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

«УТВЕРЖДАЮ»
Проректор по учебной работе _____ Г.Ю. Нагорная
«31» марта 2021 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Алгоритмизация и программирование

Уровень образовательной программы бакалавриат

Направление подготовки 09.03.04 Программная инженерия

Направленность (профиль) общий

Форма обучения очная

Срок освоения ОП 4 года

Институт Прикладной математики и информационных технологий

Кафедра разработчик РПД Прикладная 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…….....7
4.2. Содержание дисциплины ……..….7
4.2.1. Разделы (темы) дисциплины, виды учебной деятельности и формы контроля….7
4.2.2. Лекционный курс …….…….……..….8
4.2.3. Лабораторный практикум …….....9
4.2.4. Практические занятия…..…….....10
4.3. Самостоятельная работа обучающегося……..…….……11
5. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине……..…….…….…12
6. Образовательные технологии……...17
7. Учебно-методическое и информационное обеспечение дисциплины……..18
7.1. Перечень основной и дополнительной учебной литературы…….....….18
7.2. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»…….19
7.3. Информационные технологии, лицензионное программное обеспечение…….20
8. Материально-техническое обеспечение дисциплины……..…..…….....20
8.1. Требования к аудиториям (помещениям, местам) для проведения занятий.……20
8.2. Требования к оборудованию рабочих мест преподавателя и обучающихся…….....21
8.3. Требования к специализированному оборудованию……...21
9. Особенности реализации дисциплины для инвалидов и лиц с ограниченными возможностями здоровья…….....22
Приложение 1. Фонд оценочных средств……..…….……..23
Приложение 2. Аннотация рабочей программы дисциплины……..……..……..99
Рецензия на рабочую программу дисциплины……..……100
Лист переутверждения рабочей программы дисциплины……..…….....101

1. ЦЕЛЬ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Цель освоения дисциплины «Алгоритмизация и программирование» состоит в формировании у обучающихся теоретических знаний в области алгоритмизации и современного программирования, включающего в себя методы проектирования, анализа и создания программных продуктов, основанных на использовании структурной и объектно-ориентированной методологии и практических навыков составления алгоритмов, воплощения их на языке программирования Pascal, тестирования и отладки алгоритмов.

При этом ***задачами*** дисциплины являются:

– приобретение обучающимися знаний основных принципов структурного, модульного, объектно-ориентированного программирования, принципов составления алгоритмов, базовых конструкций изучаемых языков программирования, этапов решения задач на компьютере;

– применение синтаксиса языка программирования Visual Basic для написания программ, реализующих заданный алгоритм;

– овладение обучающимися методами построения блок-схемы алгоритмов и навыками реализации алгоритмов на конкретном языке программирования.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

2.1. Дисциплина «Алгоритмизация и программирование» относится к дисциплинам по выбору 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 и последующие дисциплины, направленные на формирование компетенций

№ п/п	Предшествующие дисциплины	Последующие дисциплины
1	Теоретическая информатика	Объектно-ориентированное программирование Основы программирования Ознакомительная практика

3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Планируемые результаты освоения образовательной программы (ОП) – компетенции обучающихся определяются требованиями стандарта по направлению подготовки 09.04.03 Программная инженерия 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-6	Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов	ОПК-6.1. При разработке программных продуктов анализирует языки программирования и методы разработки алгоритмов и программ
			ОПК-6.2. Применяет основы информатики и программирования для конструирования и тестирования программных продуктов
			ОПК-6.3. При решении профессиональных задач в области информационных систем и технологий использует методы алгоритмизации, языки и технологии программирования

4. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1. ОБЪЕМ ДИСЦИПЛИНЫ И ВИДЫ УЧЕБНОЙ РАБОТЫ

Вид учебной работы		Всего часов	Семестр
			№ 2
			часов
1		2	3
Аудиторная контактная работа (всего)		90	90
В том числе:			
Лекции (Л)		36	36
Практические занятия (ПЗ), Семинары (С)		18	18
Лабораторные работы (ЛР)		36	36
Контактная внеаудиторная работа		2	2
В том числе: индивидуальные и групповые консультации		2	2
Самостоятельная работа обучающегося (СРО) (всего)		61	61
Работа с лекциями		15	15
Работа с книжными источниками		15	15
Работа с электронными источниками		10	10
Подготовка к лабораторным занятиям		8	8
Реферат (Реф)		5	5
Подготовка к промежуточному контролю (ППК))		8	8
Промежуточная аттестация	Экзамен(Э)	Э	Э
	экзамен (Э) **в том числе:**	27	27
	Прием экз., час.	0,5	0,5
	Консультация, час.	2	2
	СРО, час.	24,5	24,5
ИТОГО: Общая трудоемкость	**часов**	180	180
	зач. ед.	5	5

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	ПР	СРО	всего	
1	3	4	5	6	7	8	9
Семестр 2							
1.	Раздел 1. Основы алгоритмизации процессов обработки данных.	8	8	2	16	34	Защита лабораторных и практических работ текущий тестовый контроль, контрольная работа, коллоквиум, подготовка реферата
2.	Раздел 2. Язык программирования Pascal. Управляющие операторы языка высокого уровня. Структуры данных.	24	24	14	40	102	
3.	Раздел 3. Основы тестирования и отладки программ.	4	4	2	5	15	
	Контактная внеаудиторная работа					2	индивидуальные и групповые консультации
	Промежуточная аттестация					27	Экзамен
	ИТОГО часов во 2 семестре:	36	36	18	61	180	

4.2.2. Лекционный курс

№ п/п	Наименование раздела (темы) дисциплины	Наименование темы лекции	Содержание лекции	Всего часов
1	2	3	4	5
Семестр 2				
1	Раздел 1. Основы алгоритмизации процессов обработки данных.	Основные понятия алгоритмизации.	Понятие алгоритма и его свойства. Методы разработки алгоритмов. Основные понятия языка высокого уровня. Эволюция и классификация языков программирования. Программа, порядок ее разработки и исполнения. Языки высокого уровня: алфавит, синтаксис, семантика. Концепция типа данных. Интегрированные среды программирования. Парадигмы и технологии программирования.	**8**
		Принципы разработки алгоритмов.		
		Языки и системы программирования.		
		Парадигмы программирования.		
		Принципы отладки и тестового контроля.		
2	Раздел 2. Язык программирования Pascal. Управляющие операторы языка высокого уровня. Структуры данных.	Характеристика языка.	Базовые конструкции структурного программирования и их реализация в виде управляющих конструкций языка. Линейные программы. Программирование условий: условный оператор, оператор выбора. Программирование циклов. Обработка массивов. Обработка символов и строк. Обработка записей. Обработка файлов.	**24**
		Элементы языка. Простые типы данных		
		Базовые конструкции структурного программирования		
		Работа с массивами и указателями. Структурные типы данных		
		Процедуры и функции		
		Работа с файлами		
		Языки высокого уровня: алфавит, синтаксис, семантика.		
		Концепция типа данных.		
		Обработка записей в Паскале.		
		Работа с файлами в Паскале.		
3	Раздел 3. Основы тестирования и отладки программ.	Принципы тестирования программного обеспечения.	Принципы тестирования программного обеспечения. Виды тестирования. Критерии выбора тестов. Тестирование и отладка программы на языке Pascal.	**4**
		Методы тестирования программного обеспечения.		

		Тестирование и отладка программы на языке Pascal.		
	ИТОГО часов во 2 семестре:			**36**

4.2.3. Лабораторный практикум

№ п/п	Наименование раздела (темы) дисциплины	Наименование лабораторной работы	Содержание лабораторной работы	Всего часов
1	2	3	4	5
Семестр 2				
1	**Раздел 1.** Основы алгоритмизации процессов обработки данных.	**Лабораторная работа №1 -4** Система программирования Pascal. Простейшие конструкции языка Паскаль.	Основные этапы разработки алгоритмов: постановка задачи, построение математической модели, разработка алгоритма решения задачи, проверка правильности и оценка сложности алгоритма.	8
2	**Раздел 2**. Язык программирования Pascal. Управляющие операторы языка высокого уровня. Структуры данных.	**Лабораторная работа №5** Разработка линейных алгоритмов. **Лабораторная работа №6** Разработка программ разветвляющейся структуры. Оператор выбора **Лабораторная работа №**7 Разработка программ с использованием цикла с предусловием. Разработка программ с использованием цикла с постусловием. Разработка программ с использованием цикла с параметром. **Лабораторная работа №8** Разработка программ с использованием одномерных массивов и указателей. **Лабораторная работа №9 -10** Сортировка одномерных массивов. **Лабораторная работа №11 - 13** Разработка программ с использованием двумерных массивов. Двумерные	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. Операторы языка Паскаль. Оператор	24

		массивы. **Лабораторная работа №14 - 16** Разработка программ с использование процедур и функций.	перехода. Условный оператор. Организация программ разветвляющейся структуры. Оператор выбора. Операторы цикла. Одномерные массивы. Вложенные циклы. Двумерные массивы. Разработка программ работы со структурированным и файлами.	
3	**Раздел 3.** Основы тестирования и отладки программ.	**Лабораторная работа №17 - 18** Тестирование программного обеспечения.	Тестирование программного обеспечения.	4
	ИТОГО часов во 2 семестре:			**36**

4.2.4. Практические занятия

№ п/п	Наименование раздела (темы) дисциплины	Наименование практического занятия	Содержание практического занятия	Всего часов
1	2	3	4	5
Семестр 2				
1	**Раздел 1.** Основы алгоритмизации процессов обработки данных.	**Практическая работа №1** Принципы построения алгоритмов: использование базовых структур, метод последовательной детализации, сборочный метод.	Принципы построения алгоритмов: использование базовых структур, метод последовательной детализации, сборочный метод. Разработка алгоритмов сложной структуры.	2
2	**Раздел 2**. Язык программирования Pascal. Управляющие операторы языка высокого уровня. Структуры данных.	**Практическая работа №2** Разработка линейных алгоритмов. Разработка программ разветвляющейся структуры. **Практическая работа №**3 Разработка программ с использованием цикла с предусловием. Разработка программ с использованием цикла с постусловием. Разработка программ с использованием цикла с	Процесс трансляции и выполнения программы.	14

		параметром. **Практическая работа №4 -5** Разработка программ с использованием одномерных массивов и указателей. Сортировка одномерных массивов. **Практическая работа №6** Разработка программ с использованием двумерных массивов. Сортировка двумерных массивов. **Практическая работа №7** Использование стандартных функций и процедур для работы со строками. **Практическая работа №8** Организация программ с использование процедур. Организация программ с использованием функций.		
3	**Раздел 3.** Основы тестирования и отладки программ.	**Практическая работа №9** Методы тестирования программного обеспечения.	Методы тестирования программного обеспечения.	2
	ИТОГО часов во 2 семестре:			**18**

4.3. САМОСТОЯТЕЛЬНАЯ РАБОТА ОБУЧАЮЩЕГОСЯ

№ п/п	Наименование раздела (темы) дисциплины	№ п/п	Виды СРО	Всего часов
1	3	4	5	6
Семестр 2				
1	**Раздел 1.** Основы алгоритмизации процессов обработки данных.	1.1.	Работа с лекциями, книжными, электронными источниками, подготовка к текущему тестовому контролю	16
		1.2.	Подготовка к лабораторным занятиям	
2	**Раздел 2**. Язык программирования Pascal. Управляющие операторы языка высокого уровня. Структуры данных.	2.1.	Самоподготовка: внеаудиторное чтение, работа с электронными источниками	40
		2.2.	Выполнение расчетно-графических работ	
		2.3.	Подготовка рефератов по выбранной теме, подготовка к промежуточному тестовому контролю	
3	**Раздел 3.** Основы тестирования и отладки программ.	3.1	Работа с лекциями, книжными, электронными источниками.	5
		3.2	Подготовка рефератов по выбранной теме, подготовка к промежуточному тестовому контролю	
ИТОГО часов во 2 семестре:				**61**

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

5.1. Методические указания для подготовки обучающихся к лекционным занятиям

Основными формами обучения основам программирования являются лекции, лабораторные занятия и консультации, зачет, а также самостоятельная работа.

