практического применения | <p>ОПК-7.1 Применяет языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ.</p> <p>ОПК-7.2 Программирует, отлаживает и тестирует прототипы программно-технических комплексов задач</p> <p>ОПК-7.3 Разрабатывает алгоритмы решения прикладных задач с использованием математических и современных аналитических методов</p> |

4. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1. ОБЪЕМ ДИСЦИПЛИНЫ И ВИДЫ УЧЕБНОЙ РАБОТЫ

| Вид учебной работы | | Всего часов | Семестры | |
|--|------------------------|-------------|-------------|-------------|
| | | | № 1 | № 2 |
| 1 | | 2 | 3 | 4 |
| Аудиторная контактная работа (всего) | | 144 | 72 | 72 |
| В том числе: | | | | |
| Лекции (Л) | | 54 | 36 | 18 |
| Практические занятия (ПЗ), Семинары (С) | | 18 | | 18 |
| Лабораторные работы (ЛР) | | 72 | 36 | 36 |
| Контактная внеаудиторная работа | | 4 | 2 | 2 |
| В том числе: индивидуальные и групповые консультации | | 4 | 2 | 2 |
| Самостоятельная работа обучающегося (СРО) (всего) | | 77 | 34 | 43 |
| <i>Подготовка к лабораторным работам (ЛР)</i> | | 7 | 3 | 4 |
| <i>Подготовка к практическим занятиям (ПЗ)</i> | | 7 | 3 | 4 |
| <i>Подготовка к лекционным занятиям (Л)</i> | | 10 | 6 | 4 |
| <i>Просмотр видеолекций</i> | | 4 | 2 | 2 |
| <i>Работа с книжными источниками</i> | | 12 | 5 | 7 |
| <i>Работа с электронными источниками</i> | | 12 | 5 | 7 |
| <i>Подготовка к текущему тестовому контролю</i> | | 16 | 6 | 10 |
| <i>Самоподготовка: внеаудиторное чтение, тестовый контроль</i> | | 9 | 4 | 5 |
| Промежуточная аттестация | Экзамен(Э) | Э | Э | Э |
| | экзамен (Э) | 63 | 36 | 27 |
| | в том числе: | | | |
| | Прием экз., час. | 1 | 0,5 | 0,5 |
| | Консультация, час. | 4 | 2 | 2 |
| | СРО, час. | 58 | 33,5 | 24,5 |
| ИТОГО: Общая трудоемкость | Часов | 288 | 144 | 144 |
| | зачетных единиц | 8 | 4 | 4 |

4.2. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.2.1. Разделы (темы) дисциплины, виды учебной деятельности и формы контроля

| № п/п | Наименование раздела (темы) дисциплины | Виды учебной деятельности, включая самостоятельную работу обучающихся (в часах) | | | | | Формы текущей и промежуточной аттестации |
|----------------------------|---|---|-----------|----|-----------|------------|--|
| | | Л | ЛР | ПЗ | СРО | всего | |
| 1 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Семестр 1 | | | | | | | |
| 1. | Раздел 1. Основные понятия информатики | 4 | 2 | | 6 | 12 | Коллоквиум, устный опрос, тестирование, проверка выполнения лабораторных работ |
| 2. | Раздел 2. Основы алгоритмизации | 6 | 6 | | 7 | 19 | |
| 3. | Раздел 3. Технологии программирования | 8 | 8 | | 7 | 23 | |
| 4. | Раздел 4. Программирование на языке Паскаль. | 8 | 10 | | 7 | 25 | |
| 5. | Раздел 5. Операторы языка Паскаль. | 10 | 10 | | 7 | 27 | |
| 6. | Контактная внеаудиторная работа | | | | | 2 | индивидуальные и групповые консультации |
| 7. | Промежуточная аттестация | | | | | 36 | Экзамен |
| Итого в 1 семестре: | | 36 | 36 | | 34 | 144 | |
| Семестр 2 | | | | | | | |
| 8. | Раздел 6. Подпрограммы, их назначение и классификация. Нестандартные типы данных. | 6 | 10 | 6 | 12 | 34 | Коллоквиум, устный опрос, тестирование, проверка выполнения лабораторных работ |
| 9. | Раздел 7. Записи. Комбинированный тип данных. | 4 | 10 | 4 | 8 | 26 | |
| 10. | Раздел 8. Динамические структуры и указатели. | 4 | 10 | 4 | 8 | 26 | |
| 11. | Раздел 9. Основы компьютерной графики | 2 | 4 | 4 | 8 | 18 | |

| | | | | | | | |
|-----------------------------|---|-----------|-----------|-----------|-----------|------------|--|
| 12. | Раздел 10. Компьютерные сети. Internet. Защита информации. | 2 | 2 | | 7 | 11 | |
| | Контактная внеаудиторная работа | | | | | 2 | индивидуальные и групповые консультации |
| | Промежуточная аттестация | | | | | 27 | Экзамен |
| Итого во 2 семестре: | | 18 | 36 | 18 | 43 | 144 | |
| ВСЕГО ЗА ГОД: | | 54 | 72 | 18 | 77 | 288 | |

4.2.2. Лекционный курс

| № п/п | Наименование раздела дисциплины | Наименование темы лекции | Содержание лекции | Всего часов |
|------------------|--|--------------------------------------|--|-------------|
| | | | | ОФО |
| 1 | 2 | 3 | 4 | 5 |
| Семестр 1 | | | | |
| 1. | Раздел 1. Основные понятия информатики | Тема 1. Основные понятия информатики | Предмет, задачи и структура курса, его связь с другими дисциплинами. Роль вычислительной техники в развитии современного общества. Понятие информации и информационного процесса. Формы и способы передачи информации. Сообщения и сигналы. Непрерывные и дискретные сообщения. Основные подходы к измерению количества информации. Понятие энтропии и ее свойства. Определение энтропии по формуле Шеннона. Мера Хартли. Свойства энтропии дискретных и непрерывных сообщений. Единицы измерения количества информации. Технические и программные средства реализации информационных процессов. Классификация ЭВМ и основные этапы их развития. | 4 |
| 2. | Раздел 2. Основы алгоритмизации | Тема 2. Основы алгоритмизации | Понятие алгоритма и его основные свойства: массовость, дискретность, детерминированность, результативность. Численные и логические алгоритмы. Основные этапы разработки алгоритмов: постановка задачи, построение математической модели, разработка алгоритма решения задачи, проверка правильности и оценка сложности алгоритма. Формы и способы представления алгоритмов. Правила построения алгоритмов из базовых алгоритмических конструкций. Типы алгоритмических процессов: линейные, ветвящиеся, циклические. Арифметические и итерационные циклы. Вспомогательные алгоритмы. | 6 |
| 3. | Раздел 3. Технологии программирования | Тема 3. Технологии программирования | История развития и классификация языков программирования. Краткий обзор современных парадигм программирования: процедурная, объектно-ориентированная, | 8 |

| | | | | |
|----|--|--|---|---|
| | | | <p>функциональная. Сравнительная характеристика языков программирования высокого уровня. Структура алгоритмического языка. Понятие синтаксиса, семантики, прагматики и лексики. Формальное определение грамматики языка и ее элементы. Понятие и структура системы программирования. Последовательность обработки программы от исходного текста на языке высокого уровня до исполняемого кода. Назначение и функции транслятора. Компиляторы и интерпретаторы. Основные этапы трансляции программы: лексический, синтаксический и семантический анализ, генерация и оптимизация объектного кода. Многопроходные и однопроходные компиляторы. Особенности построения интерпретаторов. Назначение и функции компоновщика. Схема функционирования редактора связей. Структура объектного и загрузочного модуля. Назначение и функции загрузчика прикладных программ. Методы трансляции адресов программы. Особенности функционирования динамических загрузчиков. Библиотеки подпрограмм как составная часть системы программирования. Статические и динамически загружаемые библиотеки. Достоинства и недостатки динамической загрузки. Понятие мобильности и обеспечение переносимости программных продуктов. Мобильные системы программирования.</p> | |
| 4. | Раздел 4. Программирование на языке Паскаль. | Тема 4. Программирование на языке Паскаль. | <p>Назначение алгоритмического языка Паскаль. Основные символы языка. Простейшие конструкции. Структура программного модуля. Классификация операторов. Синтаксис языка Паскаль. Типы данных. Алгебраические и логические операции, математические функции. Управляющие конструкции языка Паскаль. Массивы и записи в Паскале. Процедуры и функции в Паскале.</p> | 8 |

| | | | | |
|----------------------------------|---|---|--|-----------|
| | | | Динамическое распределение памяти. Указатели. | |
| 5. | Раздел 5. Операторы языка Паскаль. | Тема 5. Организация программ линейной, разветвляющейся, циклической структур. | Операторы языка Паскаль. Оператор перехода. Условный оператор. Организация программ разветвляющейся структуры. Оператор выбора. Операторы цикла. Циклы с заданным и неявным числом повторений. Одномерные массивы. Вычисление суммы и произведения. Нахождение наибольшего и наименьшего значений. Вложенные циклы. Двумерные массивы. | 10 |
| Итого часов в 1 семестре: | | | | 36 |
| Семестр 2 | | | | |
| 6. | Раздел 6. Подпрограммы, их назначение и классификация. Нестандартные типы данных. | Тема 6. Подпрограммы, их назначение и классификация. | Оформление подпрограмм и обращение к ним. Подпрограмма-функция. Подпрограмма-процедура. Переменные типы данных. Основные понятия и средства компьютерной графики в Паскале. | 6 |
| 7. | Раздел 7. Записи. Комбинированный тип данных. | Тема 7. Записи. Комбинированный тип данных. | Структура записи. Поля. Операции над записями. Оператор присоединения. | 4 |
| 8. | Раздел 8. Динамические структуры и указатели. | Тема 8. Динамические структуры и указатели. | Указатель. Стандартные процедуры размещения и освобождения динамической памяти. Стандартные функции обработки динамической памяти. | 4 |
| 9. | Раздел 9. Основы компьютерной графики | Тема 9. Основы компьютерной графики | Базовые основы компьютерной графики. Виды компьютерной графики. Основные понятия. Основные понятия растровой графики. Основные понятия векторной графики. Понятие о фрактальной графике. Программные средства компьютерной графики. | 2 |
| 10. | Раздел 10. Компьютерные сети. Internet. Защита информации. | Тема 10. Компьютерные сети. Internet. Защита информации. | Компьютерные сети. Основные характеристики. Структура и классификация компьютерных сетей. Локальные вычислительные сети (ЛВС). Структура Internet. Компьютерные вирусы и антивирусные программы. Основы защиты информации, методы защиты информации. | 2 |
| Итого часов в 2 семестре: | | | | 18 |
| Всего: | | | | 54 |

4.2.3.Лабораторный практикум

| № п/п | Наименование раздела дисциплины | Наименование лабораторного занятия | Содержание лабораторного занятия | Всего часов |
|----------------------------------|--|--|---|-------------|
| | | | | ОФО |
| 1 | 2 | 3 | 4 | 5 |
| Семестр 1 | | | | |
| 1. | Раздел 1. Основные понятия информатики | Тема 1. Основные понятия информатики | Определение энтропии по формуле Шеннона. Мера Хартли. Свойства энтропии дискретных и непрерывных сообщений. Единицы измерения количества информации. | 2 |
| 2. | Раздел 2. Основы алгоритмизации | Тема 2. Основы алгоритмизации | Основные этапы разработки алгоритмов: постановка задачи, построение математической модели, разработка алгоритма решения задачи, проверка правильности и оценка сложности алгоритма. | 6 |
| 3. | Раздел 3. Технологии программирования | Тема 3. Технологии программирования | Структура алгоритмического языка. Понятие синтаксиса, семантики, прагматики и лексики. | 8 |
| 4. | Раздел 4. Программирование на языке Паскаль. | Тема 4. Программирование на языке Паскаль. | Назначение алгоритмического языка Паскаль. Основные символы языка. Простейшие конструкции. Структура программного модуля. Классификация операторов. Синтаксис языка Паскаль. Типы данных. Алгебраические и логические операции, математические функции. Управляющие конструкции языка Паскаль. Массивы и записи в Паскале. Процедуры и функции в Паскале. | 10 |
| 5. | Раздел 5. Операторы языка Паскаль. | Тема 5. Операторы языка Паскаль. | Операторы языка Паскаль. Оператор перехода. Условный оператор. Организация программ разветвляющейся структуры. Оператор выбора. Операторы цикла. Циклы с заданным и неявным числом повторений. Одномерные массивы. Вычисление суммы и произведения. Нахождение наибольшего и наименьшего значений. Вложенные циклы. Двумерные массивы. | 10 |
| Итого часов в 1 семестре: | | | | 36 |

| Семестр 2 | | | | |
|----------------------------------|---|--|---|-----------|
| 6. | Раздел 6. Подпрограммы, их назначение и классификация. Нестандартные типы данных. | Тема 6. Подпрограммы, их назначение и классификация. | Оформление подпрограмм и обращение к ним. Подпрограмма-функция. Подпрограмма-процедура. Переменные типы данных. Основные понятия и средства компьютерной графики в Паскале. | 10 |
| 7. | Раздел 7. Записи. Комбинированный тип данных. | Тема 7. Записи. Комбинированный тип данных. | Структура записи. Поля. Операции над записями. Оператор присоединения. | 10 |
| 8. | Раздел 8. Динамические структуры и указатели. | Тема 8. Динамические структуры и указатели. | Динамическое распределение памяти. Указатели. | 10 |
| 9. | Раздел 9. Основы компьютерной графики | Тема 9. Основы компьютерной графики | Программные средства компьютерной графики. | 4 |
| 10. | Раздел 10. Компьютерные сети. Internet. Защита информации. | Тема 10. Компьютерные сети. Internet. Защита информации. | Методы защиты информации. | 2 |
| Итого часов в 2 семестре: | | | | 36 |
| Всего: | | | | 72 |

4.2.4. Практические занятия

| № п/п | Наименование раздела дисциплины | Наименование практического занятия | Содержание практического занятия | Всего часов |
|------------------|---|---|--|-------------|
| | | | | ОФО |
| 1 | 2 | 3 | 4 | 5 |
| Семестр 2 | | | | |
| 1. | Раздел 6. Подпрограммы, их назначение и классификация. Нестандартные типы данных. | Подпрограммы, их назначение и классификация. Нестандартные типы данных. | Освоение работы с процедурами и функциями | 4 |
| 2. | Раздел 7. Записи. Комбинированный тип данных. | Записи. Комбинированный тип данных. | Изучение перечисляемого, интервального типов и типа запись | 4 |

| | | | | |
|----------------------------------|--|--|---|-----------|
| 3. | Раздел 8. Динамические структуры и указатели. | Динамические структуры и указатели. | Списки: основные виды и способы реализации. Указатели. | 4 |
| 4. | Раздел 9. Основы компьютерной графики. | Основы компьютерной графики. | Освоение работы с графическими элементами | 4 |
| 5. | Раздел 10. Компьютерные сети. Internet. Защита информации. | Компьютерные сети. Internet. Защита информации. | Методы защиты информации. | 2 |
| Итого часов в 2 семестре: | | | | 18 |
| Всего: | | | | 18 |

4.3. САМОСТОЯТЕЛЬНАЯ РАБОТА ОБУЧАЮЩЕГОСЯ

| № п/п | Наименование раздела (темы) дисциплины | № п/п | Виды СРО | Всего часов |
|------------------|--|-------|---|-------------|
| | | | | ОФО |
| 1 | 2 | 3 | 4 | 5 |
| Семестр 1 | | | | |
| 1. | Раздел 1. Основные понятия информатики | 1.1. | Работа с книжными источниками | 6 |
| | | 1.2. | Работа с электронными источниками. | |
| | | 1.3. | Подготовка к текущему тестовому контролю. | |
| 2. | Раздел 2. Основы алгоритмизации | 2.1. | Проработка лекций, работа с учебниками. Подготовка к лабораторному практикуму. | 7 |
| | | 2.2. | Изучение конспекта лекций. Выполнения индивидуальных заданий по лабораторному практикуму. | |
| | | 2.3. | Работа с электронными источниками. | |
| 3. | Раздел 3. Технологии программирования | 3.1 | Работа с лекционным материалом, поиск и обзор литературы и электронных источников информации. | 7 |
| | | 3.2 | Изучение конспекта лекций для выполнения лабораторной работы. | |
| | | 3.3 | Работа с книжными источниками. | |
| | Раздел 4. | 4.1 | Подготовка к лабораторным | 7 |

| | | | | |
|----------------------------------|---|-----|---|-----------|
| 4. | Программирование на языке Паскаль. | | работам. | |
| | | 4.2 | Работа с лекционным материалом, поиск и обзор литературы и электронных источников информации. | |
| | | 4.3 | Подготовка к текущему тестовому контролю. | |
| 5. | Раздел 5. Операторы языка Паскаль. | 5.1 | Подготовка к лабораторным работам. | 7 |
| | | 5.2 | Работа с лекционным материалом, поиск и обзор литературы и электронных источников информации. | |
| | | 5.3 | Самоподготовка: внеаудиторное чтение, тестовый контроль. | |
| Итого часов в 1 семестре: | | | | 34 |
| Семестр 2 | | | | |
| 6. | Раздел 6. Подпрограммы, их назначение и классификация. Нестандартные типы данных. | 6.1 | Работа с лекционным материалом, поиск и обзор литературы и электронных источников информации. | 12 |
| | | 6.2 | Подготовка к лабораторным работам. Подготовка к текущему тестовому контролю. | |
| | | 6.3 | Самоподготовка: внеаудиторное чтение, тестовый контроль. | |
| 7. | Раздел 7. Записи. Комбинированный тип данных. | 7.1 | Подготовка к лабораторным работам. Подготовка к текущему тестовому контролю. | 8 |
| | | 7.2 | Изучение конспекта лекций для выполнения практической работы. | |
| | | 7.3 | Внеаудиторная контактная работа | |
| 8. | Раздел 8. Динамические структуры и указатели. | 8.1 | Работа с лекционным материалом, поиск и обзор литературы и электронных источников информации. | 8 |
| | | 8.2 | Самоподготовка: внеаудиторное чтение, тестовый контроль | |
| | | 8.3 | Внеаудиторная контактная работа | |
| 9. | Раздел 9. Основы компьютерной графики | 9.1 | Работа с лекционным материалом, поиск и обзор литературы и электронных источников информации. | 8 |
| | | 9.2 | Изучение конспекта лекций для выполнения практической | |

| | | | | |
|----------------------------------|--|------|---|-----------|
| | | | работы. | |
| | | 9.3 | Внеаудиторная контактная работа | |
| 10. | Раздел 10. Компьютерные сети. Internet. Защита информации. | 10.1 | Изучение конспекта лекций. Выполнения индивидуальных заданий по лабораторному практикуму. | 7 |
| | | 10.2 | Изучение конспекта лекций для выполнения практической работы. | |
| | | 10.3 | Внеаудиторная контактная работа | |
| Итого часов в 2 семестре: | | | | 43 |
| Всего: | | | | 77 |

5. ПЕРЕЧЕНЬ УЧЕБНО-МЕТОДИЧЕСКОГО ОБЕСПЕЧЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

5.1. Методические указания для подготовки обучающихся к лекционным занятиям

Лекция является основной формой обучения в высшем учебном заведении. Записи лекций в конспектах должны быть избирательными, полностью следует записывать только определения. В конспекте рекомендуется применять сокращение слов, что ускоряет запись. Вопросы, возникающие в ходе лекции, рекомендуется записывать на полях и после окончания лекции обратиться за разъяснением к преподавателю.

Работа над конспектом лекции осуществляется по этапам:

- повторить изученный материал по конспекту;
- непонятные положения отметить на полях и уточнить;
- неоконченные фразы, пропущенные слова и другие недочеты в записях устранить, пользуясь материалами из учебника и других источников;
- завершить техническое оформление конспекта (подчеркивания, выделение главного, выделение разделов, подразделов и т.п.).

Самостоятельную работу следует начинать с доработки конспекта, желательно в тот же день, пока время не стерло содержание лекции из памяти. Работа над конспектом не должна заканчиваться с прослушивания лекции. После лекции, в процессе самостоятельной работы, перед тем, как открыть тетрадь с конспектом, полезно мысленно восстановить в памяти содержание лекции, вспомнив ее структуру, основные положения и выводы.

С целью доработки необходимо прочитать записи, восстановить текст в памяти, а также исправить описки, расшифровать не принятые ранее сокращения, заполнить пропущенные места, понять текст, вникнуть в его смысл. Далее прочитать материал по рекомендуемой литературе, разрешая в ходе чтения, возникшие ранее затруднения, вопросы, а также дополнения и исправляя свои записи. Записи должны быть наглядными, для чего следует применять различные способы выделений. В ходе доработки конспекта углубляются, расширяются и закрепляются знания, а также дополняется, исправляется и совершенствуется конспект. Еще лучше, если вы переработаете конспект, дадите его в новой систематизации записей. Это, несомненно, займет некоторое время, но материал вами будет хорошо проработан, а конспективная запись его приведена в удобный для запоминания вид. Введение заголовков, скобок, обобщающих знаков может значительно повысить качество записи. Этому может служить также подчеркивание отдельных мест

конспекта красным карандашом, приведение на полях или на обратной стороне листа краткой схемы конспекта и др.

Подготовленный конспект и рекомендуемая литература используется при подготовке к практическому занятию. Подготовка сводится к внимательному прочтению учебного материала, к выводу с карандашом в руках всех утверждений и формул, к решению примеров, задач, к ответам на вопросы, предложенные в конце лекции преподавателем или помещенные в рекомендуемой литературе. Примеры, задачи, вопросы по теме являются средством самоконтроля.

Непременным условием глубокого усвоения учебного материала является знание основ, на которых строится изложение материала. Обычно преподаватель напоминает, какой ранее изученный материал и в какой степени требуется подготовить к очередному занятию. Эта рекомендация, как и требование систематической и серьезной работы над всем лекционным курсом, подлежит безусловному выполнению. Потери логической связи как внутри темы, так и между ними приводит к негативным последствиям: материал учебной дисциплины перестает основательно восприниматься, а творческий труд подменяется утомленным переписыванием. Обращение к ранее изученному материалу не только помогает восстановить в памяти известные положения, выводы, но и приводит разрозненные знания в систему, углубляет и расширяет их. Каждый возврат к старому материалу позволяет найти в нем что-то новое, переосмыслить его с иных позиций, определить для него наиболее подходящее место в уже имеющейся системе знаний. Неоднократное обращение к пройденному материалу является наиболее рациональной формой приобретения и закрепления знаний. Очень полезным, но, к сожалению, еще мало используемым в практике самостоятельной работы, является предварительное ознакомление с учебным материалом. Даже краткое, беглое знакомство с материалом очередной лекции дает многое. Обучающиеся получают общее представление о ее содержании и структуре, о главных и второстепенных вопросах, о терминах и определениях. Все это облегчает работу на лекции и делает ее целеустремленной.

5.2. Методические указания для подготовки обучающихся к лабораторным занятиям

Ведущей дидактической целью лабораторных занятий является систематизация и обобщение знаний по изучаемой теме, приобретение практических навыков по тому или другому разделу курса, закрепление полученных теоретических знаний. Лабораторные работы сопровождают и поддерживают лекционный курс. Подготовка к лабораторным занятиям и практикумам носит различный характер, как по содержанию, так и по сложности исполнения.

Многие лабораторные занятия требуют большой исследовательской работы, изучения дополнительной научной литературы. Прежде чем приступить к выполнению такой работы, обучающемуся необходимо ознакомиться обстоятельно с содержанием задания, уяснить его, оценить с точки зрения восприятия и запоминания все составляющие его компоненты. Это очень важно, так как при проработке соответствующего материала по конспекту лекции или по рекомендованной литературе могут встретиться определения, факты, пояснения, которые не относятся непосредственно к заданию. Обучающийся должен хорошо знать и понимать содержание задания, чтобы быстро оценить и отобрать нужное из читаемого. Далее, в соответствии со списком рекомендованной литературы, необходимо отыскать материал к данному заданию по всем пособиям.

Весь подобранный материал нужно хотя бы один раз прочитать или внимательно просмотреть полностью. По ходу чтения помечаются те места, в которых содержится ответ на вопрос, сформулированный в задании. Читая литературу по теме, обучающийся

должен мысленно спрашивать себя, на какой вопрос задания отвечает тот или иной абзац прорабатываемого пособия. После того, как материал для ответов подобран, желательно хотя бы мысленно, а лучше всего устно или же письменно, ответить на все вопросы. В случае если обнаружится пробел в знаниях, необходимо вновь обратиться к литературным источникам и проработать соответствующий раздел. Только после того, как преподаватель убедится, что обучающийся хорошо знает необходимый теоретический материал, что его ответы достаточно аргументированы и доказательны, можно считать обучающегося подготовленным к выполнению лабораторных работ.

5.3. Методические указания для подготовки обучающихся к практическим занятиям

В процессе подготовки и проведения практических занятий обучающиеся закрепляют полученные ранее теоретические знания, приобретают навыки их практического применения, опыт рациональной организации учебной работы.

Поскольку активность на практических занятиях является предметом внутрисеместрового контроля его продвижения в освоении курса, подготовка к таким занятиям требует ответственного отношения.

При подготовке к занятию в первую очередь должны использовать материал лекций и соответствующих литературных источников. Самоконтроль качества подготовки к каждому занятию осуществляют, проверяя свои знания и отвечая на вопросы для самопроверки по соответствующей теме.

Входной контроль осуществляется преподавателем в виде проверки и актуализации знаний обучающихся по соответствующей теме.

Выходной контроль осуществляется преподавателем проверкой качества и полноты выполнения задания.

Подготовку к практическому занятию каждый обучающийся должен начать с ознакомления с планом практического занятия, который отражает содержание предложенной темы. Тщательное продумывание и изучение вопросов плана основывается на проработке текущего материала, а затем изучение обязательной и дополнительной литературы, рекомендованной к данной теме.

Все новые понятия по изучаемой теме необходимо выучить наизусть и внести в глоссарий, который целесообразно вести с самого начала изучения курса. Результат такой работы должен проявиться в способности обучающегося свободно ответить на теоретические вопросы, его выступлении и участии в коллективном обсуждении вопросов изучаемой темы, правильном выполнении практических заданий. Предлагается следующая опорная схема подготовки к практическим занятиям.

Обучающийся при подготовке к практическому занятию может консультироваться с преподавателем и получать от него наводящие разъяснения, задания для самостоятельной работы.

1. Ознакомление с темой практического занятия. Выделение главного (основной темы) и второстепенного (подразделы, частные вопросы темы).

2. Освоение теоретического материала по теме с опорой на лекционный материал, учебник и другие учебные ресурсы. Самопроверка: постановка вопросов, затрагивающих основные термины, определения и положения по теме, и ответы на них.

3. Выполнение практического задания. Обнаружение основных трудностей, их решение с помощью дополнительных интеллектуальных усилий и/или подключения дополнительных источников информации.

4. Решение типовых заданий расчетно-графической работы.

5.4. Методические указания по самостоятельной работе обучающихся. Работа с литературными источниками и интернет ресурсами

В процессе подготовки к практическим занятиям, обучающимся необходимо обратить особое внимание на самостоятельное изучение рекомендованной учебно-методической (а также научной и популярной) литературы.

Самостоятельная работа с учебниками, учебными пособиями, научной, справочной и популярной литературой, материалами периодических изданий и Интернета, статистическими данными является наиболее эффективным методом получения знаний, позволяет значительно активизировать процесс овладения информацией, способствует более глубокому усвоению изучаемого материала, формирует у обучающихся свое отношение к конкретной проблеме.

Более глубокому раскрытию вопросов способствует знакомство с дополнительной литературой, рекомендованной преподавателем по каждой теме семинарского или практического занятия, что позволяет обучающимся проявить свою индивидуальность в рамках выступления на данных занятиях, выявить широкий спектр мнений по изучаемой проблеме.

Подготовка презентации и доклада

Для подготовки презентации рекомендуется использовать: PowerPoint, MS Word, Acrobat Reader, LaTeX-овский пакет beamer. Самая простая программа для создания презентаций – Microsoft PowerPoint. Для подготовки презентации необходимо собрать и обработать начальную информацию.

Последовательность подготовки презентации:

1. Четко сформулировать цель презентации: вы хотите свою аудиторию мотивировать, убедить, заразить какой-то идеей или просто формально отчитаться.

2. Определить каков будет формат презентации: живое выступление (тогда, сколько будет его продолжительность) или электронная рассылка (каков будет контекст презентации).

3. Отобрать всю содержательную часть для презентации и выстроить логическую цепочку представления.

4. Определить ключевые моменты в содержании текста и выделить их.

5. Определить картинки для отображения их на слайдах в соответствии с логикой, целью и спецификой материала.

6. Подобрать дизайн и форматировать слайды (количество картинок и текста, их расположение, цвет и размер).

Практические советы по подготовке презентации готовьте отдельно:

- печатный текст + слайды + раздаточный материал;
- слайды - визуальная подача информации, которая должна содержать минимум текста, максимум изображений, несущих смысловую нагрузку, выглядеть наглядно и просто;

- текстовое содержание презентации – устная речь или чтение, которая должна включать аргументы, факты, доказательства и эмоции;

- рекомендуемое число слайдов 17-22;

- обязательная информация для презентации: тема, фамилия и инициалы выступающего; план сообщения; краткие выводы из всего сказанного; список использованных источников;

- раздаточный материал – должен обеспечивать ту же глубину и охват, что и живое выступление: люди больше доверяют тому, что они могут унести с собой, чем исчезающим изображениям, слова и слайды забываются, а раздаточный материал остается постоянным осязаемым напоминанием; раздаточный материал важно раздавать в конце презентации; раздаточный материалы должны отличаться от слайдов, должны быть более информативными.

Тема доклада должна быть согласованна с преподавателем и соответствовать теме учебного занятия. Материалы при его подготовке, должны соответствовать научно-методическим требованиям вуза и быть указаны в докладе. Необходимо соблюдать регламент, оговоренный при получении задания. Иллюстрации должны быть достаточными, но не чрезмерными.

Работа обучающегося над докладом-презентацией включает отработку умения самостоятельно обобщать материал и делать выводы в заключении, умения ориентироваться в материале и отвечать на дополнительные вопросы слушателей, отработку навыков ораторства, умения проводить диспут.

Докладчики должны знать и уметь: сообщать новую информацию; использовать технические средства; хорошо ориентироваться в теме всего семинарского занятия; дискутировать и быстро отвечать на заданные вопросы; четко выполнять установленный регламент (не более 10 минут); иметь представление о композиционной структуре доклада и др.

Структура выступления

Выступление помогает обеспечить успех выступления по любой тематике. Выступление должно содержать: название, сообщение основной идеи, современную оценку предмета изложения, краткое перечисление рассматриваемых вопросов, живую интересную форму изложения, акцентирование внимания на важных моментах, оригинальность подхода.

Основная часть, в которой выступающий должен глубоко раскрыть суть затронутой темы, обычно строится по принципу отчета. Задача основной части – представить достаточно данных для того, чтобы слушатели заинтересовались темой и захотели ознакомиться с материалами. При этом логическая структура теоретического блока не должны даваться без наглядных пособий, аудио- и презентационных материалов.

Заключение – ясное, четкое обобщение и краткие выводы, которых всегда ждут слушатели.

Промежуточная аттестация

По итогам 1, 2 семестра проводится экзамен. При подготовке к сдаче экзамена рекомендуется пользоваться материалами практических занятий и материалами, изученными в ходе текущей самостоятельной работы.

Экзамены проводятся в устной форме, включает подготовку и ответы обучающегося на теоретические вопросы. По итогам экзамена выставляется оценка.

По итогам обучения проводится экзамен, к которому допускаются обучающиеся, имеющие положительные результаты по защите лабораторных работ.

6. Образовательные технологии

| № п/п | Виды учебной работы | Образовательные технологии | Всего часов |
|----------------------------------|-------------------------------------|----------------------------------|-------------|
| | | | ОФО |
| 1 | 2 | 3 | 4 |
| Семестр 1 | | | |
| 1 | Технологии программирования | Лабораторная работа, презентация | 2 |
| Итого часов в 1 семестре: | | | 2 |
| Семестр 2 | | | |
| 2. | Динамические структуры и указатели. | Лабораторная работа, презентация | 2 |
| 3. | Основы компьютерной графики | Лабораторная работа, презентация | 2 |
| Итого часов в 2 семестре: | | | 4 |
| Всего: | | | 6 |

7. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

7.1. Перечень основной и дополнительной учебной литературы

Основная литература

1. Давыдов И.С. Информатика : учебное пособие / Давыдов И.С.. — Санкт-Петербург : Проспект Науки, 2020. — 479 с. — ISBN 978-5-903090-19-8. — Текст : электронный // IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/80092.html>
2. Петров, В. Ю. Информатика. Алгоритмизация и программирование. Часть 1 : учебное пособие / В. Ю. Петров. — Санкт-Петербург : Университет ИТМО, 2016. — 93 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/66473.html>
3. Тюгашев, А. А. Основы программирования. Часть 1 : учебное пособие / А. А. Тюгашев. — Санкт-Петербург : Университет ИТМО, 2016. — 163 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/67495.html>
4. Тюгашев, А. А. Основы программирования. Часть 2 : учебное пособие / А. А. Тюгашев. — Санкт-Петербург : Университет ИТМО, 2016. — 120 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/67496.html>
5. Роганов, Е. А. Основы информатики и программирования / Е. А. Роганов. — 2-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 392 с. — ISBN 2227-8397. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <http://www.iprbookshop.ru/73689.html>

Дополнительная литература

1. Программирование на языке высокого уровня C/C++ : конспект лекций / составители С. П. Зоткин. — Москва : Московский государственный строительный университет, Ай Пи Эр Медиа, ЭБС АСВ, 2016. — 140 с. — ISBN 978-5-7264-1285-

6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/48037.html>
2. Поляков, А. Ю. Программирование : практикум / А. Ю. Поляков, А. Ю. Полякова, Е. Н. Перышкова. — Новосибирск : Сибирский государственный университет телекоммуникаций и информатики, 2015. — 55 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/55494.html>
3. Учебно-методическое пособие по дисциплине Логическое и функциональное программирование / составители М. В. Яшина, В. В. Барков, С. В. Украинский. — Москва : Московский технический университет связи и информатики, 2016. — 23 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/61490.html>
4. Букунов, С. В. Основы программирования на языке C++ : учебное пособие / С. В. Букунов. — Санкт-Петербург : Санкт-Петербургский государственный архитектурно-строительный университет, ЭБС АСВ, 2015. — 201 с. — ISBN 978-5-9227-0619-3. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/63631.html>
5. Костюкова, Н. И. Программирование на языке Си : методические рекомендации и задачи по программированию / Н. И. Костюкова. — Новосибирск : Сибирское университетское издательство, 2017. — 160 с. — ISBN 978-5-379-02016-3. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/65289.html>
6. Мухаметзянов, Р. Р. Основы программирования в Delphi : учебно-методическое пособие / Р. Р. Мухаметзянов. — Набережные Челны : Набережночелнинский государственный педагогический университет, 2017. — 137 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/66811.html>
7. Мухаметзянов, Р. Р. Основы программирования на Java : учебное пособие / Р. Р. Мухаметзянов. — Набережные Челны : Набережночелнинский государственный педагогический университет, 2017. — 114 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/66812.html>
8. Васильев, А. С. Основы программирования в среде LabVIEW : учебное пособие / А. С. Васильев, О. Ю. Лашманов. — Санкт-Петербург : Университет ИТМО, 2015. — 82 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/67494.html>
9. Программирование на языке Java : конспект лекций / А. В. Гаврилов, С. В. Клименков, А. Е. Харитонов, Е. А. Цопа. — Санкт-Петербург : Университет ИТМО, 2015. — 123 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/68692.html>
10. Основы программирования на языке Паскаль. Основные понятия алгоритмического языка Паскаль : учебное пособие для самостоятельной работы по дисциплине «Информатика» студентов 2-го курса всех направлений подготовки / составители А. Д. Кононов, А. А. Кононов. — Воронеж : Воронежский государственный технический университет, ЭБС АСВ, 2017. — 53 с. — ISBN 978-5-7731-0504-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/72929.html>
11. Иноземцева, С. А. Информатика и программирование : лабораторный практикум / С. А. Иноземцева. — Саратов : Вузовское образование, 2018. — 68 с. — ISBN 978-5-4487-0260-0. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/75691.html>

Ссылки на записи видеолекций

1. Информатика и программирование (Эльканова Л.М.) Лекция 1
<https://drive.google.com/file/d/1eKvs0ZAohbqS9beJChW6PZoyq-oL0jpp/view>
2. Информатика и программирование (Эльканова Л.М.) Лекция 2
<https://drive.google.com/file/d/13xf2wWSRoegiynRq2tyMMmmJ0AP3fs1E/view>
3. Информатика и программирование (Эльканова Л.М.) Лекция 3
<https://drive.google.com/file/d/1BOCLNawUPL4Twj4JZT8IcvM9CyVRRzsj/view>
4. Информатика и программирование (Эльканова Л.М.) Лекция 4
<https://drive.google.com/file/d/1pL9POF0ob3PAtcuS2VWhYs6kRJ7EJrb2/view>
5. Информатика и программирование (Эльканова Л.М.) Лекция 5
<https://drive.google.com/file/d/1uetHUIzSdfWyrfwl5JyAFR6twndktwbC/view>
6. Информатика и программирование (Эльканова Л.М.) Лекция 6
https://drive.google.com/file/d/1M15MoNZ2o-EILXI65FXtJVsvLsBT03B_/view

7.2. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

<http://elibrary.ru> - Научная электронная библиотека.

7.3. Информационные технологии, лицензионное программное обеспечение

| Лицензионное программное обеспечение | Реквизиты лицензий/ договоров |
|---|--|
| Microsoft Azure Dev Tools for Teaching 1. Windows 7, 8, 8.1, 10 2. Visio 2007, 2010, 2013 3. Access 2007, 2010, 2013 и т. д. | Идентификатор подписчика: 1203743421 Срок действия: 30.06.2022 (продление подписки) |
| MS Office 2003, 2007, 2010, 2013 | Сведения об Open Office: 63143487, 63321452, 64026734, 6416302, 64344172, 64394739, 64468661, 64489816, 64537893, 64563149, 64990070, 65615073 Лицензия бессрочная |
| Антивирус Dr.Web Desktop Security Suite | Лицензионный сертификат Серийный № 8DVG-V96F-H8S7-NRBC Срок действия: с 20.10.2022 до 22.10.2023 |
| Цифровой образовательный ресурс IPR SMART | Лицензионный договор № 9368/22П от 01.07.2022 г. Срок действия: с 01.07.2022 до 01.07.2023 |
| Свободное ПО: 7-Zip 9.20, Foxit Reader, Free Pascal, Lazarus, StarUML, R, RStudio, PascalABC.NET, Scilab | |

8. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

1.1. Требования к аудиториям (помещениям, местам) для проведения занятий

| | |
|---|--|
| Учебная аудитория для проведения занятий лекционного типа | Специализированная мебель: Кафедра - 1 шт., доска меловая - 1 шт., парты - 30 шт., стулья - 61 шт., Технические средства обучения, служащие для предоставления учебной информации большой аудитории: Проектор - 1 шт. Экран моторизованный - 1 шт. Ноутбук - 1 шт. |
| Учебная аудитория для проведения занятий лекционного типа | Специализированная мебель: Доска меловая - 1 шт., парты - 35 шт., стулья - 66 шт., кафедра настольная - 1 шт. Набор демонстрационного оборудования и учебно-наглядных пособий, обеспечивающих тематические иллюстрации: Настенный экран - 1 шт. Проектор - 1 шт. Ноутбук - 1 шт. |
| Лаборатория современных вычислительных систем. Лаборатория новых компьютерных технологий | Специализированная мебель: Доска меловая - 1 шт., стол компьютерный угловой преподавательский - 1 шт., стул мягкий - 1 шт., кафедра напольная - 1 шт., парты - 12 шт., компьютерные столы - 8 шт., стулья - 28 шт. Лабораторное оборудование, технические средства обучения, служащие для предоставления учебной информации большой аудитории: Экран настенный рулонный - 1 шт. Проектор - 1 шт. Персональный компьютер - 8 шт. |
| Лаборатория современных экономических информационных систем | Специализированная мебель: Парты - 6 шт., доска меловая - 1 шт., компьютерные столы - 7 шт., стол преподавательский - 3 шт., стулья - 28 шт., стол лабораторный - 3 шт. Лабораторное оборудование, технические средства обучения, служащие для предоставления учебной информации большой аудитории: Персональный компьютер - 7 шт. |
| Лаборатория компьютерной графики | Специализированная мебель: Стол преподавательский - 1 шт., компьютерные столы - 10 шт., парты - 7 шт., стулья - 24 шт., доска меловая - 1 шт. Лабораторное оборудование, технические средства обучения, служащие для предоставления учебной информации большой аудитории: Персональный компьютер - 10 шт. Экран настенный рулонный - 1 шт. |
| Учебная аудитория для проведения занятий семинарского типа, курсового проектирования (выполнение курсовых | Специализированная мебель: Стол преподавательский - 1 шт., компьютерные столы - 10 шт., парты - 7 шт., стулья - 24 шт., доска меловая - 1 шт. Лабораторное оборудование, технические средства обучения, служащие для предоставления учебной информации большой аудитории: |

| | |
|--|--|
| работ), групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации. | Персональный компьютер – 10 шт. Экран настенный рулонный – 1 шт. |
| Помещение для самостоятельной работы. | <p>Библиотечно-издательский центр. Отдел обслуживания печатными изданиями Специализированная мебель: Рабочие столы на 1 место – 21 шт. Стулья – 55 шт. Набор демонстрационного оборудования и учебно-наглядных пособий, обеспечивающих тематические иллюстрации: Экран настенный – 1 шт. Проектор – 1шт. Ноутбук – 1шт. Информационно-библиографический отдел. Специализированная мебель: Рабочие столы на 1 место - 6 шт. Стулья - 6 шт. Компьютерная техника с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду ФГБОУ ВО «СевКавГА»: Персональный компьютер – 1шт. Сканер – 1 шт. МФУ – 1 шт.</p> <p>Отдел обслуживания электронными изданиями Специализированная мебель: Рабочие столы на 1 место – 24 шт. Стулья – 24 шт. Набор демонстрационного оборудования и учебно-наглядных пособий, обеспечивающих тематические иллюстрации: Интерактивная система - 1 шт. Монитор – 21 шт. Сетевой терминал -18 шт. Персональный компьютер -3 шт. МФУ – 2 шт. Принтер –1шт.</p> |

8.2. Требования к оборудованию рабочих мест преподавателя и обучающихся

1. Рабочее место преподавателя, оснащенное компьютером с доступом в Интернет.
2. Рабочие места обучающихся, оснащенные компьютерами с доступом в Интернет, предназначенные для работы в электронной образовательной среде.

8.3. Требования к специализированному оборудованию

- нет

9. ОСОБЕННОСТИ РЕАЛИЗАЦИИ ДИСЦИПЛИНЫ ДЛЯ ИНВАЛИДОВ И ЛИЦ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ

Для обеспечения образования инвалидов и обучающихся с ограниченными возможностями здоровья разрабатывается (в случае необходимости) адаптированная образовательная программа, индивидуальный учебный план с учетом особенностей их психофизического развития и состояния здоровья, в частности применяется индивидуальный подход к освоению дисциплины, индивидуальные задания: рефераты, письменные работы и, наоборот, только устные ответы, и диалоги, индивидуальные консультации, использование диктофона и других записывающих средств для воспроизведения лекционного и семинарского материала.

В целях обеспечения обучающихся инвалидов и лиц с ограниченными возможностями здоровья комплектуется фонд основной учебной литературой, адаптированной к ограничению электронных образовательных ресурсов, доступ к которым организован в БИЦ Академии. В библиотеке проводятся индивидуальные консультации для данной категории пользователей, оказывается помощь в регистрации и использовании сетевых и локальных электронных образовательных ресурсов, предоставляются места в читальном зале.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
ПО ДИСЦИПЛИНЕ »Информатика и программирование»

1. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

«Информатика и программирование»

1. Компетенции, формируемые в процессе изучения дисциплины

| Индекс | Формулировка компетенции |
|---------------|--|
| ОПК-7 | Способен разрабатывать алгоритмы и программы, пригодные для практического применения |

2. Этапы формирования компетенции в процессе освоения дисциплины

Основными этапами формирования указанных компетенций при изучении обучающимися дисциплины являются последовательное изучение содержательно связанных между собой разделов (тем) учебных занятий. Изучение каждого раздела (темы) предполагает овладение обучающимися необходимыми компетенциями. Результат аттестации обучающихся на различных этапах формирования компетенций показывает уровень освоения компетенций обучающимся.

Этапность формирования компетенций прямо связана с местом дисциплины в образовательной программе.

| Разделы (темы) дисциплины | Формируемые компетенции (коды) |
|---|---------------------------------------|
| | ОПК-7 |
| Тема 1. Основные понятия информатики | + |
| Тема 2. Основы алгоритмизации | + |
| Тема 3. Технологии программирования | + |
| Тема 4. Программирование на языке Паскаль | + |
| Тема 5. Операторы языка Паскаль | + |
| Тема 6. Подпрограммы, их назначение и классификация | + |
| Тема 7. Записи. Комбинированный тип данных | + |
| Тема 8. Динамические структуры и указатели | + |
| Тема 9. Основы компьютерной графики | + |
| Тема 10. Компьютерные сети. Internet. Защита информации | + |

3. Показатели, критерии и средства оценивания компетенций, формируемых в процессе изучения дисциплины

ОПК-7 Способен разрабатывать алгоритмы и программы, пригодные для практического применения

| Индикаторы достижения компетенции | Критерии оценивания результатов обучения | | | | Средства оценивания результатов обучения | |
|--|--|--|--|--|--|--------------------------|
| | неудовлетв | удовлетв | хорошо | отлично | Текущий контроль | Промежуточная аттестация |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| ИДК –ОПК-7.1 Применяет языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ. | Не умеет применять знания языков программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ | Демонстрирует основы знаний основных языков программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ. | Демонстрирует основы знаний основных языков программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ. | Демонстрирует глубокие знания основных языков программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ. | Коллоквиум, устный опрос, тестирование, проверка выполнения лабораторных работ | экзамен |
| ИДК –ОПК-7.2 Программирует, отлаживает и тестирует прототипы программно-технических комплексов задач | Фрагментарное применение навыков программирования, отладки и тестирования прототипов программно-технических комплексов задач | Демонстрирует минимально необходимые навыки программирования, отладки и тестирования прототипов программно-технических комплексов задач | Демонстрирует достаточно развитые навыки программирования, отладки и тестирования прототипов программно-технических комплексов задач | Демонстрирует высокоразвитые профессиональные навыки программирования, отладки и тестирования прототипов программно-технических комплексов задач | Коллоквиум, устный опрос, тестирование, проверка выполнения лабораторных работ | экзамен |
| ИДК –ОПК-7.3 Разрабатывает алгоритмы решения прикладных задач с использованием математических и современных аналитических методов | Фрагментарное применение навыков работы при разработке алгоритмов решения прикладных задач с использованием математических и современных аналитических методов | Демонстрирует минимально необходимые навыки применения знаний при разработке алгоритмов решения прикладных задач с использованием математических и современных аналитических методов | Демонстрирует в целом успешные, но содержащее отдельные пробелы навыки применения знаний при разработке алгоритмов решения прикладных задач с использованием математических и современных аналитических методов | Демонстрирует высокоразвитые профессиональные навыки применения знаний при разработке алгоритмов решения прикладных задач с использованием математических и современных аналитических методов | Коллоквиум, устный опрос, тестирование, проверка выполнения лабораторных работ | экзамен |

4. Комплект контрольно-оценочных средств по дисциплине «Информатика и программирование»

Лабораторный практикум По дисциплине Информатика и программирование

Лабораторная работа 1

Тема: Основные понятия информатики

(Определение энтропии по формуле Шеннона. Мера Хартли. Свойства энтропии дискретных и непрерывных сообщений. Единицы измерения количества информации).

Цель работы. Научиться практически определять количество информации в различного вида дискретных сообщениях.

Теоретическое обоснование. Количество информации, содержащееся в дискретном сообщении (I) можно найти из простого соотношения

$$I = n \cdot H,$$

где n — число символов в сообщении,

H — энтропия источника сообщений, то есть среднее количество информации, приходящееся на один символ сообщения.

Энтропия источника сообщения определяется из основного соотношения теории информации (1.4), которое для удобства практического использования преобразуется к виду наиболее простому и удобному в зависимости от свойств дискретного источника сообщений.

В случае, если символы источника сообщения появляются равновероятно и взаимно независимо, то для подсчета энтропии такого рода сообщений используют формулу Хартли:

$$I = n \cdot \log_2 m (\text{бит}); \quad H_1 = \log_2 m (\text{бит/символ}),$$

где m — объем алфавита источника дискретных сообщений.

Если же символы источника сообщения генерируются с различными вероятностями, но взаимно независимы, то используют формулу Шеннона:

$$I = -n \cdot \sum_{i=1}^m P_{ai} \cdot \log_2 P_{ai} (\text{бит}),$$
$$H_2 = -\sum_{i=1}^m P_{ai} \cdot \log_2 P_{ai} (\text{бит/символ}),$$

где P_{ai} — вероятность появления символа a_i .

В случае же неравновероятного появления символов источника сообщения и наличия статистических зависимостей между соседними символами энтропию такого рода источника можно определить с помощью формулы Шеннона с условными вероятностями:

$$H_2 = -\sum_{i=1}^m P_{ai} \cdot \sum_{j=1}^m P\left(\frac{a_j}{a_i}\right) \cdot \log_2 P\left(\frac{a_j}{a_i}\right) (\text{бит/символ}),$$

где $P\left(\frac{a_j}{a_i}\right)$ — условная вероятность появления символа a_j после символа a_i .

Содержание работы.

1. Посчитать среднее количество информации, приходящееся на один символ источника дискретных сообщений (энтропию) в случаях:

а — равновероятного и взаимно независимого появления символов;

б — неравновероятного и взаимно независимого появления символов;

в — при неравновероятном появлении символов и наличии статистических связей между соседними символами.

В качестве дискретного источника сообщений взять источник с объемом алфавита $m = 34$ (аналогичный по объему алфавита тексту на русском языке: 33 буквы и пробел), а его статистические характеристики смоделировать с помощью генератора случайных чисел.

2. Подсчитать количество информации в сообщении, представляющим собой Вашу фамилию, имя и отчество, считая, что символы сообщения появляются неравновероятно и независимо. Закон распределения символов найти путем анализа участка любого текста на русском языке длиной не менее 300 символов.

Выполнение работы. Работа выполняется на персональном компьютере в программном средстве «Mathcad». Так как в этом программном продукте в качестве встроенных функций используются только функции натуральных и десятичных логарифмов, то в процессе выполнения работы необходимо выполнить переход к логарифмам по основанию 2 по формуле перехода к иному основанию:

$$\log_b N = \frac{\log_a N}{\log_a b},$$

где a — основание известных логарифмов;
 b — основание требуемых логарифмов;
 N — логарифмируемая величина.

П.1.а. Используя формулу Хартли, найти энтропию указанного источника дискретных сообщений (H_1).

П.1.б. Смоделировать закон распределения символов дискретного источника сообщений, используя оператор $\text{rnd}(A)$, который генерирует случайные числа из диапазона $[0, A]$ по следующей программе:

$m := 34$ — задание объема алфавита (m);
 $i := 1, 2, \dots, m$ — i - порядковый номер символа алфавита;
 $r(i) := \text{rnd}(1)$ — генерирование 34 случайных чисел в интервале от 0 до 1;
 $l := \sum_i r(i)$ — нахождение суммы всех $r(i)$;
 $P(i) := \frac{r(i)}{l}$ — $P(i)$ – вероятность появления i -го символа (a_i).

Проверить правильность вычислений, найдя сумму всех $P(i)$ при $i = 1, 2, \dots, m$.

Построить график закона распределения $P(i)$ Используя формулу Шеннона, определить энтропию смоделированного источника дискретных сообщений (H_2).

П.1.в. Смоделировать матрицу условных вероятностей появления символа a_j после символа a_i по следующей программе:

$m := 34$ — задание объема алфавита (m);
 $i := 1, 2, \dots, m$ }
 $j := 1, 2, \dots, m$ } — порядковый номер символа алфавита;
 $r(i, j) := \text{rnd}(1)$ — генерирование матрицы (34×34) случайных чисел в интервале от 0 до 1;
 $W_i := \sum_j r(i, j)$ — нахождение суммы элементов в каждой строке матрицы $r(i, j)$;
 $S(i, j) := \frac{r(i, j)}{W_i}$ — нормировка по строкам матрицы $r(i, j)$ с целью получения суммы

элементов в каждой строке, равной 1;

$U_j := \sum_i S(i, j)$ — нахождение сумм элементов в каждом столбце матрицы $S(i, j)$;

$PP(i, j) := \frac{S(i, j)}{U_j}$ — нормировка по столбцам матрицы $S(i, j)$ с целью получения суммы

элементов в каждом столбце равной 1.

Полученные значения элементов матрицы $PP(i, j)$ приближенно можно считать условными вероятностями появления символа под номером j после i -го символа.

Используя формулу Шеннона с условными вероятностями определить энтропию смоделированного источника дискретных сообщений (H_3).

П.2. Определить вероятность появления каждого символа (буквы) P_i путем деления числа появлений этого символа (a_i) на общее число символов (не менее 300), входящих в сообщение. В случае, если какой-либо символ (из $m = 34$) в сообщении не встретился,

считать, что он встретился 1 раз, иначе может возникнуть неопределенность в формуле Шеннона. Отсутствие в исследуемом сообщении какого-либо символа из состава алфавита источника сообщений свидетельствует лишь о том, что анализируемое сообщение не содержит достаточного числа символов (не достаточно длинное), чтобы появились все символы входящие в алфавит.

Построить график закона распределения символов (букв).

Проверить правильность полученного закона распределения, для чего найти сумму вероятностей появления каждого символа. Эта сумма должна быть равна 1.

С помощью формулы Шеннона найти энтропию (H_4) дискретного источника (текста на русском языке). Подсчитав число символов в Вашей фамилии, имени и отчестве (включая пробелы), найти количество информации, содержащейся в этом сообщении.

Контрольные вопросы.

1. Какие источники сообщений называют дискретными?
2. Для каких источников дискретных сообщений применимы формулы Хартли, Шеннона?
3. Каким образом описывается статистическая зависимость между соседними символами в дискретных сообщениях?
4. Дайте определение энтропии источника дискретных сообщений.
5. Как проверить правильность нахождения закона распределения символов источника дискретных сообщений?
6. Какой вид дискретных сообщений обладает наибольшей энтропией?

Лабораторная работа 2

Тема: Основы алгоритмизации

(Основные этапы разработки алгоритмов: постановка задачи, построение математической модели, разработка алгоритма решения задачи, проверка правильности и оценка сложности алгоритма).