Лекции составляют основу теоретического обучения и дают систематизированные основы научных знаний по дисциплине, раскрывают состояние и перспективы развития соответствующей области науки, концентрируют внимание обучающихся на наиболее сложных и узловых вопросах, стимулируют их активную познавательную деятельность и способствуют формированию творческого мышления.

Ведущим методом в лекции выступает устное изложение учебного материала, сопровождающееся использование мультимедиа аппаратуры.

Лекция является исходной формой всего учебного процесса, играет направляющую и организующую роль в самостоятельном изучении предмета. Важнейшая роль лекции заключается в личном воздействии лектора на аудиторию.

Основная дидактическая цель лекции — обеспечение ориентировочной основы для дальнейшего усвоения учебного материала. Построение лекций по дисциплине «Основы программирования» осуществляется на основе принципов научности **(**предполагает воспитание диалектического подхода к изучаемым предметам и явлениям, диалектического мышления, формирование правильных представлений, научных понятий и умения точно выразить их в определениях и терминах, принятых в науке)

Активно используется при чтении дисциплины лекция – презентация.

На лекциях раскрываются основные теоретические аспекты, приводятся примеры реализации на практике.

Специфической чертой изучения данного курса является то, что приобретение умений и навыков работы невозможно без систематической тренировки, которая осуществляется на лабораторных занятиях. Консультации проводятся с целью оказания помощи обучающимся в изучении учебного материала, подготовки их к выполнению лабораторных работ.

Основные требования к лекции

- Глубокое научное содержание.
- Творческий характер.
- Информационная насыщенность.
- Единство содержания и формы.
- Логически стройное и последовательное изложение.
- Яркость изложения.
- Учёт характера и состава аудитории.

Основное внимание в лекции сосредотачивается на глубоком, всестороннем раскрытии главных, узловых, наиболее трудных вопросов темы. Уже на начальном этапе подготовки лекции преподаватель решает вопрос о соотношении материалов учебника и лекции. Он выделяет из учебника ведущие проблемы для более глубокого и всестороннего раскрытия их в лекции.

Важным этапом является определение организационной структуры лекции, распределение времени на каждый вопрос, вводную часть и заключение.

В ходе подготовки лекции необходимо:

- Определить основное содержание и расположение материала.
- Продумать: где, как, в какой мере использовать методологические положения ведущих учёных; как использовать документы и другие материалы; в какой мере и как осуществить связь с задачами образования; где и в какой степени расположить материал

воспитательного характера; какие предложить методические советы по самостоятельной работе обучающихся;

- Как лучше использовать мультимедиа, наглядные пособия, поясняющие какие-то основные, принципиальные положения лекции.

В круг задач лектора входят:

1. Установление и поддержание контакта с аудиторией
2. Создание у слушателей интереса к предмету лекции
3. Достижение убедительности речи
4. Эмоциональное воздействие на слушателей
5. Применение наглядных пособий (мультимедиа, фантомов, приборов и т.п.)
6. Соблюдение некоторых правил на трибуне

Начало лекции Лектор должен сообщить чётко, ясно, не торопясь, название темы лекции, дать возможность слушателям записать его.

Затем изложить вводную часть, в которой сказать:

- О роли и месте данной темы в курсе;
- Дать краткую характеристику литературы;
- Сообщить о распределении времени на тему;
- Если не первая лекция по теме, то провести связь с предшествующей лекцией.

Далее сообщить план лекции, также дав возможность студентам записать вопросы. Перед изложением каждого вопроса его надо называть. Завершается рассмотрение вопроса небольшим выводом. Большую помощь в обобщении и фиксировании материала оказывает сопровождение объяснения демонстрацией материала с помощью мультимедиа аппаратуры.

Начало лекции имеет большое значение для установления контакта с аудиторией, для возбуждения у слушателей интереса к теме. В этих целях можно использовать яркий пример или остро поставленный вопрос, подчеркнуть теоретическое и практическое значение данной темы в тематическом плане курса и в практической деятельности.

Поддержание внимания слушателей на протяжении всей лекции достигается:

- Логикой изложения материала.
- Глубиной содержания материала.
- Чётким формулированием положений.
- Использованием в лекции новых интересных данных.
- Использование мультимедиа.
- Включение в лекцию материалов из практической деятельности.

Одним из сложных вопросов методики чтения лекции является обращение с текстом. Привязанность к тексту вследствие плохой подготовки, недостаточного владения материалом приводит к ослаблению связи с аудиторией. В то же время не следует, не владея соответствующими навыками, пытаться проводить лекцию без текста, по памяти. При этом допускаются ошибки, повторения, ослабление логической нити рассуждения, пропуски отдельных важных положений темы и т.п.

Заключительная часть лекции

В ней обобщаются наиболее важные, существенные вопросы лекции; делаются выводы, ставятся задачи для самостоятельной работы.

Существует твёрдый порядок, требующий, чтобы в конце лекции преподаватель оставил несколько минут для ответов на вопросы.

5.2. Методические указания для подготовки обучающихся к лабораторным занятиям

Целью лабораторных работ является формирование умений у обучающихся созданию программ, реализующих заданный алгоритм. В результате выполнения лабораторных работ обучающиеся приобретают практические навыки реализации алгоритмов на конкретном языке программирования.

Лабораторные занятия проводятся в компьютерном классе кафедры Информатики и информационных технологий, где на ЭВМ установлена среда программирования Visual Studio 2010.

Таким образом, на лабораторных занятиях обучающиеся выполнят 9 лабораторных работ.

В конце каждого лабораторного занятия обучающийся будет предъявлять преподавателю электронный вариант/распечатку разработанной им программы.

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, студентам 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 студентам 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 виды представляемых образов 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).

7. Проверить визуальное восприятие презентации.

К видам представляемых образов относятся иллюстрации, образы, диаграммы, таблицы. Иллюстрация - представление реально существующего зрительного ряда. Образы – в отличие от иллюстраций - метафора. Их назначение - вызвать эмоцию и создать отношение к ней, воздействовать на аудиторию. С помощью хорошо продуманных и представляемых образов, информация может надолго остаться в памяти человека. Их используют для убедительной демонстрации данных, для пространственного мышления в дополнение к логическому. Таблица - конкретный, наглядный и точный показ данных. Ее основное назначение - структурировать информацию, что порой облегчает восприятие данных аудиторией.

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-визуальных и визуальных материалов.

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

Промежуточная аттестация

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

5.5. Методические указания для обучающихся по написанию реферата

Подготовка рефератов направлена на развитие и закрепление у обучающихся навыков самостоятельного глубокого, творческого и всестороннего анализа научной, методической и другой литературы по актуальным проблемам дисциплины; на выработку навыков и умений грамотно и убедительно излагать материал, четко формулировать теоретические обобщения, выводы и практические рекомендации.

Рефераты должны отвечать высоким квалификационным требованиям в отношении научности содержания и оформления.

Темы рефератов, как правило, посвящены рассмотрению одной проблемы. Объем реферата может быть от 12 до 15 страниц машинописного текста, отпечатанного через 1,5 интервала, а на компьютере через 1 интервал (список литературы и приложения в объем не входят).

Текстовая часть работы состоит из введения, основной части и заключения.

Во введении обучающийся кратко обосновывает актуальность избранной темы реферата, раскрывает конкретные цели и задачи, которые он собирается решить в ходе своего небольшого исследования.

В основной части подробно раскрывается содержание вопроса (вопросов) темы.

В заключении кратко должны быть сформулированы полученные результаты исследования и даны выводы. Кроме того, заключение может включать предложения автора, в том числе и по дальнейшему изучению заинтересовавшей его проблемы.

В список литературы (источников и литературы) обучающийся включает только те документы, которые он использовал при написании реферата.

В приложении (приложения) к реферату могут выноситься таблицы, графики, схемы и другие вспомогательные материалы, на которые имеются ссылки в тексте реферата.

Самостоятельная работа является обязательной для каждого обучающегося, ее объем по дисциплине «Основы программирования» определяется учебным планом.

При самостоятельной работе обучающийся взаимодействует с рекомендованными материалами при минимальном участии преподавателя.

6. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

№ п/п	№ семестра	Виды учебной работы	Образовательные технологии	Всего часов
Семестр 2				
1	2	*Лекция* «Программирование – основные понятия»	Лекция – беседа	2
2	2	Лабораторное занятие «Алгоритмизация. Задачи на ветвление»	Мозговой штурм	2
3	2	Лабораторное занятие «Массивы. Поиск, сортировка, генерация значений элементов одномерных массивов»	Мозговой штурм	6
ИТОГО часов во 2 семестре:				10

7. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

7.1. Перечень основной и дополнительной учебной литературы

Основная литература

1. 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
2. Лубашева, Т. В. Основы алгоритмизации и программирования : учебное пособие / Т. В. Лубашева, Б. А. Железко. — Минск : Республиканский институт профессионального образования (РИПО), 2016. — 379 с. — ISBN 978-985-503-625-9. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/67689.html
3. Белева, Л. Ф. Программирование на языке С++ : учебное пособие / Л. Ф. Белева. — Саратов : Ай Пи Эр Медиа, 2018. — 81 с. — ISBN 978-5-4486-0253-5. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/72466.html
4. Тюльпинова, Н. В. Алгоритмизация и программирование : учебное пособие / Н. В. Тюльпинова. — Саратов : Вузовское образование, 2019. — 200 с. — ISBN 978-5-4487-0470-3. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/80539.html

Дополнительная литература

1. Потопахин, В. В. Современное программирование с нуля / В. В. Потопахин. — Саратов : Профобразование, 2017. — 240 с. — ISBN 978-5-4488-0006-1. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/62902.html

2. Основы алгоритмизации и программирования : лабораторный практикум / составители Е. И. Николаев. — Ставрополь : Северо-Кавказский федеральный университет, 2015. — 211 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/63112.html

3. Стенли, Липпман Язык программирования С++ : полное руководство / Липпман Стенли, Лажойе Жози ; перевод А. Слинкин. — Саратов : Профобразование, 2017. — 1104 с. — ISBN 978-5-4488-0136-5. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: http://www.iprbookshop.ru/63964.html

7.2. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

http://window.edu.ru- **Единое окно доступа к образовательным ресурсам;**
http:// fcior.edu.ru - **Федеральный центр информационно-образовательных ресурсов;**
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
ЭБС IPR SMART	Лицензионный договор № 9368/22П от 01.07.2022 г 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

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

Учебная аудитория для проведения занятий лекционного типа	Специализированная мебель: Кафедра - 1шт., доска меловая - 1шт., парты - 30шт., стулья - 61шт., Технические средства обучения, служащие для предоставления учебной информации большой аудитории: Проектор - 1шт. Экран моторизованный - 1шт. Ноутбук -1шт.
Лаборатория современных экономических информационных систем	Специализированная мебель: Парты - 6шт., доска меловая - 1шт., компьютерные столы - 7шт., стол преподавательский - 3шт., стулья - 28 шт., сол лабораторный -3 шт. Лабораторное оборудование, технические средства обучения, служащие для предоставления учебной информации большой аудитории: Компьютер в сборе - 7 шт.
Учебная аудитория для проведения занятий семинарского типа, курсового проектирования (выполнение курсовых работ), групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации	Специализированная мебель: Доска меловая - 1шт., стол преподавательский - 1шт., парты - 8шт., стулья - 26шт., компьютерные столы - 10шт., стул мягкий – 1шт. Лабораторное оборудование, технические средства обучения, служащие для предоставления учебной информации большой аудитории: ПК-10 шт.
Помещение для самостоятельной работы.	**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. **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.

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.

Приложение 1.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине Алгоритмизация и программирование

1. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ ПО ДИСЦИПЛИНЕ

Алгоритмизация и программирование

5. **Компетенции, формируемые в процессе изучения дисциплины**

Индекс	Формулировка компетенции
ОПК-6	Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов

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 обучающимися.