Теоретическое обоснование.

При использовании различных алгоритмов для решения практических, в частности экономических, задач непременно применяют и ряд сопутствующих информационных технологий: методы и средства моделирования, системы программирования, методы анализа и т.д. С прикладной точки зрения алгоритм — это не просто абстрактное описание действий, а некоторая программа (часть программного кода), реализованная на некотором языке программирования (либо в пакете прикладных программ) и представленная в виде программного продукта, надстройки, дополнения и т.д.

Опыт практической разработки алгоритмов показал, что при создании, программировании и дальнейшем сопровождении программных продуктов, реализующих алгоритмы, следует придерживаться определенной схемы действий.

Построение алгоритма включает ряд следующих этапов:

- 1) постановка задачи;
- 2) построение математической модели;
- 3) разработка алгоритма;
- 4) проверка правильности алгоритма;
- 5) программирование (реализация алгоритма);
- 6) анализ алгоритма и его сложности;
- 7) проверка (отладка) программы;
- 8) составление документации.

Состав этапов и порядок их выполнения является условным. Некоторые этапы могут вообще не выполняться либо выполняться одновременно. Особенно это относится к построению простых алгоритмов, когда результаты определенного этапа тривиальны. При решении сложных задач, напротив, некоторые этапы выполняются многократно, производится анализ их результатов и соответствующая корректировка, порядок следования может быть другим.

Для несложных задач процесс построения и реализации алгоритма может быть представлен в виде схемы, приведенной на рис.

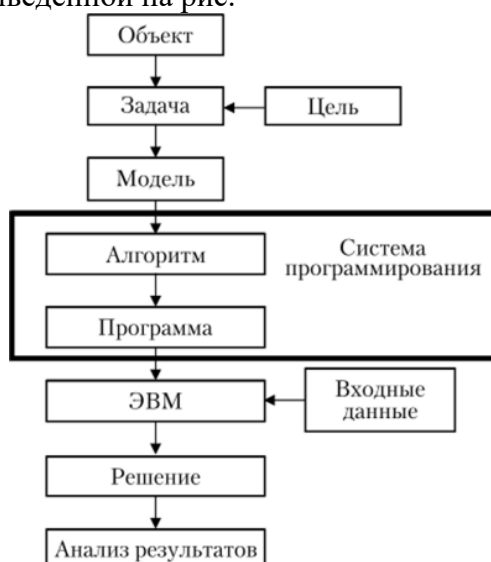


Рис.. Общая схема построения и реализации алгоритма

Ниже приведено описание каждого этапа построения алгоритма.

Постановка задачи.

Суть данного этапа интуитивно можно понимать как точное формулирование задачи, включающее в себя формулирование цели и описание в понятных разработчику (или другим заинтересованным лицам) терминах.

Под целью понимают идеализированное представление о результате работы программы (алгоритма). Для того чтобы достичь цели (или, по крайней мере, приблизиться к ней), формулируется ряд задач, которые можно понимать как укрупненные алгоритмические блоки.

Кроме цели и задач обычно определяются дополнительные вопросы, ответы на которые позволяют более точно поставить задачу. Например: «Что дано и что нужно найти?», «Какие сделаны допущения в формулировке задачи?», «Каких данных не хватает, или, наоборот, все ли перечисленные данные используются?».

Этап постановки задачи является одним из важнейших. Допущенные ошибки или неточности обычно приводят к временным задержкам на других этапах, возвращению к уже пройденным этапам или получению некорректных результатов. Считается, что на постановку сложных задач должно тратиться 40—50% всего времени построения и реализации алгоритма.

Лабораторная работа 3

Тема: Технологии программирования

(Структура алгоритмического языка. Понятие синтаксиса, семантики, прагматики и лексики).

Теоретическое обоснование

Языки программирования – это формальные языки, специально созданные для общения человека с вычислительной машиной.

Каждый язык программирования, равно как и «естественный язык» (русский, английский и т.д.) имеет алфавит, словарный запас, свою грамматику и синтаксис, а также семантику.

Задание 1. Найдите определения понятий «алфавит», «синтаксис» и «семантика». С точки зрения темы «Языки программирования»!

Существуют различные **классификации языков программирования**: с точки зрения уровня языка, с точки зрения вида обрабатываемой информации, с точки зрения поколения языка.

Классификация 1. Классификация по уровню.

1. *Языки программирования низкого уровня.* Это машинные языки и языки символического кодирования. Наборы операторов и изобразительные средства зависят от особенностей ЭВМ.

2. *Языки программирования высокого уровня.* Это машинно-независимые языки (т.к. ориентированы не на систему команд той или иной ЭВМ, а на систему операндов, характерных для записи определенного класса алгоритмов).

3. *Языки программирования сверхвысокого уровня.* Повышение уровня этих языков произошло за счет введения сверхмощных операций и операторов.

Классификация 2. Классификация по виду обрабатываемой информации.

1. *Вычислительные языки.*

2. *Языки символьной обработки.*

Классификация 3. Классификация по поколению.

1. *языки первого поколения:* машинно–ориентированные с ручным управлением памяти на компьютерах первого поколения.

2. *языки второго поколения:* с мнемоническим представлением команд, так называемые автокоды.

3. *языки третьего поколения:* общего назначения, используемые для создания прикладных программ любого типа.

4. *языки четвертого поколения:* усовершенствованные, разработанные для создания специальных прикладных программ, для управления базами данных.

5. *языки программирования пятого поколения:* языки декларативные, объектно–ориентированные и визуальные.

Кроме поколений языков выделяют **направления развития языков программирования.**

В современной информатике существуют два **основных направления развития языков программирования**: процедурное и непроцедурное.

1. Процедурное программирование возникло на заре вычислительной техники и получило широкое распространение. В процедурных языках программа явно описывает действия, которые необходимо выполнить, а результат задается только способом получения его при помощи некоторой процедуры, которая представляет собой определенную последовательность действий.

Среди процедурных языков выделяют в свою очередь:

○ **Структурные языки.** В структурных языках одним оператором записываются целые алгоритмические структуры: ветвления, циклы и т.д.

○ **Операционные языки.** В операционных языках для этого используются несколько операций.

2. Непроцедурное (декларативное) программирование появилось в начале 70-х годов 20 века, но стремительное его развитие началось в 80-е годы, когда был разработан японский проект создания ЭВМ пятого поколения, целью которого явилась подготовка почвы для создания интеллектуальных машин.

К непроцедурному программированию относятся:

○ **Функциональные языки.** В функциональных языках программа описывает вычисление некоторой функции. Обычно эта функция задается как композиция других, более простых, те в свою очередь разлагаются на еще более простые задачи и т.д. Один из основных элементов функциональных языков - рекурсия. Присваивания и циклов в классических функциональных языках нет.

○ **Логические языки.** В логических языках программа вообще не описывает действий. Она задает данные и соотношения между ними. После этого системе можно задавать вопросы. Машина перебирает известные и заданные в программе данные и находит ответ на вопрос. Порядок перебора не описывается в программе, а неявно задается самим языком. Построение логической программы вообще не требует алгоритмического мышления, программа описывает статические отношения объектов, а динамика находится в механизме перебора и скрыта от программиста.

Можно выделить еще один класс языков программирования – **объектно-ориентированные языки высокого уровня.** На таких языках не описывают подробной последовательности действий для решения задачи, хотя они содержат элементы процедурного программирования. Объектно-ориентированные языки, **благодаря богатому пользовательскому интерфейсу, предлагают человеку решить задачу в удобной для него форме.**

Языки описания сценариев предполагают стиль программирования, весьма отличный от характерного для языков системного уровня. Они **предназначаются не для написания приложения с нуля, а для комбинирования компонентов, набор которых создается заранее при помощи других языков.** Развитие и рост популярности Internet также способствовали распространению языков описания сценариев.

Задание 2.

Заполните предложенную форму классификации языков программирования. Напишите не менее трех примеров названий языков программирования в каждом пункте.

Лабораторная работа 4

Тема: Программирование на языке Паскаль.

(Назначение алгоритмического языка Паскаль. Основные символы языка.

Простейшие конструкции. Структура программного модуля. Классификация операторов.

Синтаксис языка Паскаль. Типы данных. Алгебраические и логические операции, математические функции. Управляющие конструкции языка Паскаль. Массивы и записи в Паскале. Процедуры и функции в Паскале.).

Теоретическое обоснование

Система программирования PASCAL состоит из совокупности системных программ, предназначенных для создания, отладки и выполнения Паскаль-программ.

В эту систему входят следующие части:

-текстовый редактор;

-компилятор;

-загрузчик.

PASCAL запускается программой turbo.exe

После успешного вызова системы на верхней строке экрана появляется строка главного меню. Нижняя строка экрана содержит краткую справку о назначении функциональных клавиш. Центральная часть экрана – окно редактора для ввода и редактирования текста программы.

Основные пункты главного меню: FILE, EDIT, COMPILE, RUN.

FILE - работа с файлами .

Подпункты этого пункта:

New- создание нового файла

Open–Загрузка существующего файла

Save – сохранить

Save as –сохранить как

Exit – выход из системы

EDIT-обращение к текстовому редактору

COMPILE - компиляция

RUN или CTRL+F9- выполнение

Для активизации меню используется клавиша F10.

Обработка программы написанной на языке PASCAL проходит 3 этапа: создание текста программы, компиляцию, выполнение программы.

Для создания файла выполняем: FILE => New. Появляется окно для редактирования текста программы.

После ввода текста программы нужно произвести компиляцию – перевод исходной программы в машинные коды. При этом проверяется соответствие программы правилам языка программирования (синтаксический и семантический контроль). При обнаружении ошибки компьютер выдает сообщение о ней, и указывает место ошибки. Ошибку нужно исправить и заново компилировать программу.

Компиляция инициируется системной командой COMPILER

Исполнение откомпилированной программы производится по команде RUN.

Переход в окно пользователя ALT+F5

Сохранение файла FILE => Save

Сохранение файла с именем FILE => Save as

Задание. Набрать программу и выполнить:

```
PROGRAM P1;  
  { Это моя первая программа }  
BEGIN  
  WRITELN('*****');  
  WRITELN('Моя первая программа');  
  WRITELN('*****');  
END.
```

ПРОСТЕЙШИЕ КОНСТРУКЦИИ ЯЗЫКА ПАСКАЛЬ.

Программа на языке Паскаль состоит из операторов и комментариев. Операторы состояются из простейших конструкций. К ним относятся константы, переменные, массивы, функции, выражения и служебные слова. Для записи операторов языка используют:

все латинские буквы от A до Z (прописные) и от a до z (строчные), а также знак подчеркивания;

все арабские цифры от 0 до 9;

шестнадцатеричные цифры: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F;

специальные знаки: + - * / ^ < > [] . () : ; { } @ \$ #

Следующие комбинации специальных символов являются единичными символами (их нельзя разделять пробелами) :

:= знак присваивания;

>= больше или равно;

<= меньше или равно;

<> не равно.

Буквы русского алфавита употребляются только в комментариях и текстовых константах.

Идентификаторы. Идентификатором называется символическое имя определенного программного объекта. Такими объектами являются имена констант, переменных, типов данных, процедур и функций, программ. Идентификатор - это любая последовательность букв и цифр, начинающаяся с буквы. Строчные и прописные буквы в идентификаторах и служебных словах не различаются. Длина идентификатора может быть произвольной, но значащими являются только первые 63 символа.

Комментарии. Комментарии игнорируются компилятором и на работу программы не влияют. Это любой текст, заключенный в скобки { } или (* *).

1. Константы.

В языке Паскаль в качестве данных используются целые, действительные, логические, шестнадцатеричные и текстовые (литеральные) константы.

Целые константы представляет собой последовательность цифр, перед которой может стоять знак плюс или минус. *Например:* 0; 256, -56, +845, 65849.

На языке Паскаль можно использовать целые константы различных типов. Тип **Integer** занимает 2 байта памяти, диапазон значений -32768...32767. Тип **Byte** (1 байт, 0...255), тип **Word** (2 байта, 0...65535), тип **Shortint** (1 байт, -128... 127), тип **Longint** (4 байта, $-2^{31}...2^{31}-1$).

Целые шестнадцатеричные константы записываются с префиксом \$. Они должны находиться в диапазоне от \$00000000 до \$ FFFFFFFF.

Действительные константы - числа, имеющие целую и дробную части. Они представляются в машине приближенно. Допускается запись действительных констант с **фиксированной точкой (в основной форме) и с плавающей точкой (в форме с порядком).**

В основной форме константа имеет целую часть, отделенную точкой от дробной части. Нулевая или дробная часть константы может быть опущена. *Например:* -.256, 56.255, +556.44, -26.0.

Действительная константа с плавающей точкой имеет следующую форму: <мантисса>E<порядок>, где мантисса - целое или действительное число с фиксированной точкой, порядок - целое число со знаком или без.

Например: 0.9E+5; 48E-2; 56E5; -45.2E-6; 8E+2.

Типы действительных констант: **Real** (6 байтов, $2,9*10^{39}... 1,7*10^{38}$), **Double** (8 байтов, $5*10^{324}... 1,7*10^{308}$), **Extended** (10 байтов, $3,4*10^{4932}... 1,1*10^{4932}$).

Логические константы имеют одно из двух значений: **TRUE** -истинно, **FALSE** - ложно. Тип **Boolean**.

Символьная константа - любой символ алфавита, заключенный в апострофы, например, 'W' , 't ' , '8'. **Строковая константа** это последовательность любых алфавитно-цифровых и специальных символов, имеющая длину, не превышающую 255 символов, заключенных в апострофы. *Например:* ' Решить уравнение Y=A*D'. Константе может быть поставлено в соответствие определенное имя.

2. Переменные. Простые переменные в каждый момент времени имеют одно значение, которое хранится в одной ячейке памяти. Простые переменные обозначается с помощью идентификатора.

Переменные с индексами являются элементами массивов. Массив – это упорядоченная последовательность данных одного типа. Члены этой последовательности называются элементами массива. Для хранения каждого элемента массива отводится своя ячейка. Элемент массива обозначается идентификатором массива, за которым в квадратных скобках, через запятую записываются от одного до семи индексов. Индекс представляется константой, переменной или выражением и должен быть целочисленным значением. *Например:* D[5] - означает, что это пятый элемент массива D;

A[1] - означает 1-й элемент массива A;

X[5,7] - означает элемент пятой строки, седьмого столбца матрицы X;

Z[4,2*N-1] - означает элемент четвертой строки, (2*N-1)-го столбца.

Описание типа переменных. Переменная может получать значение любой константы (целой, действительной, логической или текстовой). Указание типа переменной осуществляется с помощью описаний.

Переменные описываются в разделе описания переменных.

Описание простых переменных:

VAR <имя переменной>: <тип переменной>;

Описание массива:

VAR <имя массива>: **ARRAY**[диапазоны изменения индексов] **OF** <тип элементов>;

Пример.

VAR K,L,Z: INTEGER; B1,B2: BOOLEAN;

A: ARRAY[1..5] OF REAL;

X,Y: ARRAY[1..4,1..5] OF REAL;

Арифметические операции. К арифметическим типам данных относятся группы действительных и целых типов. К ним применимы арифметические операции и операции отношений. Операции над данными бывают унарными (применимы к одному операнду) и бинарными применимые к двум операндам). Унарная арифметическая операция одна. Это операция изменения знака.

Таблица 1. Бинарные арифметические операции Паскаля (I обозначает целые типы, R - вещественные).

| Знак | Выражение | Типы операндов | Тип результата | Операция |
|------|-----------|---------------------------|----------------|---------------------------|
| + | A+B | R,R I, I I, R; R, I | R I R | Сложение |
| - | A-B | R,R I, I I, R; R, I | R I R | Вычитание |
| * | A*B | R,R I, I I, R; R, I | R I R | Умножение |
| / | A/B | R,R I, I I, R; R, I | R R R | Вещественное деление |
| Div | A div B | I, I | I | Целое деление |
| mod | A mod B | I, I | I | Остаток от целого деления |

К арифметическим величинам могут быть применены стандартные функции Паскаля. Функция выступает как операнд в выражении. Аргументы являются в общем случае выражениями арифметического типа. Аргументы записываются в круглых скобках.

Таблица 2. Стандартные математические функции Турбо Паскаля.

| Обращение | Тип аргумент а | Тип результата | Функция |
|-----------|----------------|----------------|---|
| Pi | | R | Число $\pi=3.141592$ |
| Abs(x) | I,R | I,R | Модуль аргумента x |
| Arctan(x) | I,R | R | Арктангенс x (радианы) |
| Cos(x) | I,R | R | Косинус x (x в радианах) |
| Exp(x) | I,R | R | e^x - экспонента |
| Frac(x) | I,R | R | Дробная часть x |
| Int(x) | I,R | R | Целая часть x |
| Ln(x) | I,R | R | Натуральный логарифм x |
| Random | | R | Псевдослучайное число в интервале [0,1) |
| Random(x) | I | I | Псевдослучайное число в интервале [0,x) |
| Round(x) | R | I | Округление до ближайшего целого |
| Sin(x) | I,R | R | Синус x (x в радианах) |
| Sqr(x) | I,R | I,R | Квадрат x |

| | | | |
|----------|-----|---|---|
| Sqrt(x) | I,R | R | Корень квадратный из x |
| Trunc(x) | R | I | Ближайшее целое, не превышающее x по модулю |

При необходимости вычисления некоторых математических функций, для которых не существуют стандартных функций в языке Турбо Паскаль, их выражают через имеющиеся стандартные функции.

Например:

$$\text{Tg}(x) = \sin(x) / \cos(x)$$

$$\text{Lg}(x) = \ln(x) / \ln(10)$$

$$X^n = \text{Exp}(n * \ln(x)).$$

Результатом вычисления значения арифметического выражения является числовая константа.

Порядок выполнения операций в выражении задается скобками. При отсутствии скобок операции выполняются в порядке старшинства:

- вычисление функции;
- возведение в степень;
- умножение, деление, div или mod;
- сложение или вычитание.

Операции одного порядка выполняются последовательно слева направо.

Логическое выражение. Простейшее логическое выражение - отношение вида A&B, где & - знак операции отношения: > - больше; >= - больше либо равно; < - меньше ; <= - меньше либо равно; = - равно; <>- не равно.

A и B - арифметические выражения целого или действительного типа. Более сложные логические выражения строятся из логических констант, переменных, отношений с помощью знаков логических операций:

NOT - операция логического отрицания ; □

AND - операция логического умножения; □

OR - операция логического сложения. ∨

Логическое выражение может принимать значение или **TRUE** или **FALSE**. Порядок приоритета логических операций следующий: сначала вычисляется значение отношений, затем операции **NOT** , далее **AND** и, наконец, операция **OR**. Значения результатов логических операций приведены в таблице 3

Таблица 3

| | | | | |
|--------------|-------|-------|-------|-------|
| A | TRUE | TRUE | FALSE | FALSE |
| B | TRUE | FALSE | TRUE | FALSE |
| NOT A | FALSE | FALSE | TRUE | TRUE |
| A AND B | TRUE | FALSE | FALSE | FALSE |
| A OR B | TRUE | TRUE | TRUE | FALSE |

Функции, связывающие различные типы данных

Таблица 4

| Обращение | Тип аргумента | Тип результата | Действие |
|-----------|------------------|-------------------|---|
| Ord(x) | Любой порядковый | I | Порядковый номер значения X в его типе |
| Pred (x) | Любой порядковый | Тот же, что для X | Предыдущее относительно X значение в его типе |
| Succ(x) | Любой порядковый | Тот же, что для X | Следующее относительно X значение в его типе |
| Chr(x) | byte | Char | Символ с порядковым номером X |
| Odd(x) | I | Boolean | TRUE,если X нечетное; FALSE,если X четное |

Варианты заданий

Задание 1. Представить приведенные ниже

Задание 2. Представить приведенные

| числа в виде вещественных констант без экспонент. | ниже числа в виде вещественных констант с экспонентами. |
|---|--|
| 1) -10^3 2) 10^5 3) $211,02 \cdot 10^4$ 4) $314 \cdot 2^6$ 5) $256 \cdot 2^6$ 6) $0,26 \cdot 10^{-8}$ 7) 10^{-9} 8) $99,9$ 9) $-0,0556$ 10) $-16,301$ 11) $-39,64362802$ 12) $1,1$ 13) -0 14) $646362,7670021$ 15) $0,00345 \cdot 10^4$ | 1) $0,2360027$ 2) -10^{-4} 3) 164273 4) $-0,0345$ 5) $-0,0556$ 6) $-39,64362802$ 7) $211,02 \cdot 10^{-2}$ 8) $0,26 \cdot 10^8$ 9) $867342,17 \cdot 10^3$ 10) $0,00053$ 11) $-3,1 \cdot 10^6$ 12) $-715,1 \cdot 10^{-3}$ 13) $26,345 \cdot 10^{-4}$ 14) $-0,0043$ 15) $4,3 \cdot 10^{-5}$ |
| Задание 3. Представить приведенные ниже числа в виде вещественных констант без экспонент. | |
| 1) $4.3E-3$ 2) $-.4364E-02$ 3) $7.46E+01$ 4) $7.46>E1$ 5) $0.7425E+02$ 6) $0.0043E1$ 7) $15.23E-03$ 8) $74.63E-03$ 9) $9.909E-02$ 10) $-300.1E03$ 11) $1000.8E-02$ 12) $24.58E+01$ 13) $17.234E-03$ 14) $145.55E-04$ 15) $3.141E-06$ | |
| Задание 4. Записать арифметические выражения на языке Паскаль | |
| 1. $Y = \frac{\sqrt{a^2 + c^2} - a - b}{2}$ 2. $Z = \frac{a}{c} \times \frac{b}{d} - \frac{ab - c}{cd}$; 3. $Z = \frac{\sin \cos}{\cos \sin} \cdot \text{tg}$. 4. $Z = \frac{x+y}{y+1} - \frac{xy+1}{34x}$; 5. $Z = \frac{3te^4}{14y^2 + 8}$; 6. $A = x - \frac{x^3}{3} + \frac{x^5}{5}$; 7. $Z = 2 \cdot \frac{1}{1 + x^2}$; 8. $Z = (1 + \text{tg} x) \cdot \text{tg}(\cos)$ | 9. $Z = \ln \left((y - \sqrt{x}) \cdot x - \frac{y}{x + \frac{x^2}{4}} \right)$; 10. $Z = \frac{\ln \cos }{\ln(1+x^2)}$; 11. $Z = \left(\frac{x+1}{x-1} \right)^x + 1.8x$; 12. $Z = \left(1 + \frac{1}{x} \right)^x - 1.8x$; 13. $Z = \frac{x^2 - 7x + 10}{x^2 - 8x + 12}$; 14. $Z = \frac{\cos}{\pi x} - \ln(\cos)$; 15. $Z = 2^x - \cos(2x)$ |

Лабораторная работа № 5

ТЕМА: Разработка линейных алгоритмов.

Алгоритм линейной структуры (линейный алгоритм) - алгоритм, в котором блоки выполняются последовательно друг за другом. Такой порядок выполнения блоков называется естественным.

Самым простым действием над переменной является занесение в нее величины соответствующего типа. Иногда говорят об этом, как о присвоении переменной конкретного значения. Такая команда (оператор) в общем виде выглядит на языке Паскаль следующим образом:

<Имя переменной> := <Выражение>;

Выражение, указанное справа от знака ":", должно приводить к значению того же типа, какого и сама переменная, или типа, совместимого с переменной относительно команды присваивания. Например, переменной типа Real можно присвоить значение типа Integer или Word (впрочем, наоборот делать нельзя). Выражение будет сначала вычислено, затем, его результат будет положен в ячейки памяти, отведенные для переменной.

Операторы ввода и вывода информации

Операторы ввода (форматы операторов):

```
Read(<Список ввода>);  
Readln(<Список ввода>);
```

В таком формате эти команды позволяют вводить данные в переменные во время выполнения программы с клавиатуры. Элементами списка ввода могут быть имена переменных, которые должны быть заполнены значениями, введенными с клавиатуры.

Выполнение операторов ввода происходит так: ход программы приостанавливается, на экран выводится курсор, компьютер ожидает от пользователя набора данных для переменных, имена которых указаны в списке ввода. Пользователь с клавиатуры вводит необходимые значения в том порядке, в котором они требуются списком ввода, нажимает Enter. После этого набранные данные попадают в соответствующие им переменные и выполнение программы продолжается.

Примечание: данные при вводе разделяются пробелами.

Разница между работой процедур Read и Readln (от Read line) состоит в следующем: после выполнения Read значение следующего данного считывается с этой же строки, а после выполнения Readln - с новой строки.

Для вывода информации в Паскале также есть две команды:
Write(<Список вывода>);
Writeln(<Список вывода>);

Такой формат использования Write и Writeln позволяет выводить на экран монитора данные из списка вывода. Элементами списка вывода могут являться имена переменных, выражения, константы. Прежде чем вывести на экран компьютер значения выражений сначала вычислит. Элементы списка, также как и в операторах ввода, разделяются запятыми.

Различие между двумя операторами вывода таково: после выполнения оператора Writeln (от Write line) происходит переход на новую строку, а после выполнения инструкции Write, переход на новую строку не происходит и печать по последующим командам вывода Write или Writeln будет происходить на той же строке. При вызове оператора Writeln без параметров просто происходит переход на новую строку.

Управление символьным выводом на экран.

Дополнительные возможности управления выводом на экран дают процедуры и функции модуля CRT.

Для установления связи пользовательской программы с модулем перед разделом описаний нужно записать :

```
Uses CRT;
```

Для работы с модулем CRT необходимо ознакомиться со следующими понятиями: режимы экрана, координаты на экране, текстовое окно, цвет фона и цвет символа.

Режимы экрана. Вывод на экран может происходить в текстовом или графическом виде. Текстовые режимы различаются по количеству символьных строк и столбцов, вмещающихся на экране.

В модуле CRT каждый режим имеет определенный номер, за которым закреплено символическое имя. Для установки режима экрана используется процедура

```
TextMode(<номер режима>);
```

Например,

```
TextMode(CO80);
```

Координаты позиции. Каждая символьная позиция на текстовом экране определена двумя координатами (X,Y). Координата X- позиция в строке. Для левой крайней позиции в строке X=1. Координата Y -номер строки, в которой стоит символ. Строки нумеруются сверху вниз.

Для установления курсора на экране в позицию с координатами (X,Y) в модуле CRT существует процедура:

```
GoToXY(X,Y);
```

Процедура очистки экрана:

```
ClrScr;
```

Процедура назначения цвета фона:

```
TextBackGround(Color);
```

Процедура назначения цвета символа:

```
TextColor(Color);
```

Для организации задержки окна результатов на экране до нажатия какой-либо клавиши, перед концом программы записывают следующий оператор:

```
Repeat Until KeyPressed;
```

Пример 1. Составить блок-схему и программу для вычисления значения функции по формуле $Z = \ln|\cos X| - \exp(-X \ln 2) + \sin(2XY)$

Решение:

```
PROGRAM P1;  
VAR X,Y,Z: REAL;  
BEGIN  
WRITELN ('ВВЕДИТЕ X, Y' );  
READLN (X, Y);  
Z:=LN(ABS(COS(X)))-EXP(-  
X*LN(2))+SIN(2*X*Y)  
WRITELN ( 'Z=', Z);  
  
END.
```

Пример 2. Заданы два вектора с координатами $\vec{a}_1 = (x_1, y_1, z_1)$ и $\vec{a}_2 = (x_2, y_2, z_2)$. Определить косинус угла между векторами.