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

Разделы (темы) дисциплины	**Формируемые компетенции (коды)**
	ОПК-6
Раздел 1. Основы алгоритмизации процессов обработки данных.	+
Раздел 2. Язык программирования Pascal. Управляющие операторы языка высокого уровня. Структуры данных.	+
Раздел 3. Основы тестирования и отладки программ.	+
Раздел 1. Основы алгоритмизации процессов обработки данных.	+

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

ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов						
Индикаторы достижения компетенции	**Критерии оценивания результатов обучения**				**Средства оценивания результатов обучения**	
	неудовлетв	**удовлетв**	**хорошо**	**отлично**	Текущий контроль	Промежуточная аттестация
ОПК-6.1. При разработке программных продуктов анализирует языки программирования и методы разработки алгоритмов и программ	Не знает основные теоретические положения при разработке программных продуктов анализирует языки программирования и методы разработки алгоритмов и программ	Демонстрирует частичные знания при разработке программных продуктов анализирует языки программирования и методы разработки алгоритмов и программ	Демонстрирует знания при разработке программных продуктов анализирует языки программирования и методы разработки алгоритмов и программ	Раскрывает полное содержание при разработке программных продуктов анализирует языки программирования и методы разработки алгоритмов и программ	Защита лабораторных и практических работ текущий тестовый контроль, контрольная работа коллоквиум, подготовка реферата	экзамен
ОПК-6.2. Применяет основы информатики и программирования для конструирования и тестирования программных продуктов	Фрагментарное применение основ информатики и программирования для конструирования и тестирования программных продуктов	Демонстрирует минимально необходимые навыки основ информатики и программирования для конструирования и тестирования программных продуктов	Демонстрирует достаточно развитые навыки основ информатики и программирования для конструирования и тестирования программных продуктов	Демонстрирует высокоразвитые профессиональные навыки основ информатики и программирования для конструирования и тестирования программных продуктов		экзамен
ОПК-6.3. При решении профессиональных задач в области информационных систем и технологий использует методы алгоритмизации, языки и технологии программирования	Фрагментарное применение навыков работы при профессиональных задач в области информационных систем и технологий использует методы алгоритмизации, языки и технологии программирования	Демонстрирует минимально необходимые навыки применения знаний при решении профессиональных задач в области информационных систем и технологий использует методы алгоритмизации, языки и технологии программирования	Демонстрирует в целом успешные, но содержащее отдельные пробелы навыки применения знаний при решении профессиональных задач в области информационных систем и технологий использует методы алгоритмизации, языки и технологии программирования	Демонстрирует высокоразвитые профессиональные навыки применения знаний при решении профессиональных задач в области информационных систем и технологий использует методы алгоритмизации, языки и технологии программирования		экзамен

1. Комплект контрольно-оценочных средств по дисциплине Алгоритмизация и программирование

Практические работы
по дисциплине Алгоритмизация и программирования

ПРАКТИЧЕСКАЯ РАБОТА 1

ТЕМА: Принципы построения алгоритмов

Цель работы: научиться составлять алгоритмы графическим способом (блок-схем)

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Этапы решения задачи на ЭВМ. Работа по решению любой задачи с использованием компьютера включает в себя шесть этапов

1) постановка задачи
2) формализация задачи
3) построение алгоритма
4) составление программы на языке программирования
5) отладка и тестирование программы
6) проведение расчетов и анализ полученных результатов

Часто эту последовательность называют технологической цепочкой решения задачи на ЭВМ.

На этапе постановки задачи следует четко определить, что дано и что требуется найти. Важно описать полный набор исходных данных, необходимых для решения задачи На этапе формализации чаще всего задача переводится на язык математических формул, уравнений и отношений. Если решение задачи требует математического описания какого-то реального объекта, явления или процесса, то ее формализация равносильна получению соответствующей математической модели/

Третий этап — это построение алгоритма. Опытные программисты часто сразу пишут программы на определенном языке, не прибегая к каким-либо специальным средствам описания алгоритмов (блок-схемам, псевдокодам), однако в учебных целях полезно сначала использовать эти средства, а затем переводить полученный алгоритм на язык программирования.

Алгоритм — это последовательность команд управления каким-либо исполнителем. В школьном курсе информатики с понятием алгоритма и методами построения алгоритмов ученики знакомятся на примерах учебных исполнителей: Робота, Черепахи, Чертежника и др. Эти исполнители ничего не вычисляют. Они создают рисунки на экране, перемещаются в лабиринтах, перетаскивают предметы с места на место.

Данные и величины. Совокупность величин, с которыми работает компьютер, принято называть данными. По отношению к программе различают исходные, окончательные (результаты) и промежуточные данные, которые получают в процессе вычислений.

В каждом языке программирования существует своя концепция и своя система типов данных. Однако в любой язык входит минимально необходимый набор основных типов данных: целые вещественные, логические и символьные. С типом величины связаны три ее свойства: множество допустимых значений, множество допустимых операций, форма внутреннего представления.

Блок-схема — графическое представление алгоритма. Она состоит из функциональных блоков, которые выполняют различные назначения (ввод/вывод, начало/конец, вызов функции и т.д.).

Существует несколько основных видов блоков, которые нетрудно запомнить:

Пример №1: Рассчитать площадь и периметр прямоугольника по двум известным сторонам. Данная

задача не должна представлять особой трудности, так как построена она на хорошо известных всем нам формулах расчета площади и периметра прямоугольника, поэтому зацикливаться на выведении этих формул мы не будем.

Составим алгоритм решения подобных задач:

1) Прочитать задачу.

2) Выписать известные и неизвестные нам переменные в «дано». (В задаче №1 к известным переменным относятся стороны: a,b ;к неизвестным — площадь S и периметр P)

3) Вспомнить либо составить необходимые формулы. (У нас: S=a*b; P=2*(a+b))

4) Составить блок-схему.

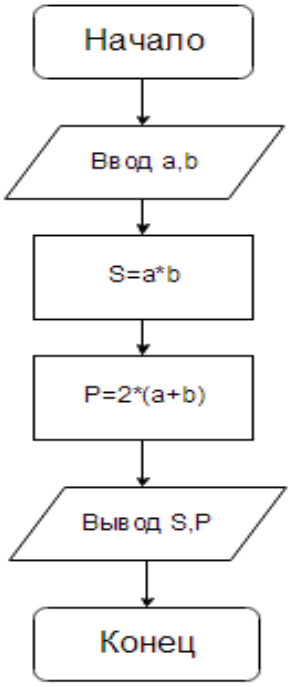
5) Записать решение на языке программирования Pascal.

Запишем условие в более кратком виде.

Дано: a,b

Найти: S,P

Блок-схема:



Словесное описание алгоритма:

Структура программы, решающей данную задачу, тоже проста:

1) Описание переменных;

2) Ввод значений сторон прямоугольника;

3) Расчет площади прямоугольника;

4) Расчет периметра прямоугольника;

5) Вывод значений площади и периметра;

6) Конец.

ЗАДАНИЕ

Составить словесно-формульный алгоритм и блок-схему для следующих задач:

1. Вычислить периметр и площадь прямоугольного треугольника по длинам двух катетов: a и b;
2. Вычислить длину окружности и площадь круга с заданным радиусом R
3. Вычислить расстояние между двумя точками с заданными координатами (x1, y1) и (x2, y2)

ПРАКТИЧЕСКАЯ РАБОТА 2

ТЕМА 1: Разработка линейных алгоритмов.

1). Цель работы: научиться составлять программы линейной структуры, реализовывать в программе оператор присваивания, процедуры ввода/вывода; строить блок-схемы линейной конструкции.

Оборудование: ПК, ИСР Pascal ABC

ТЕОРЕТИЧЕСКИЕ ДАННЫЕ

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

- программы с линейной структурой являются простейшими и используются, как правило, для реализации обычных вычислений по формулам;

- в программах с линейной структурой инструкции выполняются последовательно, одна за другой;

- алгоритм программы с линейной структурой может быть представлен следующим образом (Рис. 1):

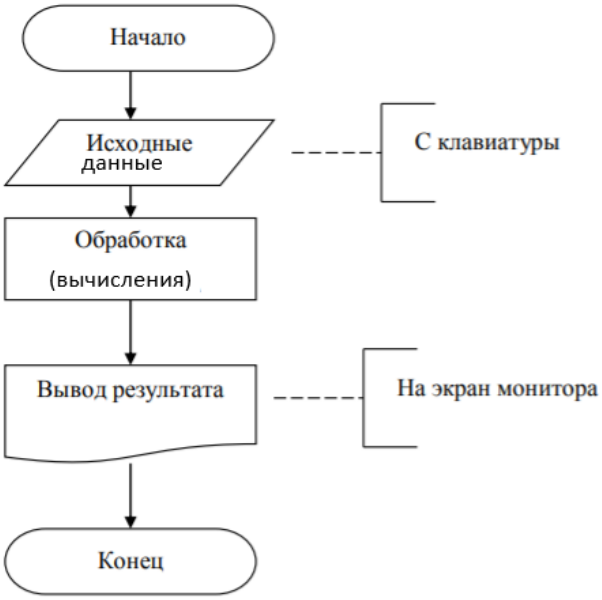


Рис. 1. Блок-схема линейной конструкции Математические стандартные функции Турбо Паскаля

Обращение	Тип аргумента	Тип результата	Функция
Pi	—	*R*	Число π = 3.1415926536E+00
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

Где I — Integer; R — Real

ЗАДАНИЕ

1. Написать программу вычисления объема цилиндра. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление объема цилиндра Введите исходные данные:
Радиус основания (см) —> 5
Высота цилиндра (см) —> 10
Объем цилиндра 1570.80 куб. см.
Для завершения работы программы нажмите <Enter>.

2. Написать программу вычисления объема цилиндра. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление объема цилиндра
Введите исходные данные:
Радиус основания (см) —> 5
Высота цилиндра (см) —> 10
Объем цилиндра 1570.80 куб. см.
Для завершения работы программы нажмите <Enter>.

3. Написать программу вычисления двух выражений:

$$z_1 = \frac{\sqrt{2b + 2\sqrt{b^2 - 4}}}{\sqrt{b^2 - 4} + b + 2}$$

$$z_2 = \sqrt{\frac{a+b}{a-3}}$$

4. Для каждой программы составить блок-схему

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие типы данные вы знаете?
2. Что такое алгоритм?
3. Приведите пример алгоритма из реальной жизни
4. Какими свойствами обладает алгоритм?
5. Что такое линейная конструкция?
6. Какие операторы используются для реализации линейной конструкции в программе?
7. Назовите процедуры ввода/вывода данных
8. Что такое формат вывода данных?
9. Перечислите основные разделы программы

ТЕМА 1: РАЗРАБОТКА ПРОГРАММ РАЗВЕТВЛЯЮЩЕЙСЯ СТРУКТУРЫ

2) Цель работы: знать о разветвляющейся конструкции алгоритма; уметь реализовывать алгоритмическую конструкцию ветвление с помощью условного оператора и оператора выбора в программе, написанной на языке Паскаль.

Оборудование: ПК, ИСР Pascal ABC

ТЕОРЕТИЧЕСКИЕ ДАННЫЕ

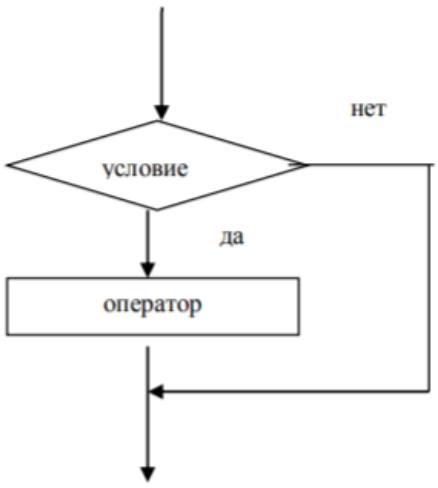
Ветвление — алгоритм, в котором предусмотрены разветвления, указанные в последовательности действий на два направления в зависимости от итогов проверки заданного условия. То есть такой алгоритм, обязательно содержит условие и в зависимости от результата выполнения условия происходит выбор действия.

Например: Если день рабочий, то идем в лицей, иначе будем отдыхать. Таких примеров можем привести много из обычной жизни и наук. К примеру, физика: Если удар упругий, то масса тела сохраняется, иначе масса изменяется.

Алгоритмическая конструкция ветвление программируется с помощью условного оператора If, который может быть представлен двумя вариантами (Таблица 1).

Условный оператор If

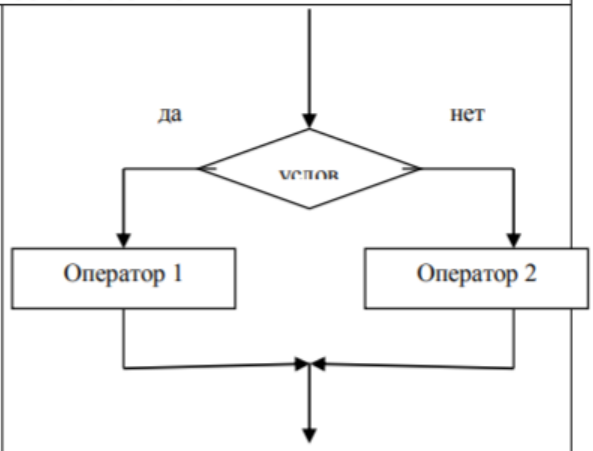
Таблица 1

Конструкция	**Графическое представление блок - схема)**
1 Вариант — неполное ветвление	
If <условие> **Then** <оператор> Неполное ветвление — в зависимости от результата проверки условия либо выполняются действия одной ветви «да» (оператор), либо эти действия не выполняются.	

2 Вариант — полное ветвление

If <условие> **Then** <оператор 1> **Else** <оператор 2>

Полное ветвление — в зависимости от результата проверки условия выполняются только оператор 1 ветви «да» или только оператор 2 ветви «нет»



Условие — это логическое выражение, которое может принимать одно из двух значений: true (истина — условие выполняется) и false (ложь — условие не выполняется).

В условии используются операции отношения (=, <> , >,<, >=, <=) и логические операции (and (И), or (ИЛИ), xor (исключающее ИЛИ), not (отрицание)). Если требуется проверить несколько условий, их объединяют с помощью логических операций.

Примеры логических выражений:

A<2

(x<>0) and (y<>0)

Если между служебными словами стоят несколько операторов, то они заключаются в операторные скобки Begin…End

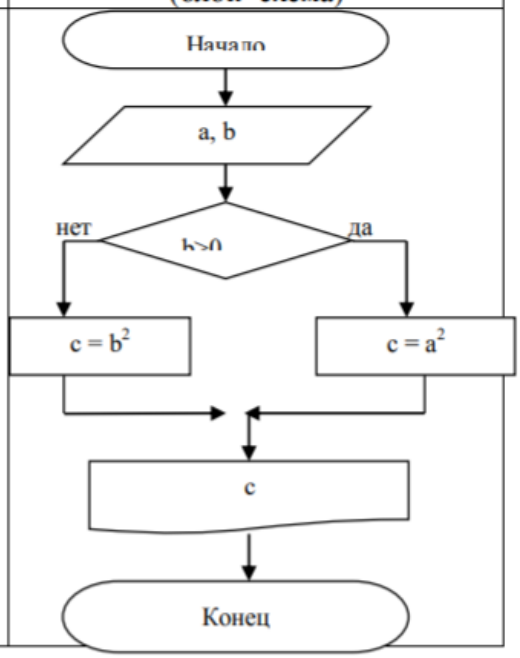
Рассмотрим пример:

Даны 2 вещественных числа. Если числа положительные, то возвести в квадрат первое число, иначе возвести в квадрат второе число.

Листинг программы

```
Program Primer_1;
Uses crt;
{Обозначим 1- ое число через переменную a, 2-ое
через переменную b, результат — c}
Var a, b, c: real;
Begin
Clrscr;
Writeln('Введите  первое число ');
Readln(a);
Writeln('Введите  второе число ');
Readln(b);
If (a>0) and (b>0) Then c:=sqr(a) Else c:=sqr(b);
Writeln('Результат = ', c:5:2);
End.
```

Графическое представление (блок -схема)



ЗАДАНИЕ

Правила пунктуации при записи операторов

1. Точка с запятой не ставится в разделах описаний после зарезервированных слов uses, label, type, const, var и ставится после завершения каждого описания

2. Точка с запятой не ставится после begin и перед end, так как эти слова являются операторными скобками, а не операторами;

3. Точка с запятой является разграничителем операторов, ее отсутствие между операторами вызывает ошибку компиляции;

4. В операторах цикла точка с запятой не ставится после служебных слов while, repeat, do, until;

5. В условных оператора точка с запятой не ставится после then и перед else

ЗАДАНИЕ

Выполните задание по варианту, назначенному преподавателем.

Вариант 1

Задание 1

Даны три действительные числа. Возвести в квадрат те из них, значения которых положительны, и в четвертую степень — отрицательные.

Задание 2

Написать программу, которая вычисляет частное от деления двух чисел. Программа должна проверять правильность введенных пользователем данных и, если они неверные (делитель равен нулю), выдавать сообщение об ошибке. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление частного.

Введите в одной строке делимое и делитель,затем нажмите <Enter>

-> 12 О

Вы ошиблись. Делитель не должен быть равен нулю.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 2

Задание 1

Даны два действительные числа. Если числа положительны найти их сумму, если отрицательны — произведение

Задание 2

Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 10% предоставляется, если сумма покупки больше 1000 руб. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление стоимости покупки с учетом скидки.

Введите сумму покупки и нажмите <Enter>

-> 1200

Вам предоставляется скидка 10%

Сумма покупки с учетом скидки: 1080.00 руб.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 3

Задание 1

Даны действительные числа x и y, не равные друг другу. Меньшее из этих чисел заменить половиной их суммы, а большее — их удвоенным произведением

Задание 2

Написать программу проверки знания даты основания Санкт-Петербурга. В случае неверного ответа пользователя программа должна выводить правильный ответ. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

В каком году был основан Санкт-Петербург?

Введите число и нажмите <Enter>

-> 1705

Вы ошиблись, Санкт-Петербург был основан в 1703 году.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 4

Задание 1

Данные два вещественных числа. Если первое число больше второго, то возвести его в третью степень, если равно второму — прибавить к нему второе число

Задание 2

Написать программу определения стоимости разговора по телефону с учетом скидки 20%, предоставляемой по воскресеньям. Ниже представлен рекомендуемый вид экрана программы во время ее работы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление стоимости разговора по телефону.

Введите исходные данные:

Длительность разговора (целое количество минут) —> 3

День недели (1 - понедельник, ... 7 — воскресенье) —> 6

Предоставляется скидка 20%.

Стоимость разговора: 5.52 руб.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 5

Задание 1

Даны три действительные числа. Если первое число больше второго, умножить данное число на 5, если первое число больше третьего — разделить на два

Задание2

Написать программу — модель анализа пожарного датчика в помещении, которая выводит сообщение «Пожароопасная ситуация», если температура в комнате превысила 60^0C

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 6

Задание 1

Даны действительные числа a, b, c. Удвоить эти числа, если $a \geq b \geq c$, иначе оставить без изменения.

Задание 2

Написать программу, которая анализирует данные о возрасте и относит человека к одной из четырех групп: дошкольник, ученик, работник, пенсионер. Возраст вводится с клавиатуры.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 7

Задание 1

Д аны три числа a, b, c. Определить какое из них равно d. Если ни одно не равно d, то найти сумму чисел a, b, c.

Задание 2

Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 3% предоставляется в том случае, если сумма покупки больше 500 руб., в 5% — если сумма больше 1000 руб. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление стоимости покупки с учетом скидки.
Введите сумму покупки и нажмите <Enter>
-> 640
Вам предоставляется скидка 3%
Сумма покупки с учетом скидки: 620.80 руб.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 8

Задание 1

Даны три действительные числа. Найти минимальное и максимальное число.

Задание 2

Написать программу проверки знания истории архитектуры. Программа должна вывести вопрос и три варианта ответа. Пользователь должен выбрать правильный ответ и ввести его номер. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Архитектор Исаакиевского собора:
1. Доменико Трезини
2. Огюст Монферран
3. Карл Росси
Введите номер правильного ответа и нажмите <Enter>
-> 3
Вы ошиблись.
Архитектор Исаакиевского собора — Огюст Монферран.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 9

Задание 1

Даны три действительные числа. Если все числа положительны, найти среднее арифметическое, иначе произведение.

Задание 2

Написать программу проверки знания истории архитектуры. Программа должна вывести вопрос и три варианта ответа, а пользователь — выбрать правильный ответ и ввести его номер. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены

полужирным шрифтом).

Невский проспект получил свое название:

1. По имени реки, на берегах которой расположен Санкт-Петербург
2. По имени близко расположенного монастыря Александро-Невской лавры
3. В память о знаменитом полководце Александре Невском

Введите номер правильного ответа и нажмите <Enter>

-> 3

Вы ошиблись.

Правильный ответ: 2.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 10

Задание 1

Даны два вещественных числа, если числа не равны нулю, возвести из в третью степень, иначе во вторую степень.

Задание 2

Написать программу, которая сравнивает два числа, введенных с клавиатуры. Программа должна указать, какое число больше, или, если числа равны, вывести соответствующее сообщение. Ниже представлен рекомендуемый вид экрана во время работы программы.

Введите в одной строке два целых числа

-> 34 67

34 меньше 67.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

ПРАКТИЧЕСКАЯ РАБОТА 4

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

Цель работы: научиться использовать операторы циклов при составлении программ на языке Паскаль; составлять блок-схему циклической структуры

Оборудование: ПК, ИСР Pascal ABC

ТЕОРЕТИЧЕСКИЕ ДАННЫЕ

Операторы цикла используются для многократного повторения аналогичных вычислений.

Для организации цикла в Паскале имеются три различных оператора.

1. Регулярный оператор For

For <параметр цикла>:=<начальное значение> **to** <конечное значение> **do** S;

S- простой или составной оператор.

– инструкция for используется для организации циклов с фиксированным, определяемым во время разработки программы, числом повторений;

– количество повторений цикла определяется начальным и конечным значениями переменной-счетчика;

– переменная-счетчик должна быть целого типа (integer).

При каждом прохождении цикла <параметр цикла>, начиная с <начального значения>, увеличивается на единицу. Цикл выполняется, пока <параметр цикла> не станет больше <конечного значения>.

Другой вариант записи оператора For:

For <параметр цикла >:=< начальное значение> **downto** <конечное значение> **do** S;

В этом случае при каждом прохождении цикла <параметр цикла> уменьшается на единицу от <начального значения> до <конечного значения>.

2. Оператор цикла While с проверкой предусловия:

While <условие> do S; {Пока выполняется условие, делать}

Цикл выполняется, пока условие истинно (true).

3. Оператор цикла Repeat с проверкой постусловия:

Repeat S until <условие>; {Выполнять до тех пор, пока не будет выполнено условие}

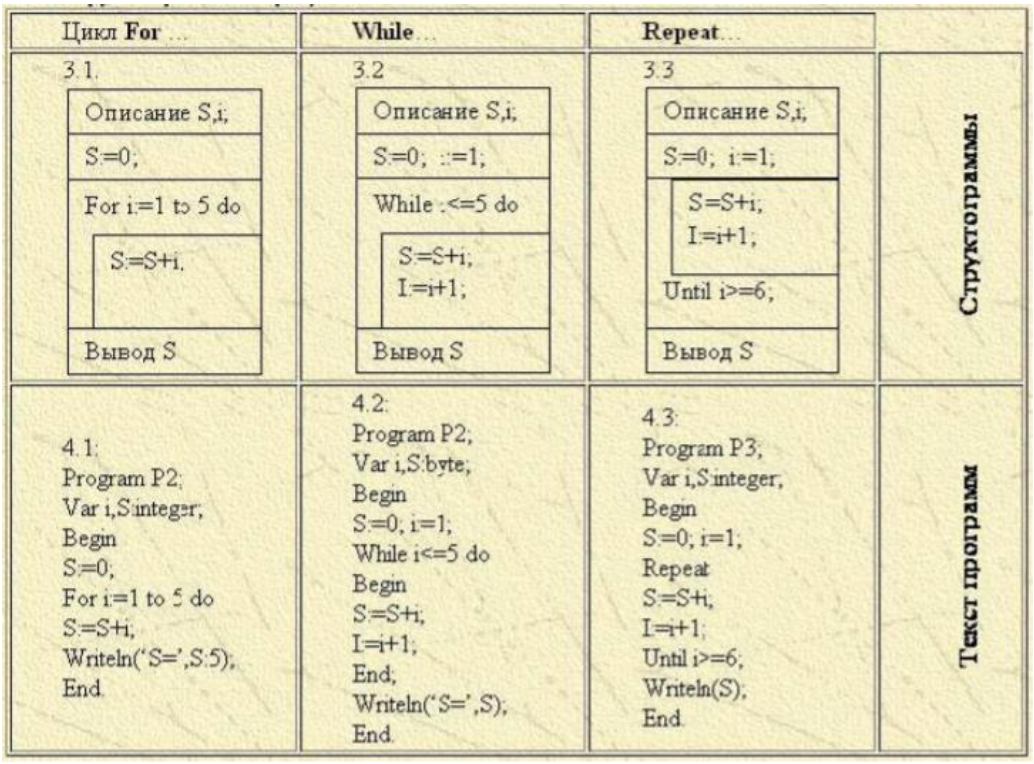
Цикл выполняется, пока условие ложно (false).