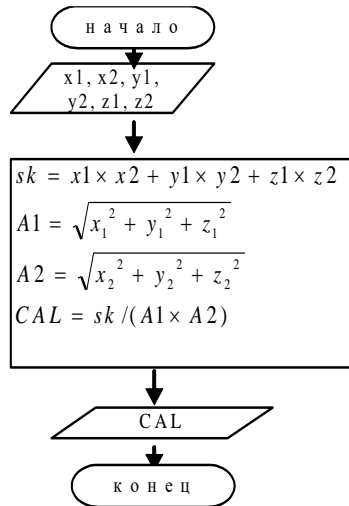
Решение: Косинус угла между двумя векторами \vec{a}_1, \vec{a}_2 определяется по формуле:

$$\cos \alpha = \frac{(\vec{a}_1, \vec{a}_2)}{|\vec{a}_1| |\vec{a}_2|},$$

где (\vec{a}_1, \vec{a}_2) - скалярное произведение,

$|\vec{a}_1| = \sqrt{x_1^2 + y_1^2 + z_1^2}$, $|\vec{a}_2| = \sqrt{x_2^2 + y_2^2 + z_2^2}$ - модули векторов. Введем обозначение:

$$sk = (\vec{a}_1, \vec{a}_2) \quad A1 = |\vec{a}_1|, \quad A2 = |\vec{a}_2|, \quad CAL = \cos \alpha.$$



PROGRAMM P2;

VAR X1, Y1, Z1, X2, Y2, Z2:REAL;

A1, A2, SK, CAL, ALFA: REAL;

BEGIN

WRITELN ('X1, Y1, Z1, X2, Y2, Z2');

READLN (X1, Y1, Z1, X2, Y2, Z2);

SK:= X1*X2+Y1*Y2+Z1*Z2;

A1:=SQRT (X1*X1+Y1*Y1+Z1*Z1);

A2:=SQRT (X2*X2+Y2*Y2+Z2*Z2);

CAL:=SK/(A1*A2);

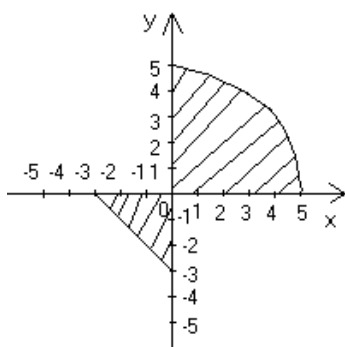
WRITELN ('CAL= ', CAL);

END.

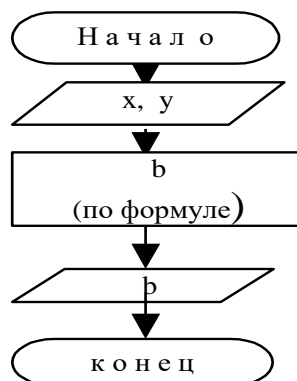
Пример 3. Для данной области составить линейную программу, которая печатает **TRUE**, если точка с координатами (x, y) принадлежит закрашенной области, и **FALSE** в противном случае.

Решение: Напишем систему неравенств определяющих, закрашенную область.

$$\begin{cases} x^2 + y^2 \leq 25 \\ x \geq 0 \\ y \geq 0 \end{cases} \quad \text{или} \quad \begin{cases} y \geq -x - 3 \\ x \leq 0 \\ y \leq 0 \end{cases}$$



Напишем систему неравенств в виде одного логического выражения



$$b = ((x^2 + y^2 \leq 25 \wedge x \geq 0 \wedge y \geq 0) \vee (y \geq -x - 3 \wedge x \leq 0 \wedge y \leq 0))$$

```

PROGRAM P3;
VAR X,Y: REAL; B: BOOLEAN;
BEGIN
WRITELN ('Введите X,Y');
READLN (X,Y);
B:=((X*X+Y*Y<=25)AND(X>=0)AND
(Y>=0))OR((Y>=-X-3)AND(X<=0)AND (Y<=0));

WRITE(B);
END.

```

Варианты заданий

Задание 1.

Составить блок-схему и программу для вычисления значения функции по формуле (все переменные принимают действительные значения)

$$1. Z = \frac{b\sqrt{b^2+4ac}}{2a}; \quad 2. Z = \frac{a}{c} \times \frac{b}{d} \times \frac{abc}{cd};$$

$$3. Z = \frac{\sin + \cos}{\cos \sin} \operatorname{tg}; \quad 4. Z = \frac{x+y}{y+1} \frac{xy-1}{34x};$$

$$5. Z = \frac{3+e^{y-1}}{1+x^2} \operatorname{tg}; \quad 6. Z = x - \frac{x^3}{3} + \frac{x^5}{5};$$

$$7. Z = \ln \left(y - \sqrt{x} \right) \left(x - \frac{y}{x + \frac{x^2}{4}} \right); \quad 8. Z = \frac{1 + \sin \sqrt{x+1}}{\cos(3-4)};$$

$$9. Z = \operatorname{tg} \operatorname{ctg} \operatorname{tg} \operatorname{ctg} \operatorname{tg} \operatorname{ctg}; \quad 10. Z = \frac{\ln |\cos x|}{\ln(1+x^2)};$$

$$11. Z = \left(\frac{x+1}{x-1} \right)^x + 1.8^2; \quad 12. Z = \left(1 + \frac{1}{x^2} \right)^x - 1.2^2;$$

$$13. Z = \frac{x^2 - 7x + 1}{x^2 - 8x + 12}; \quad 14. Z = \frac{\cos - \operatorname{tg}(\operatorname{tg})}{\pi x};$$

$$15. Z = 2^x - \cos(\pi);$$

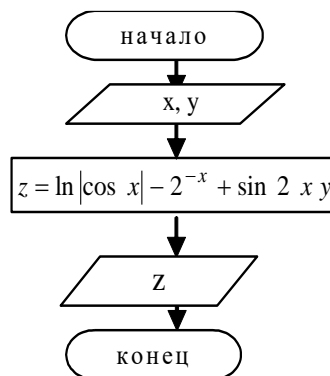
Задание 2.

Вычисление в математических задачах.

1. Вычислить периметр и площадь прямоугольного треугольника по длинам двух катетов **a** и **b**.
2. Заданы координаты трех вершин треугольника $(x_1, y_1), (x_2, y_2), (x_3, y_3)$. Найти его периметр и площадь.
3. Вычислить длину окружности и площадь круга одного и того же заданного радиуса **R**.
4. Найти произведение цифр заданного четырехзначного числа.

5. Найти два числа. Найти среднее арифметическое кубов этих чисел и среднее геометрическое модулей этих чисел.
6. Вычислить расстояние между двумя точками с данными координатами (x_1, y_1) (x_2, y_2) .
7. Даны два действительных числа x и y . Вычислить их сумму, разность, произведение и частное.
8. Дана длина ребра куба. Найти площадь грани, площадь полной поверхности и объем этого куба.
9. Дана сторона равностороннего треугольника. Найти площадь этого треугольника. Найти площадь этого треугольника, его высоту, радиусы вписанной окружностей.
10. Известна длина окружности. Найти площадь круга, ограниченного этой окружностью.
11. Найти площадь кольца, внутренний радиус которого равен r , а внешний R ($R > r$).
12. Треугольник задан величинами своих углов и радиусом описанной окружности. Найти стороны треугольника.
13. Найти площадь равнобедренной трапеции с основаниями a и b и углом α при большем основании a .
14. Вычислить корни квадратного уравнения $ax^2+bx+c=0$ с заданными коэффициентами a , b и c (предполагается, что $a \neq 0$ и что дискриминант уравнения неотрицателен).
15. Дано действительное число x . Не пользуйтесь никакими другими арифметическими операциями, кроме умножения, сложения и вычитания, вычислить за минимальное число операций $2x^4-3x^3+4x^2-5x+6$.

Лабораторная работа №6 Разработка программ разветвляющейся структуры. Оператор выбора



Решение
 большинства задач не удается свести к алгоритму линейной структуры. При этом вычислительный процесс может иметь несколько ветвей, переход к которым осуществляется в

зависимости от выполнения некоторых условий. Операторы, реализующие вычисления в каждой ветви, занимают в программе определенные места, поэтому переход к ним потребует нарушения естественного порядка выполнения операторов. Для этого используются **операторы управления**.

Метки. Оператор перехода.

Метка в стандарте языка Паскаль представляет собой целое неотрицательное число. Все используемые в программе метки должны быть перечислены в разделе описания меток, начинающемся служебным словом Label, например:

Label 1, 2, 8;

Одной меткой можно пометить только один оператор. Метка от помеченного оператора отделяется двоеточием.

Пример:

6: Writeln(14/2);

Во всех приведенных ранее программах операторы выполнялись один за другим в том порядке, в котором они были записаны в тексте. Такая алгоритмическая структура называется прямым следованием. Однако, в языке Паскаль изначально существует оператор, нарушающий прямолинейное выполнение программы, передающий управление в произвольную ее точку. Такая инструкция называется безусловным переходом и имеет такой формат:

```
Goto <метка>;
```

Оператор, к которому происходит переход должен быть помечен данной меткой.

Использовать оператор безусловного перехода следует крайне осторожно во избежание получения ошибочных результатов или полного "зацикливания" программы. Вообще, употребление данной команды среди программистов считается дурным тоном. Всегда существует возможность обойтись без него.

Условный оператор. Имеет две формы: полную и краткую.

Полная форма условного оператора:

```
If <логическое выражение>  
Then <оператор 1>  
Else <оператор 2>;
```

If – если, Then-тогда, Else-иначе - служебные слова, <оператор 1> и <оператор 2> простые или составные операторы. (Составной оператор – это любое количество операторов, заключенных в операторные скобки Begin end).

Выполняется условный оператор следующим образом: вычисляется значение логического выражения, если оно истинно, выполняется оператор 1, иначе – оператор 2.

Краткая форма условного оператора:

```
If <логическое выражение>  
Then <оператор > ;
```

Пример 1. Составить программу для решения задачи: Из двух чисел выбрать наибольшее.

```
Program Example1;  
Var A,B,C : Real; {A,B - для хранения аргументов, C - результат}  
Begin  
  Writeln('Введите два числа');  
  Readln(A,B);           {Вводим аргументы с клавиатуры}  
  If A>B Then C:=A Else C:=B; {Если A>B, то результат - A, иначе результат - B}  
  Writeln(C);           {Выводим результат на экран}  
End.
```

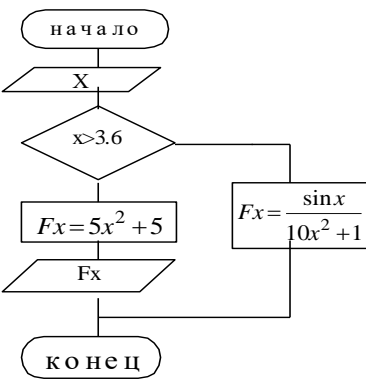
Пример 2. Составить программу для решения задачи: По заданным коэффициентам решить квадратное уравнение.

```
Program Sq1;  
Var A, B, C, D, X1, X2 : Real;  
Begin  
  Writeln ('Введите коэффициенты квадратного уравнения');  
  Readln (A,B,C);  
  D:=B*B-4*A*C;  
  If D<0 Then Writeln ('Корней нет! ')  
  Else  
  Begin  
    X1:=(-B+SQRT(D))/2/A;  
    X2:=(-B-SQRT(D))/2/A;  
    Writeln ('X1=', X1:8:3, ' X2=',X2:8:3)  
  End  
End.
```

Интересно, что в качестве оператора, который выполняется по выполнению или невыполнению условия, может выступать условный же оператор. В этом случае говорят о вложенности условных операторов. Обычно при записи условных операторов на языке Паскаль (особенно при множественных ветвлениях) команды записывают уступом вправо и вниз. Это повышает наглядность, и снижает потери времени на отладку.

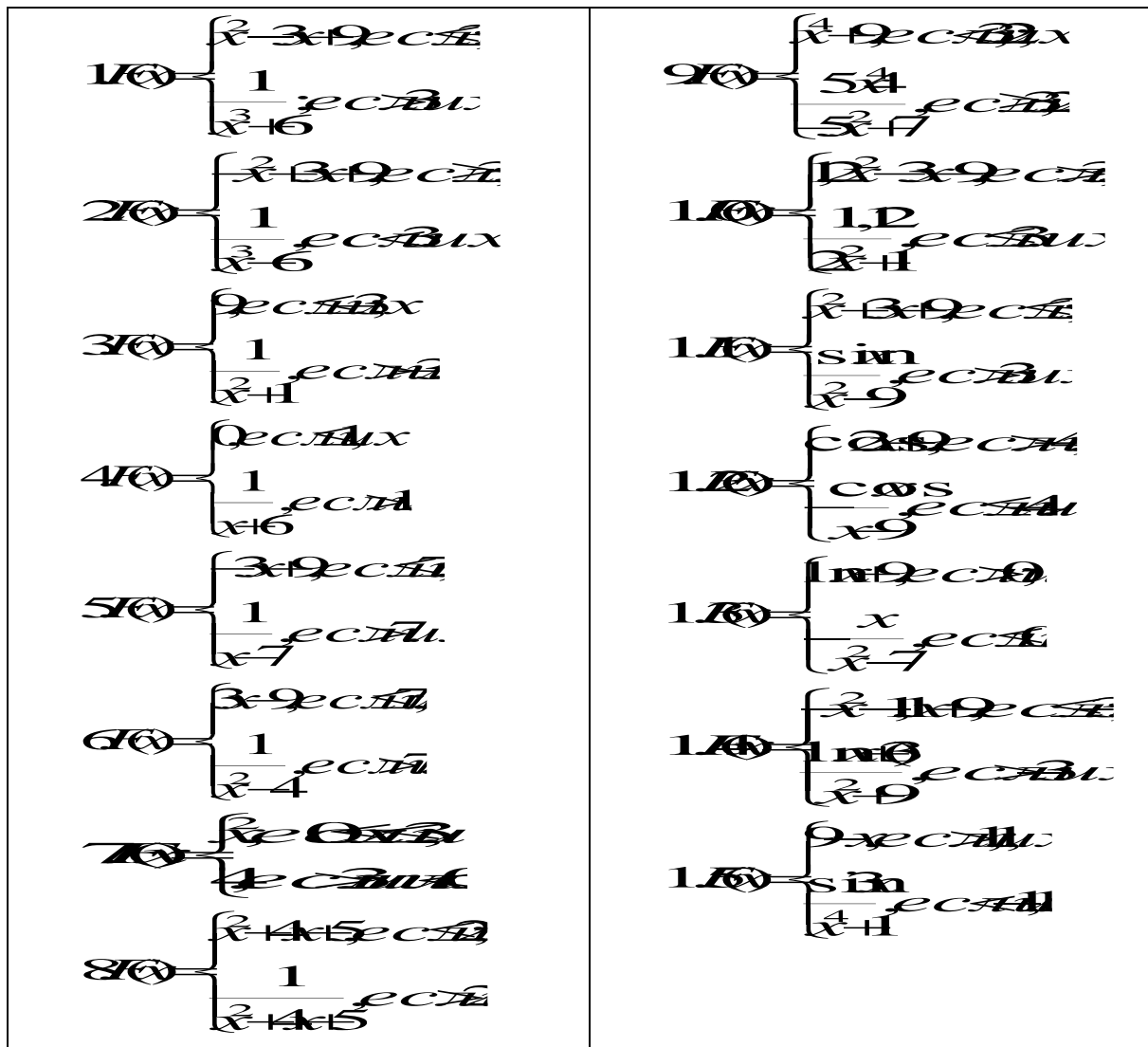
Пример 3. Составить блок-схему и программу для вычисления значения функции.

$$f(x) = \begin{cases} 5x^2 + 5 & \text{если } x > 3.6 \\ \frac{\sin x}{10x^2 + 1} & \text{если } x \leq 3.6 \end{cases}$$

| БЛОК-СХЕМА | |
|---|---|
|  <pre> graph TD Start([начало]) --> Input[/X/] Input --> Decision{x > 3.6} Decision -- Yes --> Process1[Fx = 5x^2 + 5] Decision -- No --> Process2[Fx = sin x / (10x^2 + 1)] Process1 --> Output[/Fx/] Process2 --> Output Output --> End([конец]) </pre> | <pre> PROGRAM P23; VAR X,Y:REAL; BEGIN WRITELN('Введите x'); READLN(X); IF (X.>3.6) THEN Fx:=5*SQR(X)+5 ELSE Fx:=SIN(X)/(10*SQR(X)+1); WRITE(Fx:6:2); END. </pre> |

Варианты заданий.

Составить блок-схему и программу для вычисления значения функции,.



Тема Оператор выбора

Оператор выбора используется в тех случаях, когда в зависимости от значения какого-либо выражения необходимо выполнить один из нескольких операторов.

Общий вид оператора:

CASE < Выражение > OF

Константа 1: оператор 1;

Константа 2: оператор 2;

.....

Константа n: оператор n

Else

оператор n+1

End;

Case (в случае), OF (из), End (конец) – служебные слова.

Выражение любого порядкового типа (чаще всего целого типа).

Если значение выражения равно одной из констант, то выполняется соответствующий оператор. Если значение выражения не совпадает ни с одной из констант, то управление передается к оператору n+1.

Вместо каждой константы может быть список констант.

Другая форма оператора выбора:

CASE < Выражение > OF

Константа 1: оператор 1;

Константа 2: оператор 2;

.....

Константа n: оператор n

End;

Если значение выражения не совпадает ни с одной из констант, то управление передается за пределы группы

Пример. Составить программу, которая выводит название оценки словом, при вводе ее цифрой.

Задание. Составить программу для решения задачи.

```
PROGRAM PW;  
VAR  
  N: INTEGER;  
  BAGIN  
WRITELN(' Введите оценку N');  
READLN(N);  
  CASE N OF  
1: WRITELN('КОЛ');  
2: WRITELN('ДВОЙКА');  
3: WRITELN('ТРОЙКА');  
4: WRITELN('ЧЕТЫРЕ');  
5: WRITELN('ПЯТЕРКА')  
ELSE  
WRITELN('ТАКОЙ ОЦЕНКИ НЕТ');  
END;  
END.
```

Варианты заданий.

1. Ввести номер недели и вывести соответствующий ему день недели на русском и английском языках.

2. Ввести номер месяца и вывести соответствующее ему название на русском и английском языках.

3. Введите номер месяца и напечатайте соответствующее месяцу время года.

4. Введите время (только часы) и напечатайте соответствующее этому времени сообщение: «Доброе утро», «Добрый день», «Добрый вечер», «Доброй ночи».

5. Введите число от 1 до 7. Напечатайте соответствующий номеру цвет из цветов радуги.

6. Введите число от 1 до 10. Напечатайте фамилию обучающегося с соответствующим номером в журнале группы.

7. Составить программу, которая по заданным году и номеру месяца определяет количество дней в этом месяце.

8. Для каждой введенной цифры (0-9) вывести соответствующее ей название на английском языке (0-zero, 1-one, 2-two,...).

9. Пусть элементами круга являются радиус (первый элемент), диаметр (второй элемент) и длина окружности (третий элемент). Составить программу, которая по номеру элемента запрашивала бы его соответствующее значение и вычисляла бы площадь круга.

10. _____ Пусть элементами прямоугольного равнобедренного треугольника являются:
1) катет a; 2) гипотенуза b; 3) высота h, опущенная из вершины прямого угла на гипотенузу; 4) площадь S. Составить программу, которая по номеру и значению соответствующего элемента вычисляла бы значения всех остальных элементов треугольника.

11. В старояпонском календаре был принят 12-летний цикл. Годы внутри цикла носили названия животных: крысы, коровы, тигра, зайца, дракона, змеи, лошади, овцы, обезьяны, курицы, собаки и свиньи. Написать программу, которая вводит номер некоторого года и печатает его название по старояпонскому календарю. (Справка: 1996 год - год крысы — начало очередного цикла)

12. Написать программу, которая по введенному числу от 1 до 5 (номеру курса) выдает соответствующее сообщение «Привет k-курсник». Например, если k=1, «Привет, первокурсник».

13. Написать программу, которая по введенному числу от 1 до 12 (номеру месяца)

выдает все приходящиеся на этот месяц праздничные дни (например, если введено число 1, то должно получиться: 1-е января-Новый год, 7-е января-Рождество).

14. Для целого числа k от 1 до 99 напечатать фразу «Мне k лет», учитывая при этом, что при некоторых значениях k слово «лет» надо заменить на слово «год» или «года». Например, 11 лет, 22 года, 51 год.

15. Написать программу которая бы по введенному номеру единицы измерения (1-килограмм, 2- миллиграмм, 3 - грамм, 4 - тонна, 5 — центнер) и массе M выдавала бы соответствующее значение массы в килограммах.

Лабораторная работа №7

Разработка программ с использованием цикла с предусловием. Разработка программ с использованием цикла с постусловием. Разработка программ с использованием цикла с параметром.

Алгоритм называется циклическим, если он содержит цикл. Циклом называют последовательность операторов, которая выполняется многократно при изменяющихся значениях некоторой переменной, называемой параметром цикла. Параметров в цикле может быть несколько.

Использование цикла позволяет существенно сократить объем программы.

На языке Паскаль имеется 3 вида операторов циклов.

- Оператор цикла с параметром (For)
- Оператор цикла с предварительным условием (While).
- Оператор цикла с последующим условием (Repeat).

1. Оператор цикла For .

Служит для организации цикла с известным числом повторений.

Имеет две формы:

А) For I:=m1 To m2 Do S;

Где For (для), To (до), Do (выполнить) – служебные слова. I – параметр цикла (переменная любого порядкового типа, чаще всего целого типа), m1 - начальное значение параметра цикла, m2 - конечное значение параметра цикла, (m1,m2 – константы или выражения того же типа, что и параметр I), S – циклическая часть оператора (простой или составной оператор).

Оператор работает следующим образом: сначала параметр I принимает начальное значение, равное m1, при каждом выполнении цикла параметр I увеличивается на единицу. Цикл выполняется пока $I \leq m2$, затем осуществляется выход из цикла к оператору, следующему за оператором цикла.

Б) For I:=m1 DownTo m2 Do S;

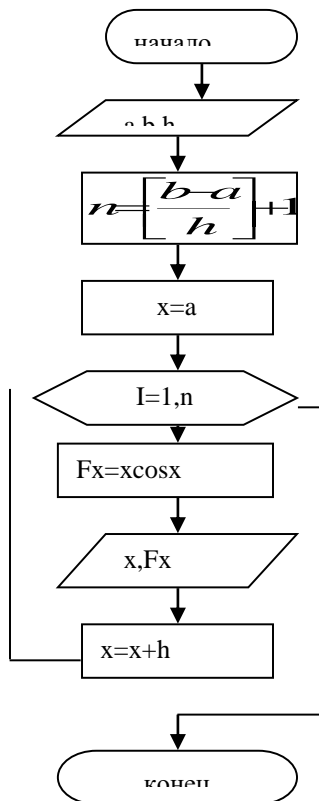
Вторая форма оператора цикла с параметром используется в случае, когда $m1 > m2$. При этом шаг изменения параметра цикла I равен -1.

Пример 1. Вычислить значения функции F_x на отрезке $[a, b]$ с шагом h . $F_x = x \cos x$

а) Организация цикла с помощью оператора цикла For

Вычислим число повторений цикла:

$$n = \left[\frac{b-a}{h} \right] + 1$$



```

Program CSP;
Var a,b,h,x,Fx: Real;
I,n: Integer;
Begin
Writeln('Введите a,b,h');
Readln(a,b,h);
n:=Trunc((b-a)/h)+1;
x:=a;
For I:=1 To n Do
Begin
Fx:=x*cos(x);
Writeln(x:3:1,Fx:5:2);
x:=x+h;
End;
End.
  
```

Оператор цикла с предусловием While.

Позволяет организовать цикл, в котором число повторений зависит от некоторого условия.

Общий вид записи оператора:

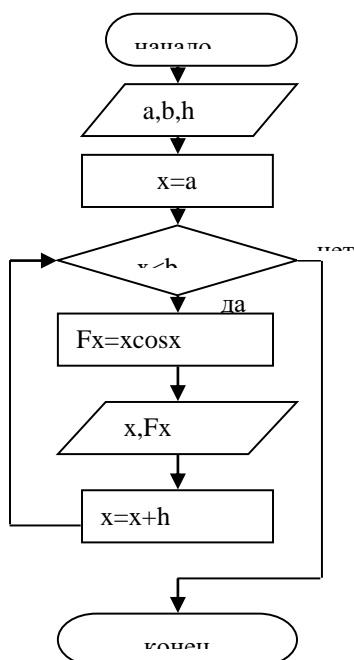
While < логическое выражение > Do S;

While (пока), Do (выполнить) – служебные слова.

S – тело цикла (простой или составной оператор).

Цикл выполняется пока логическое выражение истинно. Как только выражение станет ложным, выход из цикла.

Пример 2. Решить задачу из Примера 1, используя оператор цикла с предусловием.



```

Program CSPREDU;
Var a,b,h,x,Fx: Real;
I,n: Integer;
Begin
Writeln('Введите a,b,h');
Readln(a,b,h);
x:=a;
While x<=b Do
Begin
Fx:=x*cos(x);
Writeln(x:3:1,Fx:5:2);
x:=x+h;
End;
End.
  
```

Оператор цикла с последующим условием Repeat.

Позволяет также организовать цикл, в котором число повторений зависит от некоторого условия, записанного в конце цикла.

Общий вид записи оператора:

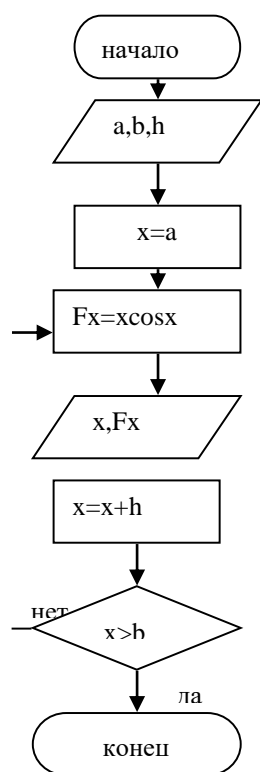
Repeat S Until <логическое выражение> ;

Repeat (повторять), Until (до тех пор пока) – служебные слова.

S – тело цикла (один или несколько операторов).

Цикл выполняется пока логическое выражение ложно. Как только выражение станет истинным, выход из цикла.

ПРИМЕР 3. Решить задачу из Примера 1, используя оператор цикла с последующим условием.



```
Program CSPOSTU;
Var a,b,h,x,Fx: Real;
I,n: Integer;
Begin
Writeln('Введите a,b,h');
Readln(a,b,h);
x:=a;
Repeat
Fx:=x*cos(x);
Writeln(x:3:1,Fx:5:2);
x:=x+h;
Until x>b;
End.
```

Варианты заданий

Составить программу и блок-схему для вычисления значений функции $F(x)$, где x изменяется на отрезке $[a,b]$ с шагом h .