Пример

Постановка задачи. Найти сумму 5 целых чисел от 1 до 5. Написать программы для определения суммы с помощью трех рассмотренных операторов цикла.

Структограмма и программа приведены в таблице 1

Таблица 1. Операторы цикла



ЗАДАНИЕ

Задание 1. Составьте программы, используя регулярный оператор цикла согласно своему варианту. Номер варианта соответствует номеру вашего рабочего ПК.

Вариант 1

1. Написать программу, которая выводит таблицу квадратов первых десяти чисел. Ниже представлен рекомендуемый вид экрана во время работы программы.

Таблица квадратов

Число Квадрат
1 1
2 4
3 9
4 16
5 25
6 36
7 49
8 64
9 81
10 100

2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы.

Вариант 2

1. Написать программу, которая выводит таблицу квадратов целых положительных чисел, введенных пользователем с клавиатуры. Ниже представлен рекомендуемый вид экрана во время работы программы.

Таблица квадратов нечетных чисел.

Число Квадрат
1 1
3 9
5 25
7 49
9 81

2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы.

Вариант 3

1. Написать программу, которая вводит с клавиатуры 5 дробных чисел и вычисляет их среднее арифметическое. Рекомендуемый вид экрана во время работы программы приведен ниже:

Вычисление среднего арифметического последовательности дробных чисел.
После ввода каждого числа нажимайте <Enter>
-> 5.4
-> 7.8

-> 3.0
-> 1.5
-> 2.3
Среднее арифметическое введенной последовательности: 4.00
2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы

Вариант 4

1. Написать программу, которая выводит на экран таблицу стоимости, например, яблок в диапазоне от 100 г до 1 кг с шагом 100. Ниже представлен рекомендуемый вид экрана программы во время ее работы (данные, введенные пользователем, выделены полужирным шрифтом).

Введите цену одного килограмма и нажмите <Enter> (копейки от рублей отделяйте точкой) -> *16.50*

Вес (гр) Стоимость (руб.)
100 1.65
200 3.30
300 4.95
400 6.60
500 8.25
600 9.90
700 11.55
800 13.20
900 14.85
1000 16.50

2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы

Вариант 5

1. Написать программу, которая выводит на экран таблицу перевода из градусов Цельсия (C) в градусы по Фаренгейту (F) для значений от 15 до 30 с шагом 1 градус. Перевод осуществляется по формуле F =C*1.8+32

2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы

Вариант 6

1. Написать программу, которая выводит таблицу значений функции $y=-2,4x^2+5x-3$ в диапазоне от – 2 до 2 с шагом 0,5. Ниже представлен рекомендуемый вид экрана во время работы программы:

X y
-2 -22,60
-1,5 -15,90
-1 -10,40
-0,5 -6,10
0 -3,00
0,5 -1,10
1 -0,40
1,5 -0,90
2 -2,60

2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы

Вариант 7

1. Написать программу, которая вводит с клавиатуры 7 дробных чисел и вычисляет сумму положительных чисел и произведение отрицательных чисел. Рекомендуемый вид экрана во время работы программы приведен ниже:

Вычисление среднего арифметического последовательности дробных чисел.
После ввода каждого числа нажимайте <Enter>
-> 1.4
-> 7.8
-> 3.0
-> -7,6
-> -9,2
-> 1.5
-> 2.3
Сумма положительных чисел равна =
Произведение отрицательных чисел равно =
2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы

Вариант 8

1. Написать программу, которая вычисляет среднее арифметическое последовательности дробных

чисел, вводимых с клавиатуры. После того, как будет введено последнее число, программа должна вывести минимальное и максимальное число последовательности.

Количество чисел должно задаваться во время работы программы. Рекомендуемый вид экрана приведен ниже. Данные, введенные пользователем, выделены полужирным шрифтом.

Обработка последовательности дробных чисел. Введите количество чисел последовательности -> 5

Вводите последовательность. После ввода каждого числа нажимайте <Enter> -> 5.4 ->7.8 -> 3.0 -> 1.5 -> 2.3

Количество чисел: 5

Среднее арифметическое: 4.00

Минимальное число:

Максимальное число:

2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы

Вариант 9

1. Написать программу, которая выводит таблицу значений функции $y=-9x^2+2x$ в диапазоне от – 3 до 3 с шагом 1. Ниже представлен рекомендуемый вид экрана во время работы программы:

X y
-3 -87
-2 -40
-1 -11
0 0
1 -7
2 -32
3 -75

2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы

Вариант 10

1. Написать программу, которая вычисляет произведение последовательности целых чисел, вводимых с клавиатуры. После того, как будет введено последнее число, программа должна вывести минимальное и максимальное число последовательности.

Количество чисел должно задаваться во время работы программы. Рекомендуемый вид экрана приведен ниже. Данные, введенные пользователем, выделены полужирным шрифтом.

Обработка последовательности дробных чисел. Введите количество чисел последовательности -> 5

Вводите последовательность. После ввода каждого числа нажимайте <Enter> -> 5 -> 7-> 3 -> 1 -> 2

Количество чисел: 5

Произведение: 210

Минимальное число:

Максимальное число:

2. Оформить отчет. Отчет должен состоять из кода программы, блок-схемы

Задание 2. Составьте программу, используя оператор Repeat по варианту, предложенному преподавателем.

Вариант 1

Написать программу, вычисляющую произведение положительных чисел, которые вводятся с клавиатуры, используя оператор Repeat. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление произведения положительных чисел.

Вводите после стрелки числа. Для завершения ввода введите ноль.

-> 45 -> 23 -> 15

Введено чисел: 3

Произведение чисел =

Вариант 2

Написать программу, которая определяет максимальное число из введенной с клавиатуры последовательности положительных чисел (длина последовательности не ограничена). Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Определение максимального числа последовательности положительных чисел.

Вводите после стрелки числа. Для завершения ввода введите ноль.

-> 56

-> 75

-> 43

-> О

Максимальное число: 75.

Вариант 3

Написать программу, которая определяет минимальное число из введенной с клавиатуры последовательности положительных чисел (длина последовательности не ограничена). Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Определение максимального числа последовательности положительных чисел.

Вводите после стрелки числа. Для завершения ввода введите ноль.

-> 6

-> 75

-> 3

-> O

Максимальное число:

Вариант 4

Написать программу, вычисляющую сумму отрицательных чисел, которые вводятся с клавиатуры, используя оператор Repeat. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление произведения положительных чисел.

Вводите после стрелки числа. Для завершения ввода введите ноль.

-> 45 -> 23 -> 15-> 5 -> 2 -> 1

Сумма чисел =

Вариант 5

Используя цикл Repeat, напишите программу, которая требует ввод пароля, например, числа 111, и если пароль правильный, то выдает на экран сообщение «Вы правильно ввели пароль». Пароль можно вводить три раза.

Вариант 6

Используя цикл Repeat, напишите программу определения идеального веса для взрослых людей по формуле: Ид. Вес = рост – 100.Выход из цикла значение роста = 250

Вариант 7

Написать программу, вычисляющую сумму и среднее арифметическое последовательности положительных чисел, которые вводятся с клавиатуры. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление среднего арифметического последовательности положительных чисел.

Вводите после стрелки числа. Для завершения ввода введите ноль.

-> 4 -> 230-> 15

Введено чисел:

Сумма чисел:

Среднее арифметическое:

Задание 3. Составьте программы, с использованием оператора While по варианту, предложенному преподавателем.

Программа, выводит таблицу значений функции для аргумента, изменяющегося в заданных пределах с заданным шагом.

Вариант	Функция	
1.	$y = \frac{\sin(3x)}{x^2},$	0<x<10
2.	$y = 1 - x + \frac{x^2}{2} + 5x^4,$	-5<x<6
3.	$y = \cos^2 x + \sin 5x$	-3<x<3
4.	$y = -4\sin(x+5) * \cos x$	-6<x<6
5.	$y = \frac{\sin(\frac{\pi}{2} - 5x)}{x^3}$	-5<x<5
6.	$y = \frac{\sin 2x + \sin 6x}{x^4}$	-3<x<6
7.	$y = \cos^3 x + \sin 10x$	-2<x<5
8.	$y = 2\sqrt{\cos x} - tgx$	-5<x<2
9.	$y = \frac{\sin 10x + x^2 - \cos(6x-2)}{x^4}$	-4<x<2
10.	$y = 2\sqrt{\cos 5x} - \sin x$	-6<x<6

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. В каких случаях в программе необходимо использовать итерационный цикл, а в каких регулярный цикл?
2. Назовите отличия итерационных циклов и цикла с параметром.
3. Какова структура оператора цикла с параметром? Как выполняется цикл с парметром?
4. Какого типа должны быть пареметр цикла, его начальное и конечное значения в цикле с параметром в языке Pascal?
5. Могут ли параметр цикла, его начальное и конечное значения в цикле с параметром в языке Pascal быть разных типов? Обоснуйте ответ.
6. Чем отличается цикл «До» от цикла «Пока»?
7. Сколько раз повторится итерационный цикл?
8. Какова структура цикла с постропроверкой условия?
9. Какова структура цикла с предпроверкой условия?

ПРАКТИЧЕСКАЯ РАБОТА 4-5

ТЕМА: Разработка программ с использованием одномерных массивов и указателей. Сортировка одномерных массивов.

Цель работы: Изучить принципы работы с одномерными массивами на языке программирования Pascal. Получение навыков применения основных алгоритмов для решения задач с использованием массивов.

Оборудование: ПК, ИСР Pascal ABC

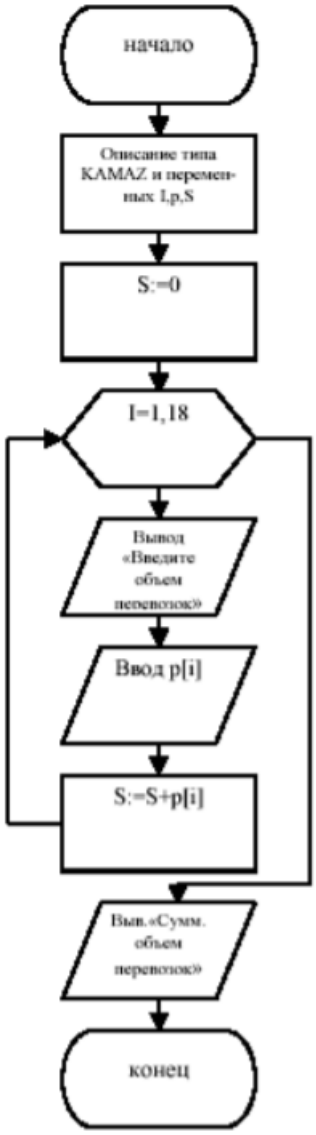
ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Нахождение суммы элементов массива

Задача 1. В автопарке, имеющем 18 машин марки КАМАЗ, каждый из КАМАЗов перевез за день определенный объем груза. Определить суммарный объем перевозок грузов за день. При решении задачи будем использовать тип массива KAMAZ для описания всех КАМАЗов автопарка; переменную P[i] для описания объема груза, перевезенного i-той машиной за день (I меняется от 1 до 18). Блок-схема алгоритма для решения данной задачи будет выглядеть следующим образом (см. Рис. 1): Текст следующий вид:

```
Program Pr1;
Uses wincrt;
Type KAMAZ=array [1..18] of real;
Var
i:integer;
p:KAMAZ;
S:real;
Begin
S:=0;
For i:=1 to 18 do
Begin
Writeln('Введите объем перевозок',I,'-ой машины, т');
Readln(p[i]);
S:=S+p[i]; End;
Writeln('Суммарный объем перевозок S=',S:8:2,'т');
End.
```

Накопление суммы в данном примере будет проводиться по шагам, при вводе значения объема перевозок для очередной машины сумма будет увеличиваться на данную величину. Аналогично реализуется и алгоритм нахождения произведения элементов массива (с заменой начального значения суммы S:=0 на начальное значение произведения S:=1 и с заменой операции сложения элементов массива «+» на операцию умножения «*»).