1

1. $F(x) = x - \sin x$

2. $F(x) = \sin^2 x$

3. $F(x) = 2\cos x - 1$

4. $F(x) = \operatorname{tg} x$

5. $F(x) = \operatorname{ctg} x + 1$

6. $F(x) = 2 \sin^2 x + 1$

7. $F(x) = \sin x + \operatorname{tg} x$

8. $F(x) = \cos x + \operatorname{tg} x$

9. $F(x) = 2 \operatorname{tg} (x/2) + 1$

10. $F(x) = 2 \operatorname{tg} (x/2) + 2 \cos x$

11. $F(x) = \sin^2 x - \cos^2 x$

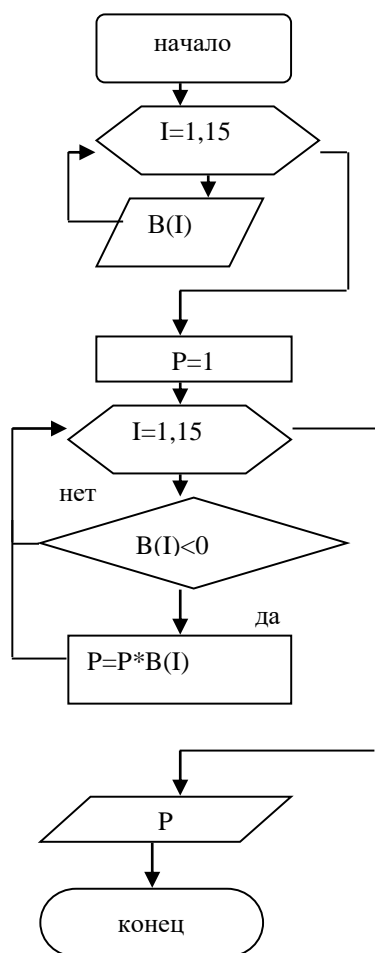
12. $F(x) = 7 \sin^2 x - 1/2 \cos x$

13. $F(x) = -\cos 2x$

14. $F(x) = \operatorname{tg} 2x - 3$

15. $F(x) = \sin x + 0,5 \cos x$

Блок-схема



Программа

```

PROGRAM WP;
VAR B:ARRAY[1..15] OF REAL;
I:INTEGER; P:REAL;
BEGIN
WRITELN('Введите элементы массива');
FOR I:=1 TO 15 DO
READ(B[I]);
P:=1.;
FOR I:=1 TO 15 DO
IF B[I]<0 THEN P:=P*B[I];
WRITELN(P);
END.
  
```

Варианты заданий.

1. а) Вычислить сумму элементов массива $X(15)$ с четными номерами.
 б) Вычислить произведение элементов массива $A(20)$.
2. а) Вычислить сумму элементов массива $A(20)$ с нечетными номерами.
 б) Вычислить произведение элементов массива $X(10)$, модуль которых больше 9.
3. а) Вычислить количество элементов массива $A(20)$ равных заданному числу x .
 б) Вычислить произведение элементов массива $A(20)$, с четными номер
4. а) Вычислить сумму элементов массива $(B_1, B_2, \dots, B_{15})$, значения, которых больше 2 и меньше 10.
 б) Найти произведение элементов массива $(B_1, B_2, \dots, B_{15})$, стоящих на четных местах.
5. а) Определить количество нулевых элементов массива $(A_1, A_2, \dots, A_{10})$.
 б) Найти произведение положительных элементов массива (D_1, D_2, \dots, D_8) , стоящих на четных местах.
6. а) Найти сумму элементов целочисленного массива $(Z_1, Z_2, \dots, Z_{20})$, значения, которых кратны 3.
 б) Вычислить произведение элементов массива $(N_1, N_2, \dots, N_{10})$, и разделить его на 2.
7. а) Вычислить сумму элементов массива $(a_1, a_2, \dots, a_{15})$, значения которых больше 2 и меньше 7.

б) Вычислить $y = \prod_{i=1}^{10} (1 + \sin i)$

8. а) Найти количество положительных элементов массива (z_1, z_2, \dots, z_9) , и разделить его на 2.

б) Найти произведение элементов массива $(x_1, x_2, \dots, x_{15})$, индексы которых кратны 3.

9. а) Составить программу для вычисления среднего арифметического отрицательных элементов массива $(z_1, z_2, \dots, z_{18})$

б) Вычислить произведение элементов массива (y_1, y_2, \dots, y_3) , модуль которых больше 5.

10. а) Составить программу для вычисления суммы и количества элементов массива $(A_1, A_2, \dots, A_{20})$, значение, которых больше 0.

б) Вычислить произведение элементов массива $(C_1, C_2, \dots, C_{10})$, удовлетворяющих условию $C_i < i$.

11. а) Составить программу для вычисления $y = \sum_{i=1}^{10} 3x_i / i$.

б) Найти произведение элементов массива $(x_1, x_2, \dots, x_{20})$, индексы которых кратны 4.

12. а) Составить программу для вычисления суммы квадратов элементов массива $(C_1, C_2, \dots, C_{10})$.

б) Вычислить $\sum_{i=1}^{15} \frac{1}{i} \sin i$.

13. а) Вычислить сумму элементов массива $(S_1, S_2, \dots, S_{10})$, у которых сумма значений индекса и элемента больше 5.

б) Найти произведение элементов целочисленного массива $(Y_1, Y_2, \dots, Y_{15})$, при делении которых на 3 получается целое число.

14. а) Составить программу для вычисления $y = \sum_{i=1}^{10} (a^3 + b)$.

б) Найти произведение и количество отрицательных элементов массива (z_1, z_2, \dots, z_8) .

15. а) Дан массив целых чисел $(J_1, J_2, \dots, J_{15})$, найти сумму нечетных элементов этого массива.

б) Вычислить $R = \left(\prod_{i=1}^{15} i \right) \wedge N$, N-количество элементов массива

Лабораторная работа №9 -10 Сортировка одномерных массивов.

Предположим, известны результаты соревнований по стрельбе, в которых принимали участие 9 человек. Расположить данные результаты в порядке возрастания набранных при стрельбе очков. Алгоритм решения данной задачи является наиболее сложным из приведенных выше примеров и требует использования вложенных циклов.

Один из способов сортировки массивов заключается в следующем. Сначала первый элемент массива в цикле сравнивается по очереди со всеми оставшимися элементами.

Если очередной элемент массива меньше по величине, чем первый, то эти элементы

переставляются местами. Сравнение продолжается далее уже для обновленного первого элемента. В результате окончания данного цикла будет найден и установлен на первое место самый наименьший элемент массива. Далее продолжается аналогичный процесс уже для оставшихся элементов массива, т.е. второй элемент сравнивается со всеми остальными и, при необходимости, переставляется с ними местами. После определения и установки второго элемента массива, данный процесс продолжается для третьего элемента, четвертого элемента и т.д. Алгоритм завершается, когда сравниваются и упорядочиваются предпоследний и последний из оставшихся элементов массива.

Программа реализации изложенного алгоритма может иметь следующий вид:

```
Program pr4;
Uses crt;
Type STREL=array[1..9]of integer;
Var
rez:strel;
i,j,s:integer;
Begin
For i:=1 to 9 do
begin
writeln('Введите результаты ',i,'-го участника');
readln(rez[i]);
end;
for i:=1 to 8 do
for j:=i+1 to 9 do
if rez[i]>rez[j] then
begin
s:=rez[j];
rez[j]:=rez[i];
rez[i]:=s;
end;
writeln('Отсортированные по возрастанию результаты:');
for i:=1 to 9 do write (rez[i]:5,' ');
end.
```

Здесь STREL – тип массива результатов стрельбы участников, rez[i] – переменная для описания результатов i-го участника (i – вспомогательная переменная s используется для перестановки местами элементов массива

ВАРИАНТЫ ЗАДАНИЙ

Исходные данные необходимо оформить в виде массива. При выполнении задания ввод исходных данных и вывод результатов сопровождать комментариями (какие данные нужно ввести и что получается в результате). Составить блок-схему программы. Оформить отчет

1. Подсчитать среднемесячную зарплату сотрудника предприятия.
2. Дан объем продукции, выпущенной заводом за год (помесячно). Найти наименьший объем. В качестве результата вывести номер месяца и объем выпущенной продукции.
3. Курс доллара в течение года менялся в диапазоне от 28руб. до 30руб. Найти наибольшее значение курса доллара. В качестве результата вывести номер месяца и значение курса доллара.
4. Известен месячный план выпуска некоторой продукции и объема выпущенной этой продукции заводом за год (помесячно). Определить, когда завод не выполнил план. Результат получить в виде: номер месяца и объем выпущенной продукции.

5. Даны результаты сдачи экзамена по информатике группы обучающихся (в группе 20 обучающихся). Подсчитать количество обучающихся, не сдавших экзамен.

6. Известна среднемесячная зарплата 10 сотрудников отдела. Расположить данные в порядке убывания.

7. Известен годовой процент на вклад с капитализацией (начисление процентов ежемесячно). Определить, сколько денег получит вкладчик в конце года, если на 1 января сумма вклада составляла 1500руб. в качестве результата вывести сумму вклада на конец каждого месяца.

8. Известны данные по продаже компьютеров в течение недели. Найти общее количество проданных компьютеров.

9. Известны данные по продаже компьютеров в течение недели. Расположить эти данные в порядке возрастания.

10. Известен месячный план выпуска некоторой продукции и объемы выпущенной продукции заводом за год (помесячно). Определить месяц, в котором было максимальное отклонение от плана. В качестве результата вывести номер месяца и отклонение.

11. Известен месячный план выпуска некоторой продукции и объемы выпущенной продукции заводом за год (помесячно). Определить, был ли выполнен годовой план.

12. Даны результаты сдачи экзамена по информатике группы обучающихся (в группе 20 обучающихся). Подсчитать количество обучающихся, сдавших экзамен без троек.

13. Известен месячный план выпуска некоторой продукции и объемы выпущенной этой продукции заводом за год (помесячно). Определить, когда завод перевыполнил план. Результат получить в виде: номер месяца и объем продукции, выпущенной сверх плана.

14. Подсчитать среднемесячную зарплату сотрудника предприятия и найти зарплату, которая наиболее близка к средней. В качестве результата вывести среднюю зарплату, наиболее близкую и ее номер в массиве.

15. Даны результаты сдачи экзамена по информатике группы из 15 обучающихся. Подсчитать количество обучающихся, не сдавших экзамен, в численном и в процентном соотношении.

Лабораторная работа №11 – 13

Разработка программ с использованием двумерных массивов.

Двумерные массивы

Исходные данные для решения многих задач удобно представить в виде таблицы.

Например, результат производственной деятельности заводов некоторой фирмы можно представить в виде таблицы (см. Таблица 1)

Колонки и строки таблицы, как правило, содержат однородную информацию, если использовать терминологию Pascal – данные одинакового типа. Поэтому в программе для хранения и обработки табличных данных можно использовать совокупность одномерных массивов. Так, приведенная выше таблица может быть представлена как совокупность одномерных массивов следующим образом:

```
Zavod1:array[1..4] of integer;  
Zavod2:array[1..4] of integer;  
Zavod3:array[1..4] of integer;
```

Каждый из приведенных массивов может хранить информацию о количестве продукции, выпущенной одним заводом.

Возможно и такое представление:

```
product1:array[1..3] of integer;  
product2:array[1..3] of integer;  
product3:array[1..3] of integer;  
product4:array[1..3] of integer;
```


Таблица 1

| | Продукт1 | Продукт2 | Продукт3 | Продукт4 |
|--------|----------|----------|----------|----------|
| Завод1 | 1500 | 14000 | 15 | 125 |
| Завод2 | 1380 | 15600 | 25 | 140 |
| Завод3 | 2500 | 13000 | 8 | 165 |

В этом случае массив предназначен для хранения информации о количестве продукции одного наименования, произведенной каждым заводом. Помимо совокупности одномерных массивов, таблица может быть представлена как двумерный массив. В общем виде описание двумерного массива выглядит следующим образом:

Имя: **array**[нижняя_граница_индекса1..верхняя_граница_индекса1,нижняя_граница_индекса2..верхняя_граница_индекса2] **of** тип

где Имя – имя массива;

array – ключевое слово, показывающее, что объявляемый элемент данных является массивом;

нижняя граница индекса1, верхняя граница индекса1, нижняя граница индекса2, верхняя граница индекса2 – целые константы, определяющие диапазоны изменения индексов и, следовательно, число элементов массива;

тип – тип элементов массива.

Приведенная выше таблица (см. Таблица 1) может быть представлена в виде двумерного массива следующим образом:

Product: array[1..3,1..4] of integer;

Чтобы использовать элемент массива, нужно указать имя массива и индексы элемента. Первый индекс обычно соответствует номеру строки таблицы, второй – номеру колонки. Так, элемент product[2,3] содержит число продуктов третьего наименования, выпущенных вторым заводом.

Двумерные массивы, в которых диапазоны индексов начинаются с 1, также называются иногда матрицами. Размерность матрицы определяется как $M*N$, где M – число строк в матрице, N – число столбцов. Если число строк матрицы равняется числу столбцов, то матрицы такого вида называются квадратными.

Элементы квадратной матрицы вида $V[1,1], V[2,2], \dots, V[3,3]$ составляют главную диагональ матрицы. Иногда вводят понятие побочной диагонали квадратной матрицы, которую составляют элементы $V[1,N], V[2,N-1], V[3,N-2], \dots, V[N,1]$, где N – число строк (столбцов) матрицы.

Приведем еще примеры описания двумерных массивов в Pascal-программах:

Type MATR=array[1..4,1..5] of integer;

Type V= array[2..9,0..6] of real;

Type C= array[-1..4,-1..4] of char.

Также допускается указание имени другого типа массива в качестве типа элементов массива, например:

Type VEC= array[1..4] of real;

MAS= array[1..5] of vec.

В результате приведенного выше описания тип массива MAS будет объявлен как тип двумерного массива, первый индекс которого будет меняться от 1 до 5, а второй индекс – от 1 до 4, т.е. размерность массива составит $5*4$ элементов.

При вводе и выводе значений элементов двумерных массивов используются вложенные циклы, в которых внешний оператор цикла, как правило, задает изменение строк массива, внутренний оператор цикла – изменение столбцов.

ПРИМЕРЫ ЗАДАЧ

1. Нахождение наибольшего элемента в заданной строке.

```
Program Max_str;
Uses crt;
Type Matr=array[1..3,1..4] of real;
Var max:real;
a:Matr;
i,j:integer;
begin
for i:=1 to 3 do
for j:=1 to 4 do
begin
writeln('Введите элемент a[',i,',',j,']');
readln(a[i,j]);
end;
max:=a[2,1];
for j:=2 to 4 do
if max<a[2,j] then max:=a[2,j];
writeln('Наибольший элемент второй
строки=',max:8:2);
end.
```

Данная программа представляет собой реализацию алгоритма нахождения наибольшего элемента вектора, полученного путем фиксирования одного из индексов двумерного массива.

2. Нахождение элементов массива, удовлетворяющих заданному условию.

Известны результаты 5 обучающихся по итогам экзаменов по химии и информатике. Найти фамилии обучающихся, сдавших оба экзамена на отлично. Для решения поставленной задачи может быть использована следующая программа (ее блок-схема представлена на рис. 2).

```
Program Sessia;
type PR=array [1..5,1..2]of integer;
Fam=array[1..5]of string[10];
var r:pr;
st:fam;
i,j:integer;
begin
for i:=1 to 5 do
begin
writeln('Введите фамилию ',i,'-го обучающегося ');
readln(st[i]);
writeln('Введите оценку данного обучающегося по
химии (от 2 до 5)');
readln(r[i,1]);
writeln('Введите оценку данного обучающегося по
информатике (от 2 до 5)');
readln(r[i,2]);
end;
for i:=1 to 5 do
if (r[i,1]=5) and (r[i,2]=5) then
writeln(' Обучающийся -отличник - ',st[i]);
```

end.

В данной программе для хранения фамилий обучающихся используется одномерный строковый массив `st` типа `Fam`, для хранения оценок обучающихся – двумерный целочисленный массив `r` типа `PR`, причем первый столбец матрицы `r` используется для хранения результатов экзамена по химии, второй столбец – экзамена по информатике. Если у некоторого обучающегося оценки за оба экзамена составили 5 баллов, то его фамилия будет выведена на экран с сообщением «обучающийся -отличник».

3. Нахождение сумм элементов строк матриц

Рассмотрим задачу нахождения сумм элементов строк матрицы на примере задачи подсчета итогов футбольного чемпионата. Пусть задана таблица результатов игр 5 команд футбольного чемпионата размером 5x5. На диагонали таблицы стоят значения 0, другие элементы таблицы равны 0, 1 или 2, где 0 баллов соответствует проигрышу команды в игре, 1 балл – ничьей, 2 балла – выигрышу. Определить сумму баллов каждой команды по результатам чемпионата.

Легко заметить, что для построения матрицы `R` результатов игр достаточно ввести лишь стоящую выше (или ниже) главной диагонали половину матрицы, т.к. результаты остальных игр могут быть рассчитаны из известного соотношения: если, например, первая команда обыграла вторую, то элемент $R[1,2]=2$, а элемент $R[2,1]=2-R[1,2]=0$; аналогично, если вторая команда сыграла вничью с третьей, то $R[2,3]=1$, $R[3,2]=2-R[2,3]=1$. Таким образом, нетрудно получить вид взаимосвязи элементов матрицы: $R[i,j]+R[j,i]=2$, где i и j меняются от 1 до 5 (кроме элементов главной диагонали). На главной диагонали матрицы `R` по условию задачи всегда стоят числа 0.

```
Program foot;
Type tab=array[1..5,1..5] of integer;
Var r:tab; i,j,s:integer;
begin
  {ввод стоящих выше диагонали элементов матрицы}
  for i:=1 to 4 do
    for j:=i+1 to 5 do
      begin
        writeln ('Введите результат игры ',i,'-й команды с ',j,' -й: 0, 1 или 2 балла');
        readln(r[i,j]);
      end;
    {заполнение стоящих на диагонали элементов нулями}
    for i:=1 to 5 do r[i,i]:=0;
    {вычисление стоящих ниже диагонали элементов матрицы}
    for i:=2 to 5 do
      for j:=1 to i-1 do r[i,j]:=2-r[j,i];
    {вывод на экран матрицы результатов игр}
    writeln('Таблица чемпионата');
    for i:=1 to 5 do
      begin
        for j:=1 to 5 do write(r[i,j]:4);
        writeln;
      end;
    {вычисление сумм элементов строк матрицы}
    for i:=1 to 5 do
      begin
        s:=0;
        for j:=1 to 5 do s:=s+r[i,j];
        writeln(i,'-ая команда набрала ',s:3,' очков');
      end;
    end.
```

ЗАДАНИЯ

Исходные данные необходимо оформить в виде двумерного массива, в части заданий использовать дополнительно и одномерные массивы. При выполнении задания ввод исходных данных и вывод результатов сопровождать комментариями (какие данные нужно ввести и что получается в результате).

1. Известен план выпуска компьютеров и количество выпущенных компьютеров тремя фирмами за шесть месяцев. Определить для каждой фирмы, был ли выполнен план по итогам шести месяцев.

2. Известно количество сделанных столов тремя фабриками за два квартала. Определить максимальное количество выпущенных столов. В качестве результата вывести месяц, в котором это было, и название фабрики.

3. Известен план выпуска компьютеров и количество выпущенных компьютеров тремя фирмами за три месяца. Определить, в каком месяце не был выполнен план третьей фирмой.

4. Известен план выпуска компьютеров и количество выпущенных компьютеров тремя фирмами за шесть месяцев. Определить для каждой фирмы количество месяцев, когда план был перевыполнен.

5. Известна заработная плата, полученная 5 сотрудниками отдела в течение года. Определить максимальную заработную плату. В качестве результата вывести фамилию и размер заработной платы.

6. Известно количество выпущенной продукции тремя заводами за первый квартал (помесечно). Найти среднемесячное количество выпущенной продукции для каждого завода.

7. Известно количество выпущенной продукции тремя заводами за первый квартал (помесечно). Найти среднемесячное количество выпущенной продукции по всем заводам.

8. Известны результаты сдачи трех экзаменов пятью обучающимися. Найти фамилии обучающихся, не сдавших оба экзамена.

9. Известно количество сделанных столов тремя фабриками за два квартала. Определить, какая фабрика выпустила максимальное количество столов по итогам шести месяцев.

10. Известна заработная плата, полученная 10 сотрудниками отдела в течение года. Определить среднемесячную зарплату по отделу.

11. Известны результаты сдачи двух экзаменов семью обучающимися. Найти фамилии обучающихся, не сдавших хотя бы один экзамен.

12. Известно количество сделанных столов тремя фабриками за два квартала. Определить, какая фабрика выпустила минимальное количество столов по итогам первых трех месяцев.

13. Известны результаты сдачи трех экзаменов десятью обучающимися. Найти средний балл каждого обучающегося и общий средний балл. Точность среднего балла – два знака после запятой.

14. Известно количество сделанных столов тремя фабриками за два квартала. Определить, какая фабрика выпустила максимальное количество столов по итогам второго квартала.

15. Известны результаты сдачи двух экзаменов десятью обучающимися. Определить фамилии обучающихся, сдавших экзамены без троек.

Лабораторная работа №14 - 16 **Разработка программ с использованием процедур и функций.**

При составлении программ для решения задач часто приходится выполнять одну и ту же последовательность действий в различных местах программы, причем обычно с разными значениями переменных. Чтобы избавиться от многократного дублирования одинаковых фрагментов стали применять подпрограммы.

Подпрограммой называется именованная группа операторов, реализующая определенный алгоритм, которую можно вызывать по имени из различных мест программы любое количество раз.

В языке программирования Паскаль определены два вида подпрограмм: **процедуры**

и функции.

В программе описание процедур и функций должно располагаться между разделами описания переменных и основной программой. Каждая процедура или функция определяется только один раз, но может вызываться многократно.

Структура процедур и функций аналогична структуре полной программы на языке Паскаль.

Заголовок подпрограммы (имя подпрограммы и список формальных параметров с указанием их типов).

Раздел описаний.

Тело подпрограммы.

В процедурах и функциях могут быть описаны собственные метки, константы, типы, собственные переменные и даже собственные процедуры и функции.

Процедурой в Паскале называется особым образом оформленный фрагмент программы, имеющий собственное имя. Упоминание этого имени в тексте программы приводит к активации процедуры с таким же именем и называется вызовом процедуры.

Для обмена информацией между процедурой и основной программой используются один или несколько параметров.

Общий вид процедуры:

Procedure <имя> (<список формальных параметров>);

<описательная часть

Begin

<тело процедуры>

End;

Результат выполнения процедуры – одно или несколько значений. Он передается в основную программу как значение ее параметров.

При вызове процедуры ее формальные параметры заменяются фактическими в порядке их следования.

<имя> (<список фактических параметров>);

Фактические параметры - это параметры, которые передаются процедуре при обращении к ней.

Формальные параметры – это переменные, фиктивно присутствующие в процедуре и определяющие тип и место подстановки фактических параметров, над которыми производятся действия.

Число и тип формальных параметров и фактических параметров должны совпадать с точностью до их следования.

Формальные параметры процедуры делятся на **параметры-переменные** и **параметры-значения**.

Параметры-переменные определяют, как правило, выходные данные процедуры (результаты обработки данных), которые передаются в основную программу. В описании формальных параметров перед параметрами-переменными ставят слово VAR.

Если формальный параметр описан как параметр-переменная, то при вызове процедуры ему должен соответствовать фактический параметр в виде переменной того же типа.

Параметры-значения- перед ними в описании формальных параметров не ставится служебное слово VAR. И в процедуре работают только значения этих параметров. В основной программе после выхода из процедуры их значения не изменяются, т.е. остаются теми же, которые были до начала работы процедуры. Если формальный параметр описан как параметр-значение, то при вызове процедуры ему может соответствовать произвольное выражение того же типа.

В программе различают *глобальные* и *локальные* переменные.

Глобальные переменные – это те переменные, которые объявлены в описании основной программы. Они доступны как в основной программе, так и во всех ее

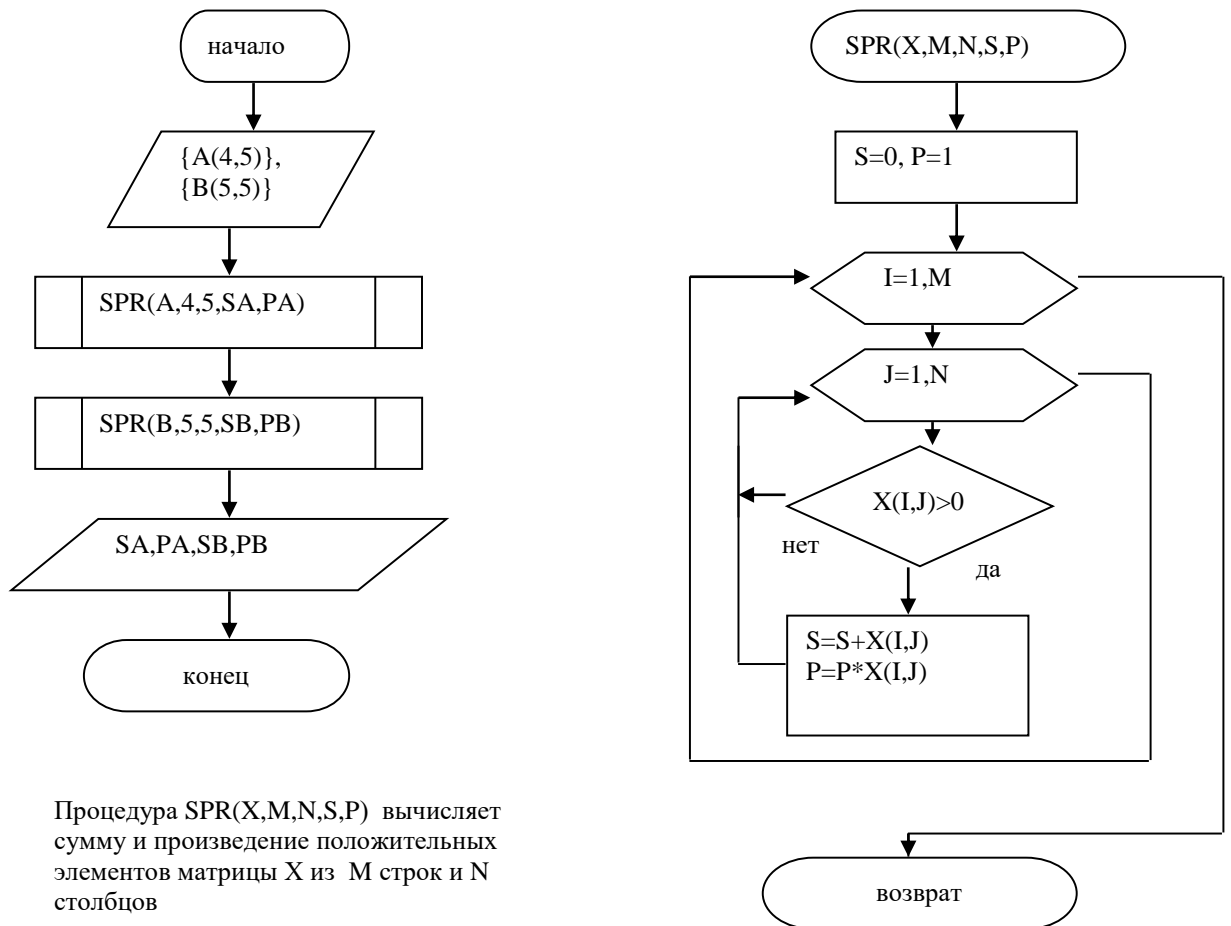
подпрограммах.

Локальные переменные- это те переменные, которые объявлены в подпрограммах. Они существуют только тогда, когда работает подпрограмма.

Локальные переменные доступны только внутри той подпрограммы, в которой они описаны.

В тех случаях, когда в процедуре и главной программе используются одни и те же имена параметров (процедура связана с главной программой посредством глобальных переменных), процедуру можно организовать без параметров.

Пример. Составить блок –схему и программу для вычисления суммы и произведения положительных элементов действительных матриц $A(4,5)$ и $B(5,5)$.