Нахождение количества элементов массива, удовлетворяющих заданному условию

Задача 2. Известны результаты экзамена по информатике одной группы из 22 студентов. Определить, сколько студентов сдали экзамен на 4 и 5. Один из вариантов решения этой задачи следующий:

На Рис. 2 представлена блок-схема алгоритма поставленной задачи.

Текст программы:

```
Program pr3;
Uses wincrt;
Label 1;
Type INF=array[1..22] of integer;
Var
stud:INF;
i,p:integer;
begin
p:=0;
for i:=1 to 22 do
begin
1: writeln('Введите оценку ',i,'-го студента');
readln(stud[i]);
if (stud[i]<1) or (stud[i]>5) then goto 1;
if stud[i]>3 then p:=p+1;
end;
writeln('На 4 и 5 сдали экзамен ',p:2,' студентов');
end.
```

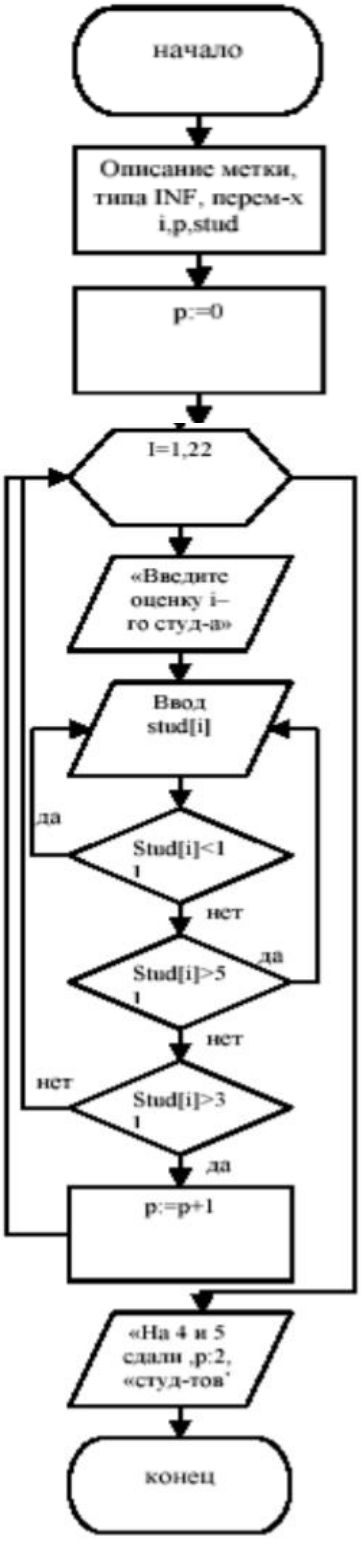


Рис. 2. Блок-схема программы

В данной программе для обозначения списка оценок по информатике использовался тип массива INF, для обозначения оценок конкретных студентов – переменная stud. Программа предусматривает проверку корректности вводимых данных: при попытке ввода несуществующей по пятибалльной системе оценки, программа повторяет ее ввод. Для этого используется оператор перехода GOTO, где имя метки, к которой осуществляется переход (в данном случае 1), описывается в разделе описания меток LABEL.

СОРТИРОВКА МАССИВА ПО ВОЗРАСТАНИЮ

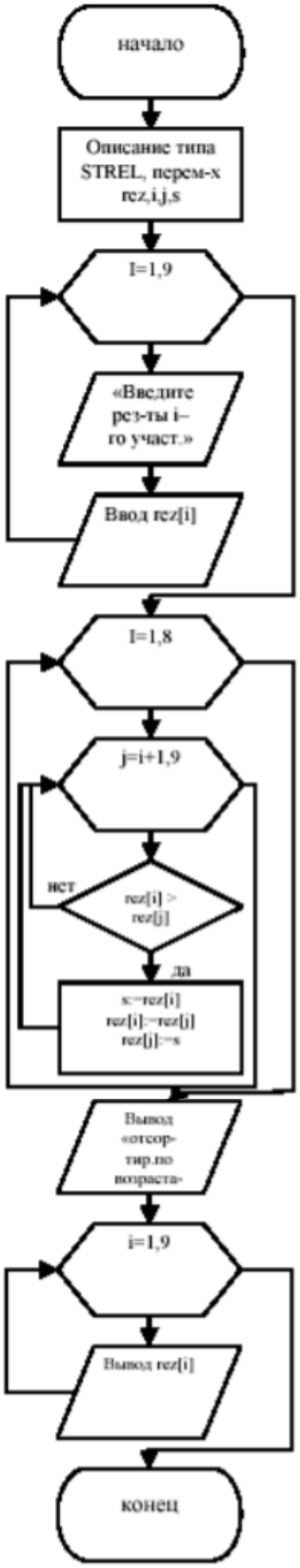
Задача 3. Предположим, известны результаты соревнований по стрельбе, в которых принимали участие 9 человек. Расположить данные результаты в порядке возрастания набранных при стрельбе очков. Алгоритм решения данной задачи является наиболее сложным из приведенных выше примеров и требует использования вложенных 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.

Блок-схема представлена на Рис. 3



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
рис. 3. Блок-схема меняется от 1 до 9).
34

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 объемы 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 студентов (в группе 20 студентов). Подсчитать количество студентов, 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 студентов (в группе 20 студентов). Подсчитать количество студентов, 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 студентов. Подсчитать количество студентов, не сдавших экзамен, в численном и в процентном соотношении.

ПРАКТИЧЕСКАЯ РАБОТА 6
ТЕМА: РАЗРАБОТКА ПРОГРАММ С ИСПОЛЬЗОВАНИЕМ ДВУМЕРНЫХ МАССИВОВ. СОРТИРОВКА ДВУМЕРНЫХ МАССИВОВ.

Цель работы: Изучить принципы работы с многомерными массивами на языке программирования Pascal. Получение навыков применения основных алгоритмов для решения задач с использованием массивов.

Оборудование: ПК, ИСР Pascal ABC

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

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_индекса 2..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 B[1,1], B[2,2]< B[3,3] составляют главную диагональ матрицы. Иногда вводят понятие побочной диагонали квадратной матрицы, которую составляют элементы B[1,N], B[2,N-1], B[3,N-2], .. B[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 B= 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. Решения данной задачи представлена на Рис. 1

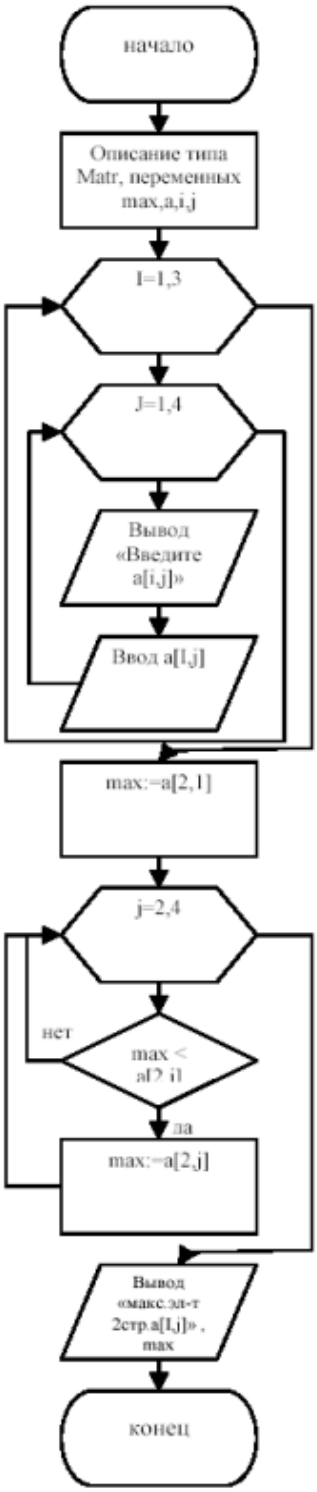


Рис. 1. Блок-схема программы Пусть задана матрица A из действительных чисел размера 3x4. Найти наибольший элемент во второй строке данной матрицы. Блок-схема алгоритма

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

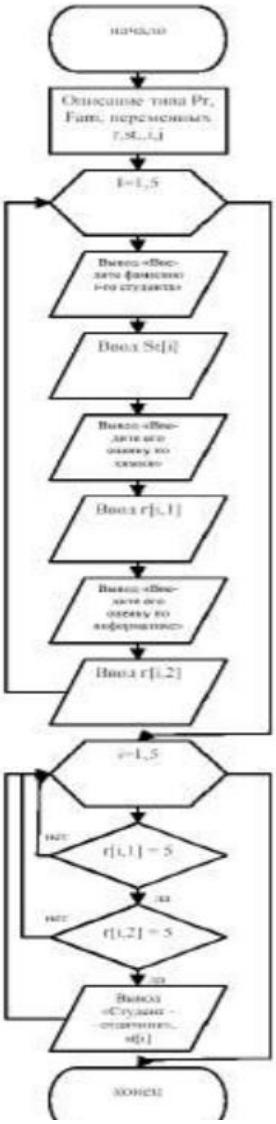


Рис. 2. Блок-схема программы

```
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 студента');
readln(st[i]);
writeln('Введите оценку данного студента по
химии (от 2 до 5)');
readln(r[i,1]);
writeln('Введите оценку данного студента 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('Студент-отличник - ',st[i]);
end.
```

В данной программе для хранения фамилий студентов используется одномерный строковый массив st типа Fam, для хранения оценок студентов – 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 студента оценки за оба экзамена составили 5 баллов, то его фамилия будет выведена на экран с сообщением «Студент-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 в ничью 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 (помесячно). Найти среднемесячное количество выпущенной продукции для каждого завода.
7. Известно количество выпущенной продукции тремя заводами за первый квартал (помесячно). Найти среднемесячное количество выпущенной продукции по всем заводам.
8. Известны результаты сдачи трех экзаменов пятью студентами. Найти фамилии студентов, 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 студентами. Найти фамилии студентов, 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 студентами. Найти средний балл каждого студента 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 студентами. Определить фамилии студентов, сдавших экзамены без троек.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что понимают под массивом данных?
2. Что называют размерностью массива?
3. Что понимают под индексом элемента массива?
4. Какой массив называется одномерным?
5. Приведите примеры одномерных массивов.
6. Как описываются одномерные массивы на языке PASCAL?
7. Как задается диапазон изменения индексов массива?
8. Как обозначаются индексы массивов на языке PASCAL?
9. Какие стандартные алгоритмы по работе с одномерными массивами Вы знаете?
11. Поясните понятия двумерного массива, матрицы.
12. Что обозначают индексы матрицы?
13. Сколько элементов в матрице из 7 строк и 9 столбцов?
14. Дайте понятие квадратной матрицы, диагоналей квадратной матрицы.
15. Приведите пример описания двумерных массивов на языке PASCAL.
16. Поясните порядок использования вложенных циклов при вводе элементов двумерного массива

ПРАКТИЧЕСКАЯ РАБОТА 7

ТЕМА: ИСПОЛЬЗОВАНИЕ СТАНДАРТНЫХ ФУНКЦИЙ И ПРОЦЕДУР ДЛЯ РАБОТЫ СО СТРОКАМИ

Цель работы

Целью работы является приобретение навыков алгоритмизации и программирования задач, оперирующих строковыми типами данных:

- ввод и вывод строковых данных;
- обработка строковых данных;
- использование стандартных процедур и функций языка Pascal для обработки строковых данных.