Программа

```

PROGRAM PPR;
TYPE MASS=ARRAY[1..5,1..5] OF REAL;
VAR
  A,B: MASS;
  I,J: INTEGER; SA,PA,SB,PB: REAL;
PROCEDURE SPR(X: MASS; M,N: INTEGER; VAR S,P: REAL);
BEGIN
  S:=0; P:=1;
  FOR I:=1 TO M DO
    FOR J:=1 TO N DO
      IF X[I,J] > 0 THEN
        BEGIN
          S:=S+ X[I,J];
          P:=P*X[I,J];
        END
    END
  END

```

```

    P:=P* X[I,J];
  END;
END;
  BEGIN
FOR I:=1 TO 4 DO
  BEGIN
FOR J:=1 TO 5 DO
  READ(A[I,J]);
  READLN;
FOR I:=1 TO 5 DO
  BEGIN
FOR J:=1 TO 5 DO
  READ(B[I,J]);
  READLN;
  SPR(A,4,5,SA,PA);
  SPR(B,5,5,SB,PB);
  WRITELN(SA:4:1,PA:4:1);
  WRITELN(SAB4:1,PB:4:1);
  END.

```

Функцией - в Паскале называется особым образом оформленный фрагмент программы, имеющий собственное имя. Результат работы функции возвращается в виде значения этой функции.

Структура функции:

```

Function <имя> (<список формальных параметров>): <тип результата>;
<описательная часть
Begin
<тело функции>
End;

```

Функцию можно использовать в качестве фактического параметра при обращении к другой функции или процедуре.

В теле любой функции нужно осуществить присваивание ей вычисленного значения. В левой части оператора присваивания в этом случае указывается имя функции.

Так же как и в случае с процедурами, различают формальные и фактические параметры функции.

Число и тип формальных и фактических параметров должны совпадать с точностью до их следования.

Тип формального параметра может быть только стандартным или ранее объявленным в разделе описания типов.

Пример. Составить программу для определения числа сочетаний $C_n^m = \frac{n!}{m!(n-m)!}$, используя функцию при вычислении факториала.

```

PROGRAM PFUN;
VAR CNM: REAL;
    N,M: INTEGER;
    FUNCTION FACT(K: INTEGER): INTEGER ;
VAR P,I: INTEGER;
  BEGIN
P:=1;
FOR I:=1 TO K DO
  P:=P*I;
  FACT:=P;

```

```

END;
BEGIN
  READLN(N,M);
  CNM:=FACT(N)/(FACT(M)*FACT(N-M));
  WRITLN(CNM);
END.

```

Рекурсивные подпрограммы.

При использовании процедур и функций может встречаться обращение к самим себе. Такое обращение называется рекурсивным. Рекурсия в языке Паскаль возможна, так как при вызове процедуры динамически создаются новые локальные переменные.

Многие математические функции можно выразить рекурсивно.

Например: $x = \begin{cases} 1 & \text{если } n=1 \\ x^{n-1} & \text{если } n > 1 \end{cases}$

или

$n = \begin{cases} 1 & \text{если } n=1 \\ n \cdot (n-1) & \text{если } n > 1 \end{cases}$

Рассмотрим подпрограмму-функцию для вычисления $n!$, использующую в своем описании рекурсивную формулу.

```

Function Factor(N: Integer): Integer;
Begin
  If N=0 Then
    Factor:=1
  Else
    Factor:=N*Factor(N-1);
  End;

```

Варианты заданий.

Составить блок схему и программу для решения задач. В задаче а) использовать функцию, в задаче б) использовать процедуру.

- а) Найти сумму положительных элементов целочисленных массивов A(6) и B(10).
- б) Вычислить сумму и количество отрицательных элементов действительных массивов X(10) и Y(5).
- а) Найти среднее арифметическое элементов целочисленных массивов C(10), D(15).
- б) Найти максимальные элементы и их порядковые номера действительных массивов A(10) и B(5).
- 3. а) Найти сумму элементов для действительных массивов A(5,5) и B(4,6).
- б) Найти сумму и количество отрицательных элементов массивов X(5,6) и Y(3,4).
- 4. а) Для действительных массивов A(5) и B(16) найти минимальные элементы.
- б) Вычислить сумму и произведение элементов целочисленных массивов X(3,2) и Y(3,4).
- 5. а) Найти среднее арифметическое положительных элементов массивов X(3,5) и Y(2,3).
- б) Вычислить сумму и количество элементов больших 5 для действительных массивов B(20) и C(10).
- 6. а) Вычислить сумму элементов массивов X(15) и Y(10).
- б) Вычислить произведение и сумму элементов массивов A(20) и B(10).
- 7. а) Вычислить сумму элементов массивов A(20) и B(15) с нечетными номерами.
- б) Вычислить произведение и количество элементов массивов X(10) и Y(12), модуль которых больше 9.
- 8. а) Вычислить количество элементов равных заданному числу x для массивов A(2,3) и B(3,4).

- б) Вычислить сумму и произведение четных элементов массивов A(20) и B(10).
9. а) Вычислить сумму элементов массивов A(10) и B(15), значения, которых больше 2 и меньше 10.
- б) Найти произведение и количество положительных элементов массива $(B_1, B_2, \dots, B_{15})$, стоящих на четных местах.
10. а) Определить количество нулевых элементов массива $(A_1, A_2, \dots, A_{10})$.
- б) Найти произведения положительных и отрицательных элементов целочисленного массива (D_1, D_2, \dots, D_8) .
11. а) Найти сумму элементов целочисленного массива $(Z_1, Z_2, \dots, Z_{20})$, значения, которых кратны 3.
- б) Вычислить количество и произведение отрицательных элементов действительной матрицы A(3,5).
12. а) Вычислить суммы элементов массивов $(a_1, a_2, \dots, a_{15})$ и $(x_1, x_2, \dots, x_{20})$, значения которых больше 2 и меньше 7.
- б) Найти сумму и произведение положительных элементов массивов A(3,5) и B(5,5).
13. а) Вычислить $R = S_1 + S_2 + S_3$, где $S_1 = \sum_{i=1}^{10} A_i$, $S_2 = \sum_{i=1}^{15} C_i$, $S_3 = \sum_{i=1}^{20} D_i$
- б) Составить программу для вычисления среднего арифметического отрицательных и среднего геометрического положительных элементов массивов (Y_1, Y_2, \dots, Y_3) и $(Z_1, Z_2, \dots, Z_{18})$.
14. а). Вычислить произведение элементов массива $(C_1, C_2, \dots, C_{10})$, удовлетворяющих условию $C_i < i$.
- б) Составить программу для вычисления суммы и количества элементов массива $(A_1, A_2, \dots, A_{20})$, значение, которых больше 0.
15. а) Вычислить $A = \frac{3K+2N^2}{\sqrt{4P}}$, где $K = \frac{20}{A}$, $N = \frac{14}{A}$, $P = \frac{5}{A}$
- б) Найти сумму и количество положительных элементов массивов A(3,5) и B(5,5).

Лабораторная работа №17 -18 Тестирование программного обеспечения.

Отладка – это процесс локализации и исправления ошибок, обнаруженных при тестировании программного обеспечения.

Локализацией называют процесс определения оператора программы, выполнение которого вызвало нарушение нормального вычислительного процесса. Для исправления ошибки необходимо определить ее причину, т.е. определить оператор или фрагмент, содержащие ошибку. Причины ошибок могут быть как очевидны, так и очень глубоко скрыты.

В целом сложность отладки обусловлена следующими причинами:

- требует от программиста глубоких знаний специфики управления используемыми техническими средствами, операционной системы, среды и языка программирования, реализуемых процессов, природы и специфики различных ошибок, методик отладки и соответствующих программных средств;
- психологически дискомфортна, так как необходимо искать собственные ошибки

и, как правило, в условиях ограниченного времени;

- возможно взаимовлияние ошибок в разных частях программы, например, за счет затирания области памяти одного модуля другим из-за ошибок адресации;
- отсутствуют четко сформулированные методики отладки.

Классификация ошибок

В соответствии с этапом обработки, на котором появляются ошибки, различают:

- синтаксические ошибки – ошибки, фиксируемые компилятором (транслятором, интерпретатором) при выполнении синтаксического и частично семантического анализа программы;
- ошибки компоновки – ошибки, обнаруженные компоновщиком (редактором связей) при объединении модулей программы;
- ошибки выполнения – ошибки, обнаруженные операционной системой, аппаратными средствами или пользователем при выполнении программы.

Методы отладки программного обеспечения

Отладка программы в любом случае предполагает обдумывание и логическое осмысление всей имеющейся информации об ошибке. Большинство ошибок можно обнаружить по косвенным признакам посредством тщательного анализа текстов программ и результатов тестирования без получения дополнительной информации. При этом используют различные методы:

- ручного тестирования;
- индукции;
- дедукции;
- обратного прослеживания.

Метод ручного тестирования

Это – самый простой и естественный способ данной группы. При обнаружении ошибки необходимо выполнить тестируемую программу вручную, используя тестовый набор, при работе с которыми была обнаружена ошибка. Метод очень эффективен, но не применим для больших программ, программ со сложными вычислениями и в тех случаях, когда ошибка связана с неверным представлением программиста о выполнении некоторых операций. Данный метод часто используют как составную часть других методов отладки.

Общая методика отладки программных продуктов, написанных для выполнения в операционных системах MS DOS и Win32:

- 1 этап – изучение проявления ошибки;
- 2 этап – определение локализации ошибки;
- 3 этап – определение причины ошибки;
- 4 этап – исправление ошибки;
- 5 этап – повторное тестирование.

Процесс отладки можно существенно упростить, если следовать основным рекомендациям структурного подхода к программированию:

- программу наращивать «сверху-вниз», от интерфейса к обрабатывающим подпрограммам, тестируя ее по ходу добавления подпрограмм;
- выводить пользователю вводимые им данные для контроля и проверять их на допустимость сразу после ввода;
- предусматривать вывод основных данных во всех узловых точках алгоритма (ветвлениях, вызовах подпрограмм).

Спецификация программы, программная спецификация (program specification) – точная и полная формулировка определенной задачи или группы задач, содержащая сведения, необходимые для построения алгоритма их решения. Содержит описание

результата, который должен быть достигнут с помощью конкретной программы, а также действий, выполняемых программой для достижения конечного результата без упоминания того, как указанный результат достигается

Практическая часть

Задание 1. Запишите вариант в отчет.

Задание 2. Согласно поставленной задаче выполните ручную отладку:

- Опишите математическую модель задачи с указанием имен и назначения переменных;

- Опишите спецификацию программы;

- Запишите алгоритм программы;

- Выполните отладку логики программы методом «грубой силы» с помощью соседа;

- Составьте тестовые наборы для проверки функционала системы.

Задание 3. Результаты выполнения практического задания запишите в отчет.

Лабораторная работа № 19-20 Компьютерные сети. Internet. Защита информации.

Методы защиты информации

Теоретические сведения

Под информационной безопасностью понимается защищенность информации и поддерживающей инфраструктуры от случайных или преднамеренных воздействий естественного или искусственного характера, направленных на нанесение ущерба владельцам или пользователям информации и поддерживающей инфраструктуры.

Под угрозой информационной безопасности понимается возможность осуществления действия, направленного против объекта защиты, проявляющаяся в опасности искажений и потерь информации.

Политика безопасности – набор законов, правил и норм поведения, определяющих, как организация обрабатывает, защищает и распространяет информацию. Выделяют следующие методы обеспечения безопасности.

Препятствие – метод физического преграждения пути злоумышленникам к информации, которая защищена от посторонних лиц.

Управление доступом – это метод защиты информации, с помощью которого для защиты используются все ресурсы самой системы (программные и технические средства). Управление доступом включает такие функции защиты, как: идентификация пользователей, персонала и ресурсов системы; аутентификация объекта или субъекта по предъявленному им идентификатору; проверка полномочий; разрешение и создание условий работы в пределах установленного регламента; регистрация обращений к защищаемым ресурсам; реагирование при попытке несанкционированных действий.

Маскировка – это метод защиты информации в каналах телекоммуникаций путем криптографического закрытия. Регламентация – метод, создающий такие условия автоматизированной обработки, хранения и передачи защищаемой информации, при которых возможности несанкционированного доступа к ней сводятся к минимуму.

Принуждение – метод, при котором пользователи и персонал вынуждены соблюдать правила обработки, передачи и использования защищаемой информации под угрозой материальной, административной или уголовной ответственности.

Побуждение – это метод, с помощью которого пользователь и персонал системы не нарушают установленные правила за счет соблюдения сложившихся моральных и этических норм.

Функционирование вышеперечисленных методов осуществляется с помощью следующих видов средств:

физические, которые могут быть представлены в виде автономных устройств (замки, решетки и т.д.);

аппаратные, которые реализуются в виде электрических, электромеханических и электронных устройств (наиболее известные аппаратные средства – это схемы контроля информации по четности, схемы защиты полей памяти по ключу и т.д.);

программные средства – это программное обеспечение, которое предназначено для выполнения функции защиты информации; организационные средства – это организационно-технические и организационно-правовые мероприятия, которые осуществляются в процессе создания и эксплуатации аппаратуры телекоммуникации для обеспечения защиты информации; законодательные средства определяются законодательными актами той страны, где они функционируют, регламентируют правила использования, обработки и передачи информации ограниченного доступа и устанавливают меры ответственности за нарушение этих правил; морально-этические средства защиты выражаются в виде норм, которые сложились традиционно по мере внедрения вычислительной техники и средств связи.

Стратегия обеспечения безопасности включает перечень основных мер, которые обеспечивают максимальную безопасность при заданном остаточном риске и минимальных затратах на их реализацию. Выделяют два основных направления стратегии защиты информации: анализ средств защиты; определение факта вторжения.

Организация, функционирующая на основе корпоративной информационной системы, вынуждена регулярно проверять, насколько используемые средства защиты информации соответствуют положениям принятой в организации политике безопасности. Этот процесс помогают автоматизировать средства анализа защищенности. Выделяют средства анализа защищенности сетевых протоколов и сервисов, а также средства анализа защищенности операционных систем.

Не менее важным является своевременность выявления факта вторжения. Для этого необходимы методы, позволяющие оперативно обнаруживать и предотвращать нарушения безопасности. Механизмы, применяемые в современных системах обнаружения атак, основаны на аппарате математической статистики (статистический метод), а также широко используются экспертные системы и нейронные сети.

Системы обнаружения атак можно классифицировать по способу реагирования, способу выявления атаки и способу сбора информации об атаке. Поскольку современные корпоративные информационные системы базируются на использовании сетей, то актуальным является применение различных сетевых протоколов защиты: SSL, SET, IPSec, SWAN, SMIME.

Задание а. Проанализировать средства антивирусной защиты и описать конфигурацию антивирусного пакета. **б.** Описать: классификацию вирусов и средств защиты; виды антивирусных программных продуктов; характеристики наиболее популярных антивирусных пакетов.

Методические указания по выполнению задания

1. Определите на вашем рабочем месте установленный антивирусный пакет. Просмотрите его настройки и опишите его конфигурацию (используйте экранные копии изображений).

2. Загрузите браузер для работы в сети Internet (например, Internet Explorer, Opera).

3. В адресной строке наберите адрес любой поисковой системы (например, www.yandex.ru, www.google.ru, www.poisk.com).

4. В поисковую строку введите ключевые слова для поиска.

5. Поочередно нажимая на ссылки из полученного списка ресурсов, найдите нужную информацию.

6. Используя буфер обмена, скопируйте информацию в текстовый редактор MS Word.

7. Систематизируйте полученную информацию и подготовьте отчет по

лабораторной работе.

Контрольные вопросы

1. Информационная безопасность.
2. Угроза информационной безопасности.
3. Подходы к классификации угроз информационной безопасности.
4. Способы воздействия угроз на объекты информационной безопасности.
5. Политика безопасности.
6. Методы обеспечения безопасности.
7. Средства обеспечения безопасности.
8. Средства анализа защищенности.
9. Системы обнаружения атак.
10. Правовое обеспечение безопасности информационных систем.

**Практические работы
по дисциплине Информатика и программирования**

ПРАКТИЧЕСКАЯ РАБОТА 1

**ТЕМА: ПОДПРОГРАММЫ, ИХ НАЗНАЧЕНИЕ И КЛАССИФИКАЦИЯ.
НЕСТАНДАРТНЫЕ ТИПЫ ДАННЫХ.**

Цели работы:

- Научиться описывать процедуры и функции в программе;
- Научиться задавать фактические и формальные параметры;
- Научиться организовывать вызов процедуры и функции в программе

Оборудование: ПК, ИСР Pascal ABC

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Подпрограмма — это последовательность операторов, которые определены и записаны только в одном месте программы, однако их можно вызвать для выполнения из одной или нескольких точек программы. Каждая подпрограмма определяется уникальным именем.

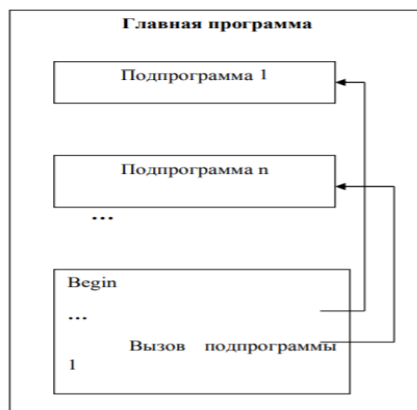


Рис. 1 Структура программы с подпрограммами

Различают два вида подпрограмм — это процедуры и функции.

Главная программа

...

Подпрограмма 1

Подпрограмма n

Begin

...

Вызов подпрограммы

1

...

Процедура и функция — это именованная последовательность описаний и операторов.

При использовании процедур или функций Pascal — программа должна содержать текст процедуры или функции и обращение к процедуре или функции. Тексты процедур и функций помещаются в раздел описаний процедур и функций.

Процедура — это независимая именованная часть программы, которую можно вызвать по имени для выполнения определённой в ней последовательности действий. Функция отличается от процедуры тем, что возвращает результат указанного при её описании типа.

Вызов функции может осуществляться из выражения, где имя функции используется в качестве операнда.

Заголовок процедуры имеет вид: PROCEDURE <имя> [(<сп. ф. п .>)] ;
Заголовок функции: FUNCTION <имя> [(<сп.ф.п.>)] : <тип>;
Здесь <имя> - имя подпрограммы (правильный идентификатор);
<сп.ф.п.> - список формальных параметров;
<тип> - тип возвращаемого функцией результата.

Описание, определение и вызов процедур
Описание процедуры производится в разделе описаний основной программы. Любая процедура оформляется аналогично программе, может содержать заголовок, разделы описаний и операторов. Синтаксис заголовка процедуры:

```
Procedure <Name>(<Список формальных параметров>);  
{Раздел описаний}  
Begin ... {Раздел операторов процедуры} End;  
где  
Procedure - служебное слово;  
Name - произвольный идентификатор, определяющий имя процедуры.  
Procedure MyProc (A,B,C: Real; var X1,X2: Real);  
Begin  
WriteLn('A=',A, ' B=', B, 'C=', C);  
X1:=A+B;  
X2:=A*B-C  
End;
```

Разделы описаний процедуры подобно основной программе могут содержать разделы описания меток (Label), констант (Const), типов (Type), переменных (Var) и раздел процедур и функций. Раздел операторов помещается после служебного слова Begin и заканчивается служебным словом End, после End ставится " ; ".

В основной программе процедуры располагают перед разделом операторов (телом программы) основной программы.

Формальные параметры — это переменные, посредством которых передаются данные из места вызова процедуры в её тело, либо из процедуры в места вызова. Список формальных параметров может отсутствовать, при этом символ " ; " ставится сразу за именем процедуры и данные из места вызова процедуры в её тело не передаются.

Для вызова процедуры на исполнение к ней необходимо обратиться.

Вызов процедуры производится указанием имени процедуры и списком фактических параметров:

```
Name(<Список фактических параметров>);  
MyProc(K, L+M, 12, Y1, Y2);
```

Выполнение оператора вызова процедуры состоит в том, что все формальные параметры заменяются соответствующими фактическими. После выполнения процедуры происходит передача управления в основную программу, т.е. начинает выполняться оператор, следующий за оператором вызова процедуры.

Фактические параметры — это переменные (или значения заданные явно), которые передаются в процедуры на место формальных параметров. Если в вызываемой процедуре отсутствует список формальных параметров, то список фактических параметров тоже отсутствует.

Количество фактических параметров должно соответствовать количеству формальных параметров; соответствующие фактические и формальные параметры должны совпадать по порядку записи и по типу данных.

Описание, определение и вызов функции

Оформляется функция аналогично процедуре. Отличительной особенностью

функции является то, что она возвращает только один результат выполнения. Этот результат обозначается именем функции и возвращается (передается) в основную программу (место вызова). Функция состоит из заголовка, раздела описаний и раздела операторов.

```
Function <Name>(<Список формальных параметров>):<Type>;  
... {Раздел описаний}  
Begin  
... {Раздел операторов процедуры}  
Name:=<выражение соответствующего типа>;  
...  
End;
```

где Function - служебное слово;

Name - произвольный идентификатор, определяющий имя функции. В отличие от процедур в разделе операторов тела функции обязательно должен быть хотя бы один оператор присвоения имени функции выражения или значения соответствующего типа. После работы функции результат присваивается имени функции.

Таким образом, алгоритм можно оформить в виде функции в том случае, если в качестве результата получается одно единственное значение. Для вызова функции достаточно указать ее имя (с фактическими параметрами) в любом выражении, где тип результата функции будет приемлем. Имя функции можно использовать в арифметических выражениях и других командах.

Пример 1. Разработать функцию, определяющую по двум катетам гипотенузу прямоугольного треугольника.

```
Function Gepoten(a,b:real):real;  
Begin  
Gepoten:=Sqrt(Sqr(a)+Sqr(b))  
End;
```

Вызов функции из основной программы может выглядеть следующим образом: $z:=Gepoten(x, y)$; {z присваивается значение гипотенузы} или `WriteLn('Значение гипотенузы', Gepoten(x, y))`;

Передача параметров в подпрограммы

Передача параметров в подпрограмму может осуществляться несколькими способами.

Параметры процедур и функций могут быть следующих видов: параметры-значения, параметры-переменные, параметры-константы и не типизированные параметры.

Группа параметров без предшествующего ключевого слова является списком параметров-значений

Группа параметров, перед которыми следует ключевое слово Const и за которыми следует тип, является списком параметров-констант.

Группа параметров, перед которыми стоит ключевое слово Var и за которыми следует тип, является списком типизированных параметров-переменных.

Группа параметров, перед которыми стоит ключевое слово Var или Const за которыми не следует тип, является списком не типизированных параметров-переменных.

Передача параметров по значению

При передаче параметров по значению в формальный параметр передается копия значения соответствующего фактического параметра. Примерами параметров-значений служат параметры A, B и C в процедуре с заголовком MyProc. В этом случае фактическим параметром, соответствующим A, либо B, либо C, может быть любое выражение или переменная типа Real.

Для параметров-значений компилятор при вызове процедуры выделяет место в

сегменте стека (специальная область оперативной памяти) для каждого формального параметра, вычисляет значение фактического параметра и передает его в ячейку, соответствующую формальному параметру, выполняет тело процедуры. После завершения работы процедуры, формальные параметры уничтожаются, а фактические остаются неизменными (по значению).

Формальный параметр-значение обрабатывается, как локальная по отношению к процедуре или функции переменная, за исключением того, что он получает свое начальное значение из соответствующего фактического параметра при активизации процедуры или функции.

Соответствующее фактическое значение параметра-значения должно быть выражением и его значение не должно иметь файловый тип или какой-либо структурный тип, содержащий в себе файловый тип. Фактический параметр должен иметь тип, совместимый по присваиванию с типом формального параметра-значения. Если параметр имеет строковый тип, то формальный параметр будет иметь атрибут размера, равный 255.

Приступая к решению задач этого раздела, следует вспомнить, что:

- для передачи информации в процедуру следует использовать параметры, а не глобальные переменные, т. е. объявленные вне процедуры;

- тип каждого фактического параметра (константы или переменной) в инструкции вызова процедуры должен соответствовать типу соответствующего формального параметра, указанного при объявлении функции;

- если в инструкции объявления процедуры перед именем формального параметра нет слова `var`, то в качестве формального параметра в инструкции вызова процедуры можно использовать константу или переменную соответствующего типа. Если слово `var` присутствует в инструкции, то формальным параметром можно назначить только переменную;

- если аргумент процедуры применяется для возврата результата в программу, вызвавшую эту процедуру, то перед именем аргумента нужно поставить слово `var`.

Пример 1

Программа вычисления степени по вводимым пользователем значением числа и показателя степени.

```
program Func;
uses crt;
var
a,z,r: integer;
m:integer;
{Функция вычисления степени, N, X – формальные параметры}
function Stepen(n:integer; x:integer):integer;
var
i:integer;
y: integer;
52
begin
y:=1;
for I:=1 to n do {Цикл вычисления n-ой степени числа x}
y:=y*x;
stepen:=y {Присваивание функции результата вычисления степени}
end; {Конец функции}
Begin
clrscr;
writeln('vvedite znachenie chisla a i pokazatel stepeni m');
```

```

readln(a);
readln(m);
z:=Stepen(m,a);
Writeln('z=', z:3);
end.

```

Пример № 2

Даны два натуральных числа a и b . Требуется определить наибольший общий делитель трех величин. Определить наибольший общий делитель трех величин $a+b$, $|a-b|$, $a*b$

Идея решения состоит в следующем математическом факте: $\text{НОД}(x, y, z) = \text{НОД}(\text{НОД}(x,y),z)$

```

Program NOD;
Uses crt;
var
a,b,c: integer;
Procedure Evklid (M,N: integer; Var k: integer); {m,n – формальные параметры
(параметрыаргументы, k – параметр-результат)}
Begin
while m<>n Do
If m>n
then m:=m-n else n:=n-m;
k:=m
End;
Begin
clrscr;
write('a=');
readln(a);
Write('b=');
readln(b);
Evklid(a+b, ABC(a-b), a*b);
Evklid(c, a*b, c);
Writeln('NOD=',c)
End.

```

ЗАДАНИЕ

Напишите программу на языке программирования, используя процедуры и функции, по варианту, предложенному преподавателем.

Вариант 1

Дана целочисленная квадратная матрица 4×4 . Определить:

- 1) произведение элементов во второй строке (оформить в виде функции)
- 2) сумму элементов главной диагонали (оформить в виде процедуры)
- 3) составить блок-схемы

Вариант 2

Дана целочисленная матрица размером 4×3 . Определить:

- 1) Количество нулевых элементов в 4 строке. (Оформить в виде функции).
- 2) Максимальный элемент матрицы (оформить в виде процедуры)
- 3) составить блок-схемы

Вариант 3

Дана целочисленная прямоугольная матрица. Определить:

- 1) Сумму положительных элементов матрицы (оформить в виде функции)
- 2) Минимальный элемент матрицы (оформить в виде процедуры)
- 3) составить блок-схемы

Вариант 4

Дана целочисленная квадратная матрица 3×3 . Определить:

- 1) количество отрицательных элементов матрицы (оформить в виде функции)
- 2) количество положительных элементов в 3 строке (оформить в виде процедуры)
- 3) составить блок-схемы

Вариант 5

1. Сформировать одномерный массив B с помощью генератора случайных чисел в интервале от 0 до 50 (оформить в виде процедуры).

2. Подсчитать для сформированного массива B сумму положительных элементов (оформить в виде функции).

3. Составить блок-схемы

Вариант 6

1. Сформировать многомерный массив размерностью 3×3 , ввод элементов осуществлять с клавиатуры (оформить в виде процедуры).