Оборудование: ПК, ИСР Pascal ABC

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Обработка строковых данных является необходимым элементом программ, работающих с текстами. К таким программам относятся программы лингвистического анализа текстов, текстовые редакторы, программы, работающие в диалоговом (интерактивном) режиме, программы, содержащие текстовые пояснения результатов своей работы.

Строка – последовательность символов (от 0 до 255), заключенная в апострофы. При составлении программ используются строковые константы и строковые переменные.

1. Определение через описание типа в разделе описания типов.

Формат

TYPE

<имя типа> = STRING [максимальная длина строки];

VAR

<идентификатор1, идентификатор2,…> : <имя типа>;

где

STRING – зарезервированное слова (строка);

Максимальная длина строки – наибольшее допустимое количество символов переменной данного типа (. 255).

Например,

```
TYPE
FLO = STRING [130];
FTK = STRING; {По умолчанию длина строки равна 255}
VAR
ST1 : FLO;
ST2, ST3 : FTK;
```

2. Определение непосредственно в разделе описания переменных

Формат

```
VAR
ST4, ST5 : STRING [60];
ST6, ST7 : STRING;
```

ПРИМЕР ЗАДАЧИ

Разработать программу, удаляющую из вводимой с клавиатуры строки пробелы между словами и записывающую в массив N длину (число символов) каждого слова. Длина текста – не более 80 символов. Число слов – не более 10. Наличие более одного символа ‘пробел’ подряд свидетельствует о конце строки.

Используемые в программе идентификаторы приведены в таблице 1.

Таблица 1

Обозначения	Тип данных	Примечание
A	STRING	Исходный текст, символьные данные
K	INTEGER	Количество символов в слове
L	INTEGER	Порядковый номер слова
I	INTEGER	Параметр цикла
A[I]		Текущий символ исходного текста
N	ARRAY [1…10] OF INTEGER	Массив, содержащий значения длины каждого слова исходного текста
N[L]		Значение длины слова номер L
J	INTEGER	Параметр цикла, используемого для перемещения всех следующих символов исходного текста на одну позицию влево после того, как обработано очередное слова.
PR	INTEGER	Переменная для управления повторной работой программы
OTVET	BYTE	Переменная для управления началом обработки введенной строки

Листинг программы

```
Program Prim1;
Uses Crt;
Label 4;
VAR
N: ARRAY [1..10] OF INTEGER;
I, J, K, L: INTEGER;
A: STRING [80];
PR, OTVET:BYTE;
BEGIN
CLRSCR;
REPEAT
```

```
REPEAT
WRITELN (' Введите через пробел');
READLN (A);
WRITELN('Исходная строка');
WRITELN(A);
WRITELN ('Работаем дальше? 1 -да,0 -нет');
READLN (OTVET);
UNTIL OTVET=1;
K:=0;
L:=0;
PR:=0;
FOR I:=1 TO length(a) DO
IF (A[I]= ' ') THEN
BEGIN
L:=L+1;
N[L]:=K;
IF (A[I+1]=' ')THEN GOTO 4;
FOR J:=i TO length(a) DO
A[J]:= A[J+1];
K:=1
END
ELSE
begin
K:=K+1;
N[L+1]:= K-l;
end;
4:WRITELN ('Результирующая строка');
WRITELN (A);
WRITELN ('№ слова число букв');
FOR I:=1 TO L+1 DO
WRITELN (' N[',I,'] =',N[I]:6);
WRITELN('Обработать еще одну строку? 1 –да 0 -нет');
READLN(PR);
UNTIL PR=0
END.
```

Протокол работы программы

Протокол работы программы показан на рис. 1.

```
 Введите через пробел
ААА ВВВВ СССССС НННННННН
Исходная строка
ААА ВВВВ СССССС НННННННН
Работаем дальше? 1- да, 0 - нет
1
Результирующая строка
ААААВВВВСССССННННННН
№ слова       число букв
   N[1] =     3
   N[2] =     4
   N[3] =     5
   N[4] =     7

Обработать еще одну строку? 1 - да 0 - нет
0
```

Рис. 1. Протокол работы программы

ЗАДАНИЕ

1. В заданном тексте удалить символ ‘,’ и подсчитать число удаленных символов. Предусмотреть возможность задания с клавиатуры удаляемого символа.

2. В заданном тексте заменить словосочетание «свернутые обороты» на словосочетание «разделенные обороты» и подсчитать число произведенных замен.

3. Из заданного предложения выбрать и вывести на экран только те символы, которые встречаются в нем только один раз (в том порядке, в каком они встречаются в тексте) 4. В заданном предложении найти самое длинное и самое короткое слова и подсчитать, на сколько больше символов в самом длинном слове.

5. Проверить, встречается ли в заданном предложении словосочетание «Остаток счета».

6. Проверить, встречается ли в заданном предложении словосочетание «Сальдо счета».

7. Для каждого слова заданного предложения указать долю согласных букв. Определить слово, в котором доля согласных максимальна.

8. Из заданного текста выбрать цифры и записать в массив N, а буквы- в массив B. Все остальные символы записать в массив S.

9. Удалить из заданного текста все пробелы, подсчитать длину получившегося текста и число удаленных пробелов.

10. Для заданного предложения указать слова, в котором доля гласных букв максимальна.

11. Отредактировать предложение, удаляя из него лишние пробелы, оставив только по одному пробелу между словами. Подсчитать число удаленных пробелов.

12. Проверить, имеется ли в заданном тексте баланс открывающихся и закрывающихся круглых скобок.

13. Дана последовательность из 10 слов. Вывести слова в обратном порядке.

14. Дана последовательность из 10 слов. Вывести все слова, входящие в эту последовательность по одному разу.

15. Дана последовательность из 8 слов. Вывести входящие в эту последовательность слова, расположив их по алфавиту.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое строка?
2. Для чего в программировании используются строки?
3. Как объявляется строка в программе?
4. Как обозначается элемент строки?
5. Назовите процедуры для обработки строк
6. Назовите функции для обработки строк

ПРАКТИЧЕСКАЯ РАБОТА 8
ТЕМА: ОРГАНИЗАЦИЯ ПРОГРАММ С ИСПОЛЬЗОВАНИЕ ПРОЦЕДУР. ОРГАНИЗАЦИЯ ПРОГРАММ С ИСПОЛЬЗОВАНИЕМ ФУНКЦИЙ

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.

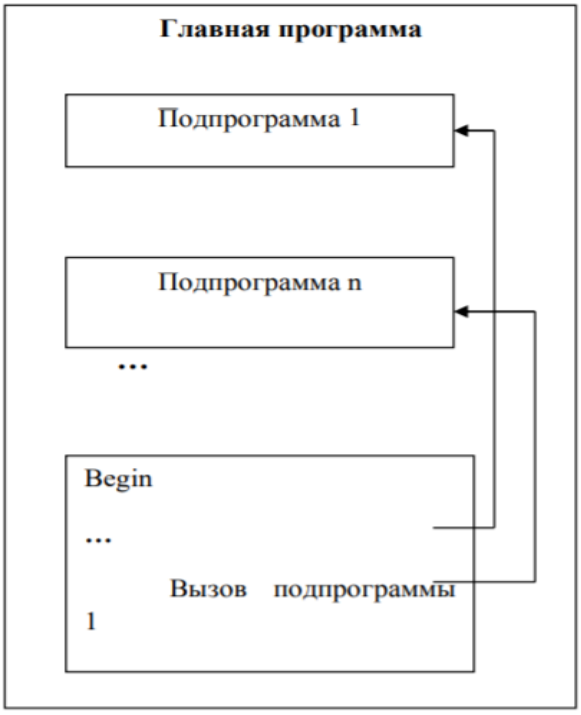


Рис. 1 Структура программы с подпрограммами

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 определѐнной в ней последовательности действий. Функция отличается от процедуры тем, что возвращает результат указанного при еѐ 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 еѐ 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 еѐ тело не передаются.

Для вызова процедуры на исполнение к ней необходимо обратится.

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 отличии 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 передаèтся 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 передаèт 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: НОД(x, y, z) = НОД(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, к – 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 4x4. 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 4х3. 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 3х3. 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 3х3, 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 4х2 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 4х4. 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 №9

Тема: Методы тестирования программного обеспечения.

Цель. Изучить процессы тестирования и отладки программного обеспечения.

Оборудование. ПК

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

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.

Контрольные вопросы

1. Какие методы тестирования вы знаете?
2. В чем заключаются методы «черного» и «белого» ящика?
3. На каком этапе проводится ручная отладка?
4. Опишите методы отладки.

Лабораторный практикум
По дисциплине Алгоритмизация и программирование

Лабораторная работа 1-4
Тема: СИСТЕМА ПРОГРАММИРОВАНИЯ PASCAL

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 COMPILE

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 составляются 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,l,2,3,4,5,6,7,8,9,A,B,C,D,E,F;

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

Следующие комбинации специальных символов являются едиными 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-l)-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 аргумента	Тип результата	Функция
Pi		R	Число π=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:

Tg(x)=sin(x)/cos(x)

Lg(x)=ln(x)/ln(10)

X^n=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 А&Б, где & - знак операции отношения: > - больше; >= -больше либо равно; < - меньше ; <= - меньше либо равно;

= - равно; ,<>- не равно.

А и Б - 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

А	TRUE	TRUE	FALSE	FALSE
Б	TRUE	FALSE	TRUE	FALSE
NOT A	FALSE	FALSE	TRUE	TRUE
А AND Б	TRUE	FALSE	FALSE	FALSE
А OR Б	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 числа в виде вещественных констант без экспонент.

1) -10^3 2) 10^5
3) $211{,}02\cdot10^4$ 4) $314\cdot2^6$
5) $256\cdot2^6$ 6) $0{,}26\cdot10^{-8}$
7) 10^{-9} 8) 99,9
9) -0,0556 10) -16,301
11) -39,64362802 12) 1,1
13) -0 14) 646362,7670021
15) $0{,}00345\cdot10^4$

Задание 2. Представить приведенные ниже числа в виде вещественных констант с экспонентами.

1)0,2360027 2)-10^{-4}
3)164273 4)-0,0345
5)-0,0556 6)-39,64362802
7)$211{,}02\cdot10^{-2}$ 8)$0{,}26\cdot10^8$
9)$867342{,}17\cdot10^3$ 10)0,00053
11)$-3{,}1\cdot10^6$ 12)$-715{,}1\cdot10^{-3}$
13)$26{,}345\cdot10^{-4}$ 14)-0,0043
15)$4{,}3\cdot10^{-5}$

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. $Y=\dfrac{a\sqrt{b+4ac}-c^3}{2a}$

2. $Z=\dfrac{a}{c}\,x\,\dfrac{b}{d}-\dfrac{ab-c}{cd}$;

3. $Z=\dfrac{\sin x+\cos x}{\cos x-\sin x}tg\,x$

4. $Z=\dfrac{x+y}{y+1}-\dfrac{xy+1}{3+4x}$;

5. $Z=\dfrac{3+e^{y-1}}{1+x^2\lvert y-tg\,\phi\rvert}$;

6. $A=x-\dfrac{x^3}{3}+\dfrac{x^5}{5}$;

7. $Z=2\sin(x)-\dfrac{1}{1+2x+5}$;

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

10. $Z=\dfrac{\ln|\cos x|}{\ln(1+x^2)}$;

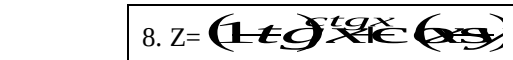
11. $Z=\left(\dfrac{x+1}{x-1}\right)^x+1{,}8y^2$;

12. $Z=\left(1+\dfrac{1}{x^2}\right)^x-1{,}2y^3$;

13. $Z=\dfrac{x^2-7x+10}{x^2-8x+12}$;

14. $Z=\dfrac{\cos(x)-1{,}6\ln(x)}{\pi+2x}$;

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

8. Z= 	

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 строчки, 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 строчку, а после выполнения инструкции Write, переход на новую строчку не происходит и печать по последующим командам вывода Write или Writeln будет происходить на той же строчке. При вызове оператора Writeln без параметров просто происходит переход на новую строчку.