2. Подсчитать для сформированного массива произведение элементов главной диагонали (оформить в виде функции).

3. Составить блок-схемы

Вариант 7

1. Сформировать многомерный массив T размерностью 4×2 с помощью генератора случайных чисел в интервале от 0 до 100 (оформить в виде процедуры)

2. Посчитать для сформированного массива T сумму элементов первой строки (оформить в виде функции)

3. Составить блок-схемы

Вариант 8

1. Сформировать одномерный массив D , состоящий из 10 элементов с помощью генератора случайных чисел в интервале от 0 до 200 (оформить в виде процедуры)

2. Подсчитать для сформированного массива D среднее арифметическое (оформить в виде функции)

3. Составить блок-схемы

Вариант 9

Дана целочисленная квадратная матрица 4×4 . Определить:

1) произведение положительных элементов матрицы (оформить в виде функции)

2) минимальный элемент во второй строке (оформить в виде процедуры)

3) составить блок-схемы

Вариант 10

Дана целочисленная прямоугольная матрица. Определить:

1) сумму отрицательных элементов (оформить в виде функции)

2) максимальный элемент в первой строке

3) составить блок-схемы

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое подпрограмма?
2. Что такое процедура?
3. Что такое функция?
4. Что такое фактический параметр?
5. Что такое формальный параметр?
6. Как фактические параметры должны соответствовать формальным параметрам?
7. Чем описание процедуры отличается от описания функции?
8. Как осуществляется вызов процедуры?
9. Как осуществляется вызов функции?

Практическая работа №2

Тема: ЗАПИСИ. КОМБИНИРОВАННЫЙ ТИП ДАННЫХ.

Цель. Изучить записи и комбинированный тип данных.

Оборудование. ПК

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Запись представляет собой объединение фиксированного числа логически связанных компонентов, называемых полями. Компоненты записи различаются между собой именами полей и могут относиться к разным типам.

Тип запись описывается по схеме:

Type < имя типа записи >=record

<Имя поля 1>: тип 1;

<Имя поля2>: тип 2;

.....

<Имя поля n>: тип n;

End;

Опишем тип Abitur – сведения об абитуриенте (фамилия, имя, отчество, оценки по 3 предметам, средний балл).

Type Abitur = Record

Fam, Im, Otch: String[20];

Pr1,Pr2,Pr3:1..5;

Sroc: Real;

End;

Var Ab: Abitur;

Записи предоставляют возможность прямого доступа к любому полю. Доступ к отдельным полям переменных входящих в запись, осуществляется с помощью составных имен, включающих в себя имя записи и имя ее компонента (поля), разделенных точкой. Например, для полей переменной Ab типа Abitur можно использовать операторы:

Redln(Ab.Fam,Ab.Im);

Ab.Pr1:=4;

Ab.Pr2:=5;

Ab.Pr3:=5;

Ab.Sroc:=(Ab.Pr1+ Ab.Pr2+ Ab.Pr3);

При неоднократном обращении к одному и тому же полю записи или к нескольким полям одной и той же записи удобно воспользоваться оператором присоединения. Общий вид записи оператора присоединения:

With v Do s;

Где v – имя записи, s – оператор (простой или составной).

Используя оператор присоединения к предыдущему примеру, можно записать:

With Ab Do

Begin

Redln(Fam,Im);

Pr1:=4;

Pr2:=5;

Pr3:=5;

Sroc:=(Pr1+ Pr2+Pr3);

End;

Оператор присоединения позволяет использовать вместо составных имен

непосредственно имена полей. При этом имя записи выносится в заголовок оператора присоединения. Внутри оператора к полям записи обращаются только по имени поля, а транслятор подставляет имя записи.

Пример. Составить программу для решения задачи.

В Группе 30 обучающихся. По известным фамилиям и оценкам по 4 предметам найти среднюю оценку каждого обучающегося и выдать на печать фамилию и средний балл лучшего обучающегося.

Для решения задачи потребуется массив, в котором будут храниться сведения обо всех обучающемсях группы, причем эти сведения будут иметь вид записи.

```
Program zapis;  
Type  
Sved=record  
Fam:string[25];  
p1,p2,p3,p4:1..5;  
sroc:Real;  
End;  
Var spisok: Array[1..30] of sved;  
I,n: Integer;  
champ: Real;  
Begin  
    {Ввод сведений о обучающемсях}  
    For I:=1 To 30 Do  
        With spisok[I] Do  
            Begin  
                Writeln('Введите фамилию обучающегося ');  
                Readln(Fam);  
                Writeln(' Введите оценки по 4 предметам');  
                Readln(p1,p2,p3,p4);  
            End;  
        For I:=1 To 30 Do  
            {Нахождение среднего балла обучающегося}  
            With spisok[I] Do  
                sroc:=(p1+p2+p3+p4)/4;  
            champ:=0; {Нахождение максимального среднего балла}  
            For I:=1 To 30 Do  
                If spisok[I].sroc>=champ Then champ:= spisok[I].sroc;  
            {Печать списка обучающихся с максимальным средним баллом}  
            For I:=1 To 30 Do  
                If spisok[I].sroc=champ Then  
                    With spisok[I] Do  
                        Writeln(Fam:30,' ',sroc:8:3);  
                    Readln;  
                End.  
    End.
```

Варианты заданий.

1. В отделе работает восемь человек. Вывести на экран следующие данные: фамилию, имя, отчество, стаж работы работников, чей возраст не превышает 30 лет. Исходные данные ввести с клавиатуры.

2. Сотрудник налоговой инспекции оштрафовал за день шесть человек. Вывести на экран фамилии, номера машин, сумму штрафа для водителей, оштрафованных более чем на 200 рублей и общую сумму штрафов. Исходные данные ввести с клавиатуры.

3. Расчетная ведомость содержит данные о фамилии, табельном номере и сумме заработка за январь и февраль семи работников. Вывести на экран все данные о работниках, чей январский заработок превышает февральский. Исходные данные ввести с клавиатуры.

4. Акт инвентаризации материальных ценностей содержит шесть наименований предметов (шкаф, стол и т.д.), их количество и стоимость. Вывести на экран данные акта и суммарную стоимость по наименованиям. Исходные данные ввести с клавиатуры.

5. Реестр поступлений за коммунальные услуги за год жильцов восьмиквартирного дома содержит лицевые счета и суммы поступлений за отопление, газ и электроэнергию. Вывести на экран данные реестра и общую сумму оплаты по каждому счету. Исходные данные ввести с клавиатуры.

6. Ведомость заработной платы содержит фамилии десяти работников, их табельные номера, сумму зарплаты, сумму премии, составляющей 20% от зарплаты и сумму к выдаче. Подсчитать сумму премии и сумму к выдаче, остальные данные ввести с клавиатуры. Вывести на экран ведомость.

7. Журнал отгрузки готовой продукции содержит наименования шести видов продукции, количество, цену за единицу продукции, суммарную стоимость по наименованиям. Вычислить суммарную стоимость, остальные данные ввести с клавиатуры. Вывести на экран все данные журнала.

8. Реестр актов на покупку продуктов содержит наименование десяти продуктов. Ввести с клавиатуры стоимость за 1 кг, количество продукта в кг. Найти общую стоимость по наименованиям. Вывести на экран все данные реестра.

9. Ведомость затрат по шести заказам содержит номер заказа, фамилию, сумму затрат по трем статьям расходов и общую сумму затрат по каждому заказу. Вычислить общую сумму затрат, остальные данные ввести с клавиатуры. Вывести на экран все данные ведомости.

10. Ведомость выработки по пяти бригадам за три месяца содержит наименование месяца, сумму выработки по бригадам за месяц, общую сумму выработки за месяц. Вычислить общую сумму выработки, остальные данные ввести с клавиатуры. Вывести на экран все данные ведомости.

11. Ведомость расхода материалов на строительство содержит шесть наименований материалов, расход материала в рублях, допустимые нормы расхода. Эти данные ввести с клавиатуры. Вывести на экран наименование, расход и сумму расхода сверх нормы для материалов, по которым допущен перерасход.

12. Выработка статистических данных содержит информацию о фамилии, сумме доходов и количестве членов семьи и десяти работников предприятия. Эти данные ввести с клавиатуры. Вычислить сумму доходов, приходящуюся на одного члена семьи каждого работника. Вывести на экран все имеющиеся данные.

13. Расчетная ведомость содержит данные о фамилии, табельном номере, сумме зарплаты и подоходном налоге восьми работников отдела. Вычислить сумму налога, составляющего 13% от зарплаты, остальные данные ввести с клавиатуры. Вывести на экран все данные ведомости.

14. Ведомость расходов материалов на строительство содержит данные о наименовании материала, количестве израсходованного материала, стоимости единицы материала. Вычислить сумму расходов на закупку материала и общую сумму расходов. Вывести на экран все данные ведомости.

15. Расчетная ведомость бригады за три месяца содержит информацию о выработке бригады в рублях, суммарной заработной плате. Эти данные ввести с клавиатуры. Если выработка бригады превышает суммарную зарплату, то бригаде выплачивается премия в размере 10 % выработки. Вывести на экран информацию за июль, август, сентябрь о выработке бригады, суммарной заработной плате, премии (если премия не начисляется выводится 0) и общей сумме заработка.

Практическая работа №3

Тема: ДИНАМИЧЕСКИЕ СТРУКТУРЫ И УКАЗАТЕЛИ..

Цель. Изучить динамические структуры и указатели..

Оборудование. ПК

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

В Паскале существует два способа выделения памяти под данные: статистический и динамический.

Статистическими называются такие величины, память под которые выделяются во время компиляции и сохраняются в течение всей работы программы. Под динамические данные память отводится во время выполнения программы.

Использование динамических величин позволяет увеличить объем обрабатываемых данных, создавать структуры данных переменного размера. Если потребность в каких то данных отпала до окончания программы, то занятую ими память можно освободить для другой информации.

Работа с динамическими величинами связана с использованием ссылочного типа. Величины, имеющие ссылочный тип, называют указателями.

Указатель содержит адрес поля в динамической памяти, хранящего величину определенного типа. Сам указатель располагается в статической памяти.

Адрес величины – это номер первого байта поля памяти, в котором располагается величина. Размер поля однозначно определяется типом.

Величина ссылочного типа (указатель) описывается в разделе описания переменных следующим образом:

```
Var <идентификатор>:<ссылочный тип>;
```

В стандарте Паскаля каждый указатель может ссылаться на величину только одного определенного типа, который называется базовым для указателя. Имя базового типа и указывается в описании в следующей форме:

```
<ссылочный тип>:=^<имя типа>;
```

Пример описания указателей:

```
Type Massiv=Array[1..100] Of Integer;
```

```
Var P1: ^Integer;
```

```
    P2: ^Char;
```

```
    PM: ^Massiv;
```

Здесь P1 – указатель на динамическую величину целого типа;

P2 – указатель на динамическую величину символьного типа;

PM – указатель на динамический массив, тип которого задан в разделе Type.

Сами динамические величины не требуют описания в программе, поскольку во время компиляции память под них не выделяется.

Память под динамическую величину, связанную с указателем, выделяется в результате выполнения стандартной процедуры **NEW**.

Формат обращения к этой процедуре:

```
NEW(<указатель>);
```

Считается, что после выполнения этого оператора создана динамическая величина, имя которой имеет следующий вид:

```
<имя динамической величины>:=<указатель>^;
```

Пусть в программе, в которой имеется приведенное выше описание, присутствуют операторы:

```
NEW(P1); NEW(P2); NEW(PM);
```

После их выполнения в динамической памяти оказывается выделенным место под три величины, которые имеют идентификаторы:

```
P1^, P2^, PM^
```

Например, обозначение P1^ можно расшифровать так: динамическая переменная, на которую ссылается указатель P1.

Работа с динамическими переменными происходит точно так же, как со статическими переменными соответствующих типов. Им можно присваивать значения, их можно использовать в качестве операндов в выражениях, параметров подпрограммы и т.п.

Например:

```
P1^:=24;
```

```
P2^:='W';
```

```
For I:=1 To 100 Do P1^[I]:=I;
```

Кроме процедуры NEW значение указателя может определяться оператором присваивания:

```
<указатель>:=<>;
```

В качестве ссылочного выражения можно использовать:

Указатель;

Ссылочную функцию (т.е. функцию, значением которой является указатель);

Константу NIL.

NIL – это зарезервированная константа, обозначающая пустую ссылку, т.е. ссылку, которая ни на что не указывает. При присваивании базовые типы указателя и ссылочного выражения должны быть одинаковыми. Константу NIL можно присваивать указателю с любым базовым типом.

До присваивания значения ссылочной переменной (с помощью оператора присваивания или процедуры NEW) она является неопределенной.

Ввод и вывод указателей не допускается.

Если динамическая величина теряет свой указатель, то она становится «мусором» (информация, которая занимает память, но уже не нужна).

В Паскале имеется стандартная процедура, позволяющая освобождать память от данных, потребность в которых отпала. Ее формат:

```
Dispose(<указатель>);
```

Например, если динамическая переменная P^ больше не нужна, то оператор

```
Dispose(P);
```

Удалит ее из памяти. После этого значение указателя P становится неопределенным. Особенно существенным становится эффект экономии памяти при удалении больших массивов.

Работа с динамическими данными замедляет выполнение программы, поскольку доступ к величине происходит в два шага: сначала ищется указатель, затем по нему – величина. Т.е. выигрыш в памяти компенсируется проигрышем во времени.

Пример1. Создать вещественный массив из 10000 чисел, заполнить его случайными числами в диапазоне от 0 до 1. Вычислить среднее значение массива. Очистить динамическую память. Создать целочисленный массив размером 10000, заполнить его случайными целыми числами в диапазоне от -100 до 100 и вычислить его среднее значение.

```
Program Sr;
```

```
Const Nmax=10000;
```

```
Type Diapazon=1..Nmax;
```

```
MasInt=Array[Diapazon] Of Integer;
```

```
MasReal=Array[Diapazon] Of Real;
```

```
Var PInt: ^MasInt;
```

```
PReal:=MasReal;
```

```
I,Midint: LongInt;
```

```
MidReal: Real;
```

```
Begin
```

```
MidInt:=0; MidReal:=0;
```

```
Randomize;
```

```
New(PReal); {Выделение памяти под вещественный массив}
```



```

{Вычисление и суммирование массива}
For I:=1 To NMax Do
Begin
PReal^[I]:=Random;
MidReal:=MidReal+PReal^[I];
End;
Dispose(PReal); {Удаление вещественного массива}
New(PInt); {Выделение памяти под целочисленный массив}
{Вычисление и суммирование целочисленного массива}
For I:=1 To NMax Do
Begin
Pint^[I]:=Random(200)-100;
MidInt:=MidInt+Pint^[I];
End;
Dispose(Pint); {Удаление целочисленного массива}
{Вывод средних значений}
Writeln('Среднее целое равно:',MidInt Div NMax);
Writeln('Среднее вещественное равно:',(MidReal Div NMax):10:6);
End.

```

Связанные списки.

Связанный список схематически выглядит так:

$x_1|P_2 \rightarrow x_2|P_3 \rightarrow x_3|P_4 \rightarrow x_4|Nil$

Каждый элемент списка состоит из двух частей: информационной части (x_1, x_2 и т.д.) и указателя на следующий элемент списка (P_1, P_2 и т.д.).

Последний элемент имеет пустой указатель Nil.

Связанный список обладает тем замечательным свойством, что его можно дополнить по мере поступления новой информации. Добавление происходит путем присоединения нового элемента к концу списка. Значение Nil в последнем элементе заменяется ссылкой на новый элемент цепочки:

$x_1|P_2 \rightarrow x_2|P_3 \rightarrow x_3|P_4 \rightarrow x_4|P_5 \rightarrow x_5|Nil$

Связанный список не занимает лишней памяти. Память расходуется в том объеме, который требуется для поступившей информации. В программе для представления элементов цепочки используется комбинированный тип (запись).

Пример.

```

Type PE=^Elem;
Elem=Record
T: Real;
P:PE;
End;

```

PE - ссылочный тип на переменную типа Elem. Этим именем обозначен комбинированный тип, состоящий из двух полей: T – вещественная величина, P – указатель на динамическую величину типа Elem.

Пример 2. Сформировать связанный список в ходе ввода данных.

```

PROGRAM PD;
Type PE=^Elem;
Elem=Record T: Real; P:PE; End;
VAR Elem, Beg: PE; x: Real; ch: Char;
Begin

```

```

{Определение адреса начала списка и его сохранение}

```

```

NEW(Elem); Beg:=Elem;

```

```

{Диалоговый ввод значений с занесением их в список и организацией связи между
элементами}

```

```

While Elem^.P<> Nill Do
Begin
Writeln('Введите число:'); Readln(Elem^.T);
Writeln(' Повторить ввод ? (Y/N)'); Readln (ch);
If (ch='n') OR (ch='N') Then Elem^.P:=Nill Else Begin
NEW(Elem^.P); Elem:=Elem^.P); End;
Writeln(' Ввод данных закончен ');
{Вывод полученной числовой последовательности}
Elem:=Beg;
Repeat
Writeln(Elem^.T:8:3);
Elem:=Elem^.P
Until Elem=Nill;
End.

```

Ссылочная переменная Beg используется для сохранения адреса начала цепочки. Всякая обработка цепочки начинается с первого элемента.

Варианты заданий.

1. Составить программу занесения в динамическую память вещественного массива из 10000 чисел, хранящихся в файле на магнитном диске, а также поиска в нем значения и номера первого максимального элемента.
2. Связанный список представляет собой символьную цепочку из строчных латинских букв. Описать процедуру или функцию, которая определяет, является ли список пустым.
3. Написать программу, которая вставляет в список L новый элемент за каждым вхождением элемента E.
4. Составить программу, которая удаляет из списка L все элементы E, если таковые имеются.
5. Связанный список представляет собой символьную цепочку из строчных латинских букв. Описать процедуру или функцию, которая меняет местами первый и последний элементы непустого списка.
6. Описать процедуру, которая удаляет из непустого списка L первый элемент.
7. Описать процедуру, которая удаляет из непустого списка L последний элемент.
8. Описать процедуру, которая удаляет из непустого списка L первый неотрицательный элемент, если такой есть.
9. Описать процедуру, которая удаляет из непустого списка L второй элемент.
10. Написать программу. Заданный во входном файле текст (за ним следует точка) распечатать в обратном порядке.
11. Написать программу. Дана непустая последовательность натуральных чисел, за которой следует 0. Напечатать порядковые номера тех чисел последовательности, которые имеют наибольшую величину.
12. Дано целое $n > 0$, за которым следует n вещественных чисел. Напечатать эти числа в порядке их убывания.
13. Описать процедуру или функцию, которая проверяет на равенство списки L1 и L2.
14. Описать процедуру или функцию, которая добавляет в конец списка L1 все элементы списка L2.
15. Описать процедуру или функцию, которая переносит в начало непустого списка L его последний элемент.

Практическая работа №4

Тема: ОСНОВЫ КОМПЬЮТЕРНОЙ ГРАФИКИ.

Цель. Изучить основы компьютерной графики.

Оборудование. ПК

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Система программирования Паскаль снабжена несколькими модулями или библиотеками, в которых хранятся все ее процедуры и функции.

Библиотека GRAPH.TPU содержит процедуры и функции для поддержки графического режима.

Подключение графической библиотеки: **USES GRAPH;**

Инициализация (включение) графического режима.

Экран может работать в текстовом или графическом режимах.

Включение графического режима:

Procedure InitGraph(Gd, Gm: Integer; Path: String);

Параметр Gd является кодом графического адаптера (т.е. электронной схемы, управляющей выводом информации на экран). (Графические адаптеры: GGA, EGA, VGA).

Каждый графический адаптер позволяет использовать несколько графических режимов, отличающихся количеством цветов и разрешающей способностью.

Параметр Gm предназначен для того, чтобы указать какой из режимов следует включить.

Параметр Path содержит путь к файлу, который называется Egavga.bgi. В этом файле содержится драйвер, необходимый для работы монитора в графическом режиме. Если файл находится в текущем каталоге, указывают ' '.

Возможно также автоматическое определение типа драйвера и установка режима. Этот прием позволяет программе работать с разными типами мониторов.

Gd:=Detect;

InitGraph(Gd, Gm, 'C:\TP\BGI');

При этом автоматически устанавливается режим с наибольшей разрешающей способностью и цветностью.

Функция GraphRezult: Integer; возвращает специальный код, в зависимости от того, как прошло выполнение любой графической процедуры или функции.

Код 0 (grOk) – успешное выполнение.

Выключение графического режима:

Procedure CloseGraph;

Пример программы, которая рисует отрезок прямой на экране.

Program Ex1;

Uses Graph;

Var Gd, Gm: Integer;

Begin

Gd:=VGA;

Gm:=VGAhi;

Initgraph(Gd,Gm, ' ');

If GraphRezult=grOk then

Begin Line(0,0,639,475);

Readln; CloseGraph;

End;

End.

Система координат.

Каждая точка на экране - пиксел. Количество пикселов уместающихся на экране по вертикали и горизонтали, называют разрешающей способностью. Разрешающая

способность экрана в режиме VGAhi – 640x480. Координаты отсчитываются с левого верхнего угла экрана. Можно определить максимальные координаты по осям, соответствующие данному драйверу, с помощью функций:

Function GetMaxX;

Function GetMaxY;

Некоторые графические процедуры и функции:

Procedure SetColor(Color: Word); { Установка цвета линий }

Procedure SetBkColor(Color: Word); { Установка цвета фона }

Таблица цветов кодов.

| Имя константы | Номер цвета | Цвет |
|---------------|-------------|------------------|
| Black | 0 | Черный |
| Blue | 1 | Темно-синий |
| Green | 2 | Темно-зеленый |
| Cyan | 3 | Бирюзовый |
| Red | 4 | Красный |
| Magenta | 5 | Фиолетовый |
| Brown | 6 | Коричневый |
| LightGray | 7 | Светло-серый |
| DarkGray | 8 | Темно-серый |
| LightBlue | 9 | Синий |
| LightGreen | 10 | Светло-зеленый |
| LightCyan | 11 | Светло-бирюзовый |
| LightRed | 12 | Розовый |
| LightMagenta | 13 | Малиновый |
| Yellow | 14 | Желтый |
| White | 15 | Белый |

Procedure ClearDevice; { Очистка экрана }

Область вывода изображения может быть ограничена любым прямоугольником в пределах экрана. Такая область называется графическим окном.

Procedure SetViewPort(X1,Y1,X2,Y2:Integer; Clip:Boolean); { Устанавливает положение графического окна. (X1,Y1) – координаты левого верхнего угла окна, (X2,Y2) – координаты правого нижнего угла, Clip – ограничитель фигур; если Clip=True, то все построения производятся только в пределах окна, в противном случае они могут выходить за его пределы. }

Procedure MoveTo(X,Y: Integer); { Перемещает курсор в точку с координатами (X,Y). }

Procedure PutPixel(X,Y: Integer; Color: Word); { Выставляет точку на графическом экране. }

X,Y – координаты точки, Color – цвет точки. }

Пример. Составить программу которая устанавливает по центру экрана графическое окно размером 100x100, заливает его желтым фоном и заполняет синими точками, расположенными через 4 позиции.

Program GR;

Uses Graph;

Var Gd, Gm: Integer;

X,Y,X1,Y1,X2,Y2, Xc, Yc: Integer;

Begin

{Инициализация графического режима}

Gd:=Detect;

InitGraph(Gd, Gm, ‘ ‘);

{Определение координат центра экрана}

```

Xc:=GetMaxX Div 2;
Yc:=GetMaxY Div 2;
{Определение координат графического окна}
X1:=Xc-50;
Y1:=Yc-50;
X2:=Xc+50;
Y2:=Yc+50
{Установка графического окна}
SetViewPort(X1,Y1,X2, Y2, True);
{Установка цвета фона и очистка экрана}
SetBkColor(Yellow);
ClearDevice;
{Расстановка точек в окне}
For X:=1 To 25 Do
For Y:=1 To 25 Do
PutPixel(4*X, 4*Y, Blue);
{Задержка изображения на экране до нажатия <ENTER>}
Readln;
{Выход из графического режима}
CloseGraph;
END.

```

Графические примитивы.

В любом графическом пакете существуют процедуры рисования основных геометрических фигур: прямых линий, окружностей, эллипсов, прямоугольников и т.п. Такие фигуры называются графическими примитивами.

Основные процедуры рисования графических примитивов, имеющихся в модуле Graph:

Procedure Line(X1,Y1,X2,Y2: Integer): {Рисует линию с заданными координатами концов (X1,Y1) и (X2,Y2)}

Procedure LineTo(X,Y: Integer); {Линия от текущей точки до точки с координатами X,Y}

Procedure LineRel(DX,DY: Integer); Линия от текущей точки до точки с заданными приращениями координат DX,DY

Procedure Rectangle(X1,Y1,X2,Y2: Integer); Прямоугольник с заданными координатами верхнего левого угла (X1,Y1) и нижнего правого угла (X2,Y2)

Procedure Circle(X,Y: Integer; R: Word); {Окружность с центром в точке (X,Y) и радиусом R – в пикселях}

Procedure Arc(X,Y: Integer; BegA, EndA,R: Word); {Дуга окружности с центром в точке (X,Y), радиусом R, начальным углом BegA и конечным углом EndA. Углы измеряются в градусах против часовой стрелки от направления оси X.}

Procedure Ellipse(X,Y: Integer; BegA, EndA, RX, RY: Word); {Эллиптическая дуга с центром в точке X,Y с начальным и конечным углами BegA и EndA, горизонтальным радиусом RX и вертикальным радиусом RY.}

Procedure SetLineStyle(Ln,P,T: Word); {Задание параметров линии. Ln – стиль линии: 0-сплошная, 1-пунктирная, 2-штрихпунктирная, 3 – штриховая, 4 – заданная пользователем; P шаблон: 0 – во всех случаях для Ln от 0 до 3; T –толщина линии: 1 – нормальная толщина, 3 – толстая линия}

Закраска и заполнение.

Среди графических примитивов существуют закрашенные области. Цвет закрашки определяется процедурой SetColor. Кроме того, можно управлять рисунком закрашки (типом заполнения). Это может быть сплошная закрашка, заполнение редкими точками, крестиками, штрихами и т.д.

Procedure SetFillStyle(Fill,Color:Word); { Определяет тип заполнения (Fill) и цвета заполнения (Color)}.

Таблица констант орнамента заполнения(для процедуры SetFillStyle)

| Имя | Значение | Назначение |
|----------------|----------|---------------------------------|
| EmptyFill | 0 | Заполнение цветом фона |
| SolidFill | 1 | Однородное заполнение цветом |
| LineFill | 2 | Заполнение – |
| LtSlashFill | 3 | Заполнение /// |
| SlashFill | 4 | Заполнение /// толстыми линиями |
| BkSlashFill | 5 | Заполнение \\ толстыми линиями |
| LtBkSlashFill | 6 | Заполнение \\\ |
| HatchFill | 7 | Заполнение клеткой |
| XHatchFill | 8 | |
| InterleaveFill | 9 | |

Procedure Bar(X1,Y1,X2,Y2: Integer); {Заполняет прямоугольную область с заданными координатами углов}

Procedure FillEllips(X,Y,RX,RY: Integer);{Обведенный линией, закрашенный эллипс}

Procedure Sector(X,Y,: Integer; BegA,EndA,RX,RY: Word); {Обведенный линией и закрашенный эллипсный сектор}

Procedure PieSlice_X,Y: Integer; BegA, EndA: Word); {Обведенный линией и закрашенный сектор окружности}

Procedure FloodFill(X,Y: Integer; Border: Word); {Закрашивает любую область, ограниченную замкнутой линией. (X,Y) – любая точка внутри области, Border –цвет граничной линии}

Модуль Graphе позволяет выводить на графический экран тексты.

Procedure OutTextXY(X,Y: Integer; Txt: String); {Выводит в графическое окно символьную строку Txt, начиная с указанной позиции (X,Y)}

Варианты заданий.

Составить программу с помощью которой можно:.

1. Нарисовать разноцветную мишень.
2. Нарисовать круговую диаграмму, состоящую из 10 заполненных секторов, используя различные орнаменты и цвета заполнения.
3. Нарисовать заполненные различным орнаментом и цветом заполнения треугольник и прямоугольник
4. Нарисовать на белом фоне закрашенную синим цветом трапецию.
5. Нарисовать на экране шахматное поле.
6. Нарисовать свои инициалы в виде заполненных многоугольников.
7. Изобразить горизонтальную последовательность состоящую из 16 различных заполненных окружностей.
8. Изобразить горизонтальную последовательность состоящую из 16 различных заполненных эллипсов.
9. Нарисовать картинку «автомобиль».
10. Нарисовать картинку «робот»
11. Нарисовать разноцветную ромашку, используя процедуры рисования окружности и эллипса.
5. Нарисовать окружность с заключенным в него треугольником.
6. Нарисовать окружность с заключенным в него прямоугольником.
7. Нарисовать окружность с заключенной в него трапецией.
8. На белом фоне изобразить шаблон вашего индекса так, как он выглядит на почтовом конверте.

**Вопросы для устного опроса
по дисциплине «Информатика и программирование»**

1. Общие принципы организации и работы компьютера.
2. Классификация ЭВМ. Поколения ЭВМ.
3. Персональный компьютер. Архитектура современного персонального компьютера. Разновидности программ для компьютеров.
4. Основные компоненты ПК. Периферийные устройства ПК.
5. Операционные системы WINDOWS **, WINDOWS 20**. Общие сведения. Работа с файлами, каталогами, папками.
5. Текстовый редактор Microsoft Word. Назначение и основные функции.
6. Электронные таблицы Microsoft Excel. Назначение и основные функции.
7. Системы управления базами данных. Назначение и основные функции.
8. Аппаратура компьютера.
9. Системное программное обеспечение. Основные понятия и определения. Операционные системы DOS, WINDOWS.
10. Прикладное программное обеспечение и его назначение.
11. Понятие информации. Общее представление об информации. Понятие носителя информации. Формы представления и передачи информации.
12. Этапы решения. Задачи на ЭВМ. Виды алгоритмов, их свойства. Алгоритмизация при решении задач.
13. Алфавит языка Pascal. Константы и переменные языка Pascal.
14. Основные операторы языка.
15. Структура программного модуля.
16. Простейшие конструкции языка Pascal.
17. Организация программ линейной структуры.
18. Операторы перехода.
19. Условные операторы.
20. Организация программ разветвляющейся структуры.
21. Понятие цикла. Операторы цикла.
22. Одномерные массивы. Вычисление суммы и произведения.
23. Нахождение наименьшего и наибольшего значений.
24. Вычисления в цикле с несколькими одновременно изменяющимися параметрами.
25. Вложенные циклы.
26. Двумерные массивы.
27. Подпрограммы, их виды и назначения.
28. Подпрограмма-процедура.
29. Подпрограмма-функция.
30. Комбинированный тип данных - запись.
31. Динамические структуры и указатели.
32. Основные понятия и средства компьютерной графики
33. Файлы. Файловый тип данных.
34. Комбинированный тип данных.
35. Строковый тип данных.
36. Оператор присоединения.
37. Понятие множества. Множественный тип данных.

38. Программное обеспечение сетей ЭВМ.
39. Локальные и глобальные вычислительные сети.
40. Общие сведения об Internet. Перспективы развития сети Internet.
41. Проблема защиты информации и подходы к ее решению.
42. Основные понятия защиты информации.
43. Угрозы безопасности и каналы утечки информации.
44. Классификация методов и средств защиты информации. Специфика программных средств.
45. Организация базы учетных записей пользователей в ОС Windows

**Вопросы на экзамен
по дисциплине: Информатика и программирование**

1. Общие принципы организации и работы компьютера.
2. Классификация ЭВМ. Поколения ЭВМ.
3. Персональный компьютер. Архитектура современного персонального компьютера. Разновидности программ для компьютеров.
4. Основные компоненты ПК. Периферийные устройства ПК.
5. Операционные системы WINDOWS **, WINDOWS 20**. Общие сведения. Работа с файлами, каталогами, папками.
6. Текстовый редактор Microsoft Word. Назначение и основные функции.
7. Электронные таблицы Microsoft Excel. Назначение и основные функции.
8. Системы управления базами данных. Назначение и основные функции.
9. Аппаратура компьютера.
10. Системное программное обеспечение. Основные понятия и определения. Операционные системы DOS, WINDOWS.
11. Прикладное программное обеспечение и его назначение.
12. Понятие информации. Общее представление об информации. Понятие носителя информации. Формы представления и передачи информации.
13. Этапы решения. Задачи на ЭВМ. Виды алгоритмов, их свойства. Алгоритмизация при решении задач.
14. Алфавит языка Pascal. Константы и переменные языка Pascal.
15. Основные операторы языка.
16. Структура программного модуля.
17. Простейшие конструкции языка Pascal.
18. Организация программ линейной структуры.
19. Операторы перехода.
20. Условные операторы.
21. Организация программ разветвляющейся структуры.
22. Понятие цикла. Операторы цикла.
23. Одномерные массивы. Вычисление суммы и произведения.
24. Нахождение наименьшего и наибольшего значений.
25. Вычисления в цикле с несколькими одновременно изменяющимися параметрами.
26. Вложенные циклы.
27. Двумерные массивы.
28. Подпрограммы, их виды и назначения.
29. Подпрограмма-процедура.
30. Подпрограмма-функция.
31. Комбинированный тип данных - запись.
32. Динамические структуры и указатели.
33. Основные понятия и средства компьютерной графики
34. Файлы. Файловый тип данных.
35. Комбинированный тип данных.
36. Строковый тип данных.
37. Оператор присоединения.
38. Понятие множества. Множественный тип данных.

39. Программное обеспечение сетей ЭВМ.
40. Локальные и глобальные вычислительные сети.
41. Общие сведения об Internet. Перспективы развития сети Internet.
42. Проблема защиты информации и подходы к ее решению.
43. Основные понятия защиты информации.
44. Угрозы безопасности и каналы утечки информации.
45. Классификация методов и средств защиты информации. Специфика программных средств.
46. Организация базы учетных записей пользователей в ОС Windows

СЕВЕРО-КАВКАЗСКАЯ ГОСУДАРСТВЕННАЯ АКАДЕМИЯ

Кафедра Общей информатики

20____ – 20____ учебный год

Экзаменационный билет № 1

по дисциплине: Информатика и программирование

для обучающихся направления подготовки 09.03.03 - Прикладная информатика

1. Алфавит языка Pascal. Константы и переменные языка Pascal.
2. Локальные и глобальные вычислительные сети.
3. Задача: Зашифровать и дешифровать открытый текст: Р= «Информатика и программирование» с ключом К= Фамилия (обучающегося) методом многоалфавитной подстановки на ключе К.

Зав. кафедрой

Эльканова Л.М.

СЕВЕРО-КАВКАЗСКАЯ ГОСУДАРСТВЕННАЯ АКАДЕМИЯ

Кафедра Общей информатики

20____ – 20____ учебный год

Экзаменационный билет № 2

по дисциплине: Информатика и программирование

для обучающихся направления подготовки 09.03.03 - Прикладная информатика

1. Подпрограммы, их виды и назначения.
2. Одномерные массивы. Понятия: массив, структуры данных, одномерный, двумерный массив. Объявление. Пример описания одномерного массива.
3. Задача: Задан одномерный массив $A[1..20]$. Найти сумму максимального и минимального элементов.

Зав. кафедрой

Эльканова Л.М.

Задачи на экзамен

по дисциплине Информатика и программирование

ТЕМА: АЛГОРИТМЫ ЛИНЕЙНОЙ СТРУКТУРЫ

1. Составить блок-схему и программу для вычисления значения функции по формуле (все переменные принимают действительные значения)

$$Z = \frac{\sqrt{b^2 - 4ac} + c}{2a}$$

2. Составить блок-схему и программу для вычисления значения функции по формуле (все переменные принимают действительные значения)

$$Z = \ln(y - \sqrt{|x|}) \left(x - \frac{y}{x + \frac{x^2}{4}} \right)$$

3. Составить блок-схему и программу для вычисления значения функции по формуле (все переменные принимают действительные значения)

$$Z = \ln(x^2 + y^2)$$

ТЕМА: АЛГОРИТМЫ РАЗВЕТВЛЯЮЩЕЙСЯ СТРУКТУРЫ

4. Составить блок-схему и программу для вычисления значения функции

$$F(x) = \begin{cases} x^2 - 3x + 9, & \text{если } x \leq 3; \\ \frac{1}{x^3 + 6}, & \text{если } x > 3. \end{cases}$$

5. Составить блок-схему и программу для вычисления значения функции

$$F(x) = \begin{cases} -x^2 + 3x + 9, & \text{если } x \geq 3; \\ \frac{1}{x^3 - 6}, & \text{если } x < 3. \end{cases}$$

6. Составить блок-схему и программу для вычисления значения функции

$$F(x) = \begin{cases} 9, & \text{если } x \leq -3; \\ \frac{1}{x^2 + 1}, & \text{если } x > -3. \end{cases}$$

ТЕМА. ОПЕРАТОР ВЫБОРА

7. Ввести номер недели и вывести соответствующий ему день недели на русском и английском языках. Составить программу для решения задачи.

8. Ввести номер месяца и вывести соответствующее ему название на русском языке. Составить программу для решения задачи.

9. Введите номер месяца и напечатайте соответствующее месяцу время года. Составить программу для решения задачи.

ТЕМА: АЛГОРИТМЫ ЦИКЛИЧЕСКОЙ СТРУКТУРЫ

10. Составить программу и блок-схему для вычисления значений функции $F(x)=x-\sin x$, где x изменяется на отрезке $[a,b]$ с шагом h .
11. Составить программу и блок-схему для вычисления значений функции $F(x)=\sin^2 x$, где x изменяется на отрезке $[a,b]$ с шагом h .
12. Составить программу и блок-схему для вычисления значений функции $F(x)=2\cos x-1$, где x изменяется на отрезке $[a,b]$ с шагом h .

ТЕМА: ВЫЧИСЛЕНИЕ СУММЫ И ПРОИЗВЕДЕНИЯ

13. Вычислить сумму элементов массива $X(15)$ с четными номерами. Составить блок-схему и программу.
14. Вычислить сумму элементов массива $A(20)$ с нечетными номерами. Составить блок-схему и программу.
15. Вычислить количество элементов массива $A(20)$ равных заданному числу x . Составить блок-схему и программу.
16. Вычислить произведение элементов массива $A(20)$. Составить блок-схему и программу.
17. Вычислить произведение элементов массива $X(10)$, модуль которых больше 9. Составить блок-схему и программу.
18. Вычислить произведение элементов массива $A(20)$, с четными номерами. Составить блок-схему и программу.

ТЕМА: ВЫЧИСЛЕНИЕ СУММЫ ЧЛЕНОВ БЕСКОНЕЧНОГО РЯДА

19. Дан числовой ряд $\sum_{n=1}^{\infty} a_n$ и некоторое число E . Найти сумму тех членов ряда, модуль которых больше или равен E , где $a_n = \frac{(-1)^{n-1}}{n^n}$.
20. Дан числовой ряд $\sum_{n=1}^{\infty} a_n$ и некоторое число E . Найти сумму тех членов ряда, модуль которых больше или равен E , где $a_n = \frac{1}{2^n} + \frac{1}{3^n}$.
21. Дан числовой ряд $\sum_{n=1}^{\infty} a_n$ и некоторое число E . Найти сумму тех членов ряда, модуль которых больше или равен E , где $a_n = \frac{2n-1}{2^n}$.

ТЕМА: НАХОЖДЕНИЕ МИНИМАЛЬНОГО И МАКСИМАЛЬНОГО ЭЛЕМЕНТОВ МАССИВА

22. В целочисленном массиве $B(20)$ поменять местами наибольший и наименьший элементы.
23. Дан действительный массив $A(15)$. Найти наименьший элемент из положительных элементов массива.
24. В действительном массиве $B(20)$ найти наибольший элемент из отрицательных элементов.
25. В массиве $B(20)$ найти максимальный из элементов с четными номерами.

ТЕМА: ВЫЧИСЛЕНИЕ ЗНАЧЕНИЯ ПОЛИНОМА ПО СХЕМЕ ГОРНЕРА

26. Составить блок-схему и программу для вычисления значения полинома $y = 3a^8 - 18a^5 - a^4 + 5a^3 - 2a + 1$, используя формулу Горнера.
27. Составить блок-схему и программу для вычисления значения полинома $v = 3c^{10} + 8c^9 - 4c^7 + 11c^6 - 9c^5 + 7c + 4$, используя формулу Горнера.
28. Составить блок-схему и программу для вычисления значения полинома $w = 13d^6 + 17d^5 - 8d^4 + 5d^3 + 16d^2 + 5d - 2$, используя формулу Горнера.
29. Составить блок-схему и программу для вычисления значения полинома $t = 15f^{11} + 13f^{10} + 4f^8 - 2f^7 + 16f^5 - 3f + 1$, используя формулу Горнера.
30. Составить блок-схему и программу для вычисления значения полинома $z = 14a^{12} + 12a^{11} - 8a^7 + 4a^5 - 12a^3 - 6$, используя формулу Горнера.

ТЕМА: ВЛОЖЕННЫЕ ЦИКЛЫ

31. Найти максимальный элемент массива $X(5,6)$.
32. Найти минимальный из положительных элементов действительного массива $A(4,4)$.
33. Вычислить сумму квадратов положительных элементов целочисленного массива $B(5,4)$.
34. Дана действительная матрица $X(5,5)$ и натуральное число m . Вычислить произведение тех элементов матрицы сумма индексов которых равна m .
35. Вычислить количество и сумму отрицательных элементов массива $X(5,4)$.
36. Поменять местами максимальный и минимальный элементы действительного массива $X(5,4)$.

ТЕМА: СТРУКТУРИРОВАННЫЕ ТИПЫ ДАННЫХ. МАССИВЫ.

37. Имеется N точек, расположенных в произвольном порядке на плоскости. Найти две точки, расстояние между которыми наименьшее.
38. Составить базу данных о пассажирах самолета, предусмотрев поля: Ф.И.О., багаж (вес, сумма страховки по каждому виду багажа), пункт следования.
Составить программу, позволяющую вывести
 - все данные о пассажирах,
 - список пассажиров, следующих до определенной станции,
 - список пассажиров, имеющих багаж весом выше данного.

ТЕМА: СТРОКОВЫЙ ТИП ДАННЫХ.

39. Даны целые положительные числа M , N , число D и набор из M чисел. Сформировать матрицу размера $M \times N$, у которой первый столбец совпадает с исходным набором чисел, а элементы каждого следующего столбца равны сумме соответствующего элемента предыдущего столбца и числа D (в результате каждая строка матрицы будет содержать элементы арифметической прогрессии).
40. Создать каталог из журналов и статей. Выдавать информацию о публикациях, удовлетворяющих тому или иному критерию, например, изданных с 2000 года.

ТЕМА: МНОЖЕСТВА.

41. Дана матрица размера $M \times N$. Вывести ее элементы в следующем порядке: первая строка слева направо, вторая строка справа налево, третья строка слева направо, четвертая строка справа налево и т. д.

42. В массиве хранятся данные об учащихся: школа, фамилия, класс. Вывести список учеников, которые учатся в восьмом классе.

ТЕМА: ЗАПИСИ.

43. Дан целочисленный двумерный массив, размерности $n \times m$, найти наименьший элемент массива и номер строки, в которой он находится

44. Во время лыжных соревнований в центральный судейский компьютер поступают данные в следующем виде: номер участника, его фамилия, страна и показанный результат. Составить программу, которая после ввода информации выдает таблицу результатов участников в порядке ухудшения.

ТЕМА: ПОДРОГРАММЫ В ЯЗЫКЕ ПРОГРАММИРОВАНИЯ ПАСКАЛЬ

45. Изменить последовательность столбцов матрицы так, чтобы элементы их первой строки были отсортированы по возрастанию.

46. Осуществить ввод общей информации (автор, название) о содержимом библиотеки (книги, журналы, газеты). Для книг осуществить дополнительно ввод года издания; для журналов ввести год издания и номер журнала; для газет - год, месяц и день выхода газеты. Осуществить вывод информации, поиск литературы по типу издания.

ТЕМА: ПРОЦЕДУРЫ И ФУНКЦИИ

47. Матрицу 10×20 заполнить случайными числами от 0 до 15. Вывести на экран саму матрицу и номера строк, в которых число 5 встречается три и более раз.

48. Создать файл, содержащий сведения о месячной заработной плате рабочих завода. Каждая запись содержит поля – фамилия рабочего, наименование цеха, размер заработной платы за месяц. Количество записей – произвольное. Вычислить общую сумму выплат за месяц по цеху X, а также среднемесячный заработок рабочего этого цеха. Напечатать для бухгалтерии ведомость для начисления заработной платы рабочим этого цеха

ТЕМА: ГРАФИКА В ПАСКАЛЕ

49. Вводятся пять вещественных чисел. Записать в первый столбец матрицы целую часть чисел, во второй - дробную часть, приведенную к пятизначному целому, в третий столбец - знак числа: 0 для положительных чисел и 1 - для отрицательных.

50. В файл записать информацию о сотрудниках некоторого предприятия: фамилия, домашний адрес, телефон, образование, оклад. Напечатать список сотрудников, имеющих высшее образование.

**Вопросы для коллоквиумов
по дисциплине: Информатика и программирование**

1. Предмет и задачи информатики, понятие информации.
2. Информационные процессы и технологии.
3. Системы счисления.
4. Кодирование информации в компьютере.
5. Технические средства реализации информационных процессов.
6. Основные функциональные части компьютера.
7. Общие принципы организации и работы компьютера.
8. Классификация ЭВМ. Поколения ЭВМ
9. Определение, назначение, состав и функции операционных систем.
10. Требования к современным операционным системам.
11. Основные определения: вычислительная система, компьютер, конфигурация, аппаратное обеспечение, программы, программное обеспечение, программная конфигурация.
12. Категории программного обеспечения.
13. Текстовый процессор MS Word.
14. Табличный процессор MS Excel.
15. Понятие алгоритма.
16. Свойства алгоритмов. Виды алгоритмов.
17. Алгоритмизация и программирование, языки программирования высокого уровня.
18. Назначение алгоритмического языка PASCAL.
19. Основные символы языка.
20. Простейшие конструкции.
21. Структура программного модуля.
22. Классификация операторов
23. Операторы языка Паскаль.
24. Организация программ линейной структуры.
25. Оператор перехода. Условный оператор.
26. Организация программ разветвляющейся структуры.
27. Оператор выбора.
28. Операторы цикла.
29. Циклы с заданным и неявным числом повторений.
30. Одномерные массивы.
31. Вычисление суммы и произведения.
32. Нахождение наибольшего и наименьшего значений.
33. Вложенные циклы.
34. Двумерные массивы.
35. Оформление подпрограмм и обращение к ним.
36. Подпрограмма-функция. Подпрограмма-процедура.
37. Переменные типы данных.
38. Основные понятия и средства компьютерной графики в Паскале.
39. Компьютерные сети.
40. Основные характеристики.

41. Структура и классификация компьютерных сетей.
42. Локальные вычислительные сети (ЛВС). Структура Internet.
43. Компьютерные вирусы и антивирусные программы.
44. Основы защиты информации, методы защиты информации.

**Тестовые вопросы
по дисциплине: «Информатика и программирование»**

Формируемая компетенция ОПК-7

1. **Периферийные устройства выполняют функцию...**
управления работой ЭВМ по заданной программе
ввода-вывода информации
оперативного сохранения информации
обработки данных, вводимых в ЭВМ
2. **Троянской программой является...**
программа, вредоносное действие которой выражается в удалении и/или модификации системных файлов компьютера
программа, заражающая компьютер независимо от действий пользователя
программа, проникающая на компьютер пользователя через Интернет
вредоносная программа, которая сама не размножается, а выдает себя за что-то полезное, тем самым пытаясь побудить пользователя переписать и установить на свой компьютер программу самостоятельно
3. **Предмет информатики — это:**
язык программирования;
устройство робота;
способы накопления, хранения, обработки, передачи информации;
информированность общества.
4. **Архитектура компьютера — это:**
Техническое описание деталей устройств компьютера;
описание устройств для ввода-вывода информации;
описание программного обеспечения для работы компьютера;
описание устройства и принципов работы компьютера, достаточное для понимания пользователя.
5. **Укажите, в каком файле может храниться рисунок**
TEST.EXE;
ZADAN.TXT;
COMMAND.COM;
CREML.BMP.
6. **Файлом называется** _____
7. **Алгоритм — это** понятное и точное предписание _____ совершить последовательность действий, направленных на _____ поставленной _____ или цели.
8. **Свойство алгоритма — дискретность — обозначает:**
что команды должны следовать последовательно друг за другом;
что каждая команда должна быть описана в расчете на конкретного исполнителя;
разбиение алгоритма на конечное число простых шагов;
строгое движение как вверх, так и вниз.
9. **Какой тип алгоритма должен быть выбран при решении квадратного уравнения:**
линейный;
циклический;
разветвляющийся;
циклически-разветвляющийся.
10. **Разветвляющийся алгоритм — это:**
присутствие в алгоритме хотя бы одного условия;

набор команд, которые выполняются последовательно друг за другом;
многократное исполнение одних и тех же действий;
другое.

11. Наиболее эффективным средством контроля данных в сети являются...

системы архивации
антивирусные программы
RAID-диски

пароли, идентификационные карты и ключи.

12. В состав интегрированного пакета Microsoft Office входят:

система управления базами данных
векторный графический редактор
растровый графический редактор.

13. Наиболее известными способами представления графической информации:

векторной и растровый
физический и логический
точечный и пиксельный
параметрический и структурный

14. Одним из направлений развития информатики является...

компьютерная графика
теория графов
начертательная геометрия
инженерная графика

15. Информацию, изложенную на доступном для получателя языке называют:

16. Устройство вывода предназначено для...

Обучения, игры, расчетов и накопления информации
Программного управления работой вычислительной машины
Передачи информации от машины человеку

17. Тестовый редактор это прикладная _____ предназначенная для _____ и _____ текстового документа

18. Расширение файла это:

Увеличение объема файла на некоторое количество байт
Часть имени файла, которая является идентификатором типа информации содержащейся в файле
Процесс наполнения файла информацией в редакторе

19. Программы, обеспечивающие взаимодействие пользователя, компьютера и других программ называются:

Прикладные программы
Операционные системы
Системы разработки

20. Редактирование электронных таблиц осуществляется в программе:

MS WORD
MS EXCEL
WORD PAD

21. var - это

раздел описания меток
раздел описания типов
раздел описания переменных

22. Топология сети это:

Вид соединения сетевых компьютеров между собой и другими внешними устройствами

Система идентификации компьютера в сети

Аудит компьютерной сети

23. Алфавит языка программирования – это _____

24. Операционная система выполняет функции

25. Графическим редактором называется программа, предназначенная для ...

создания графического образа текста

редактирования вида и начертания шрифта

работы с графическим изображением

построения диаграмм.

26. Линейным называется алгоритм, в котором

27. Форма организации действий, при которой одно и то же действие выполняется несколько раз до тех пор, пока соблюдается некоторое условие, называется _____

28. $a:=2;$

$b:=8;$

$c:=a+b;$

$s:=a*b;$

результатом выполнения этого алгоритма будет:

$c=10 \ s=16$

$c=1 \ s=16$

$c=16 \ s=10$

$c=10 \ s=10$

29. _____ обеспечивает перевод программ с языка высокого уровня на язык более низкого уровня.

30. Инструкция для компьютера по выполнению задания, написанная на компьютерном языке называется

31. Укажите последовательность действий выполняемых при сохранении готовой программы:

Нажать "Файл"

Нажать "Сохранить"

Выбрать "Сохранить как"

Выбрать место сохранения и имя файла

5. Методические материалы, определяющие процедуры оценивания компетенции

5.1 Критерии оценивания качества устного ответа

Оценка «отлично» выставляется за глубокое знание предусмотренного программой материала, за умение четко, лаконично и логически последовательно отвечать на поставленные вопросы.

Оценка «хорошо» – за твердое знание основного (программного) материала, за грамотные, без существенных неточностей ответы на поставленные вопросы.

Оценка «удовлетворительно» – за общее знание только основного материала, за ответы, содержащие неточности или слабо аргументированные, с нарушением последовательности изложения материала.

Оценка «неудовлетворительно» – за незнание значительной части программного материала, за существенные ошибки в ответах на вопросы, за неумение ориентироваться в материале, за незнание основных понятий дисциплины.

5.2 Критерии оценки к выполнению лабораторных работ

Критерии выполнения отчета на max балл Лабораторная работа выполнена полностью, без погрешностей и замечаний.

Критерии выполнения отчета на min балл Лабораторная работа полностью выполнена.

Критерии оценки принятого отчета (в диапазоне от min до max балла)

- программный код не оптимален;
- использованы глобальные переменные;
- не на все вопросы получены верные ответы при защите работы;

Критерии дополнительных баллов за личностные качества

- работа выполнена верно с первого раза, на занятии по расписанию;
- соблюдение рекомендуемого стиля программирования;
- наличие, отсутствие или неполнота смысловых комментариев в программе.

5.3 Критерии оценивания коллоквиума

Оценка «отлично» выставляется за глубокое знание предусмотренного программой материала, за умение четко, лаконично и логически последовательно отвечать на поставленные вопросы.

Оценка «хорошо» – за твердое знание основного (программного) материала, за грамотные, без существенных неточностей ответы на поставленные вопросы.

Оценка «удовлетворительно» – за общее знание только основного материала, за ответы, содержащие неточности или слабо аргументированные, с нарушением последовательности изложения материала.

Оценка «неудовлетворительно» – за незнание значительной части

5.4 Критерии оценивания тестирования

При тестировании все верные ответы берутся за 100%.

90%-100% отлично

75%-90% хорошо

60%-75% удовлетворительно

менее 60% неудовлетворительно

5.5 Критерии оценивания результатов освоения дисциплины на экзамене

Оценка **«отлично»** выставляется за глубокое знание предусмотренного программой материала, содержащегося в основных и дополнительных рекомендованных литературных источниках, за умение четко, лаконично и логически последовательно отвечать на поставленные вопросы, за умение анализировать изучаемые явления в их взаимосвязи и диалектическом развитии, применять теоретические положения при решении практических задач.

Оценка **«хорошо»** – за твердое знание основного (программного) материала, включая расчеты (при необходимости), за грамотные, без существенных неточностей ответы на поставленные вопросы, за умение применять теоретические положения для решения практических задач.

Оценка **«удовлетворительно»** – за общее знание только основного материала, за ответы, содержащие неточности или слабо аргументированные, с нарушением последовательности изложения материала, за слабое применение теоретических положений при решении практических задач.

Оценка **«неудовлетворительно»** – за незнание значительной части программного материала, за существенные ошибки в ответах на вопросы, за неумение ориентироваться в расчетах, за незнание основных понятий дисциплины.