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, умещающихся 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

Решение:

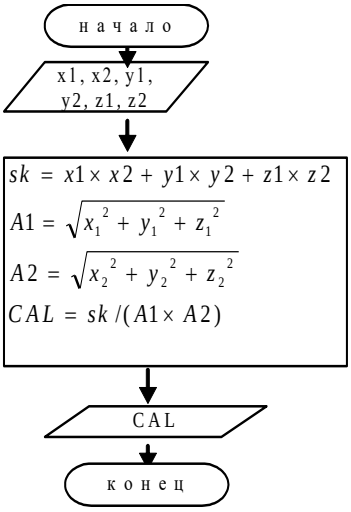
```
PROGRAM P1;
VAR X,Y,Z: REAL;
BEGIN
WRITELN ('ВВЕДИТЕ X, У' );
READLN (X, Y);
Z:=LN(ABS(COS(X)))-EXP(-X*LN(2))+SIN(2*X*Y)
WRITELN ( 'Z=', Z);

END.
```

Пример 2. Заданы два вектора с координатами $\overline{a_1}=(x_1,y_1,z_1)$ и $\overline{a_2}=(x_2,y_2,z_2)$. Определить косинус угла между векторами.

Решение: Косинус угла между двумя векторами $\vec{a_1},\vec{a_2}$ определяется по формуле: $\cos\alpha=\frac{(a_1,a_2)}{|a_1||a_2|}$, где

$(a_1,a_2)=x_1x_2+y_1y_2+z_1z_2$ - скалярное произведение, $|\overline{a_1}|=\sqrt{x_1^2+y_1^2+z_1^2}$,

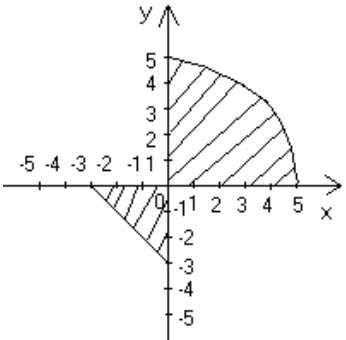
$|\overline{a_2}|=\sqrt{x_2^2+y_2^2+z_2^2}$ - модули векторов. Введем обозначение: $sk=(\overline{a_1},\overline{a_2})$ $A1=|\overline{a_1}|$, $A2=|\overline{a_2}|$, $CAL=\cos\alpha$.

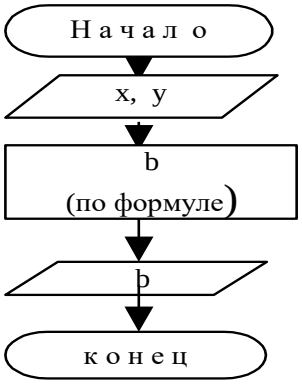



```
PROGRAMM P2;
VAR X1, Y1, Z1, X2, Y2, Z2:REAL;
  A1, A2,  SK, CAL,  ALFA: REAL;
BEGIN


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:=SORT (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.





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

апишем систему неравенств в виде одного огического выражения

$b = ((x^2 +$ [illegible]

```
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. $Z = \frac{b\sqrt{B+4ac}}{2a} - \frac{3}{ac+b}$;

2. $Z = \frac{a}{c} \times \frac{b}{d} - \frac{ab+c}{cd}$;

3. $Z = \frac{\sin x + \cos x}{\cos x - \sin y} tg$;

4. $Z = \frac{x+y}{y+1} - \frac{xy+1}{34x}$;

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

6. $Z = x - \frac{x^3}{3} + \frac{x^5}{5}$;

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

8. $Z = \frac{1+\sin\sqrt{x+1}}{\cos(1,2y-4)}$;

9. $Z = (1+tg)^{\sin x} + \cos y$;

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

11. $Z = \left(\frac{x+1}{x-1}\right)^x + 18y^2$;

12. $Z = \left(1+\frac{1}{x^2}\right)^x - 1,2y^3$;

13. $Z = \frac{x^2-7x+10}{x^2-8x+12}$;

14. $Z = \frac{\cos 1,6(x+y)}{\pi+2x} + 2$;

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

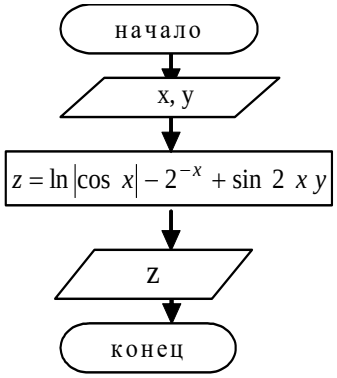
Задание 2.

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 $\mathbf{ax^2+bx+c=0}$ с заданными коэффициентами **a, b** и с (предполагается, что $\mathbf{a \neq 0}$ 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 $\mathbf{2x^4-3x^3+4x^2-5x+6}$.

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.

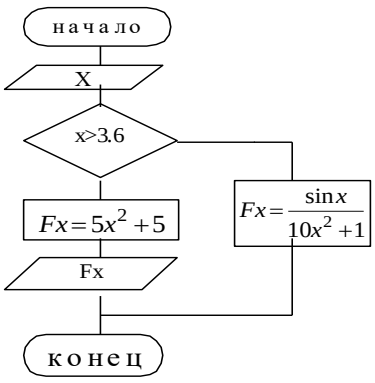
```
    Program Sq1;
Var A, B, C, D, X1, X2 : Real;
Begin
    Writeln ('Введите коэффициенты квадратного уравнения');
Readln (A,B,C);
D:=B*B-4*A*C;
If D<0 Then Writeln ('Корней нет! ')
Else
Begin
    X1:=(-B+SQRT(D))/2/A;
X2:=(-B-SQRT(D))/2/A;
Writeln ('X1=', X1:8:3, ' X2=',X2:8:3)
    End
    End.
```

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.

$$F(x)=\begin{cases} 5x^2+5, & \text{если } x>3.6 \\ \dfrac{\sin x}{10x^2+1}, & \text{если } x\le 3.6 \end{cases}$$

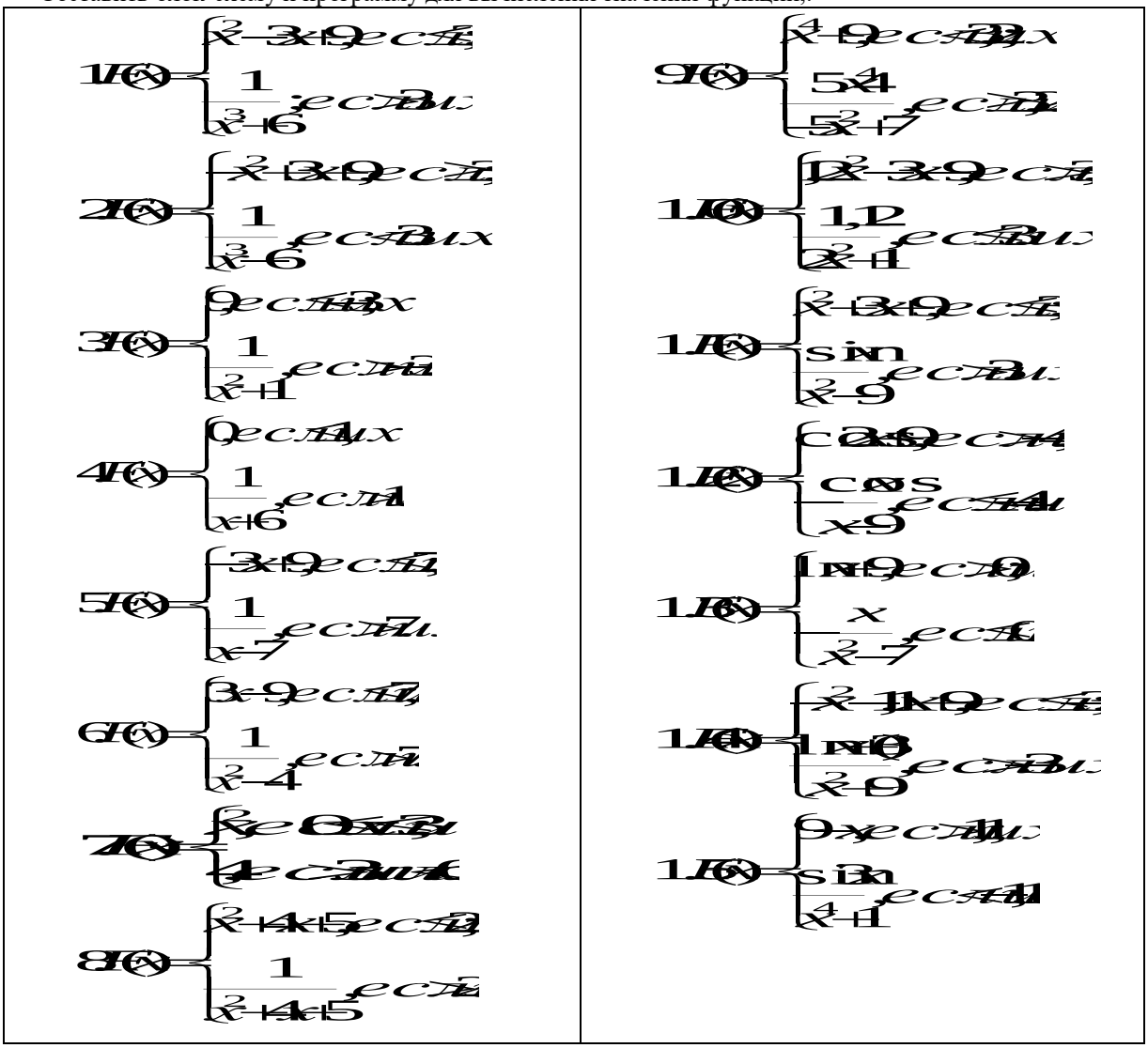
БЛОК-СХЕМА




```
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.
```

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 студента 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 соответсвующее 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≤m2, затем осуществляется выход из цикла к оператору, следующему за оператором цикла.

Б) For I:=m1 DownTo m2 Do S;

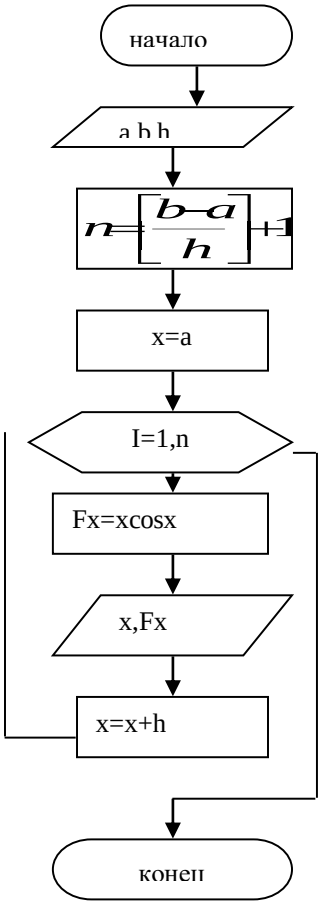
Вторая форма оператора цикла с параметром используется в случае, когда m1>m2. При этом шаг изменения параметра цикла I равен -1.

Пример 1. Вычислить значения функции Fx на отрезке [a,b] с шагом h. Fx=xcosx

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:

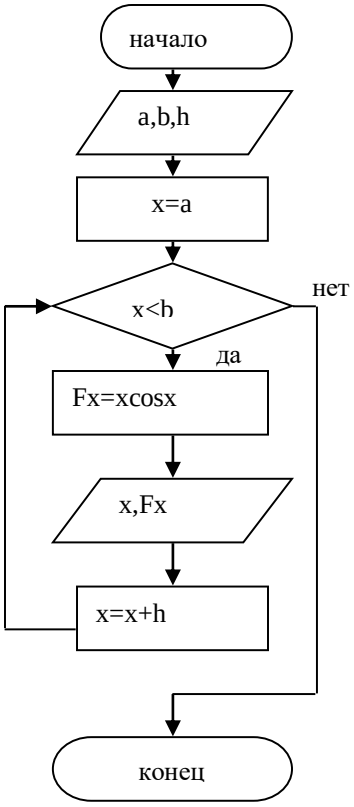
While < логическое выражение> Do S;

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:

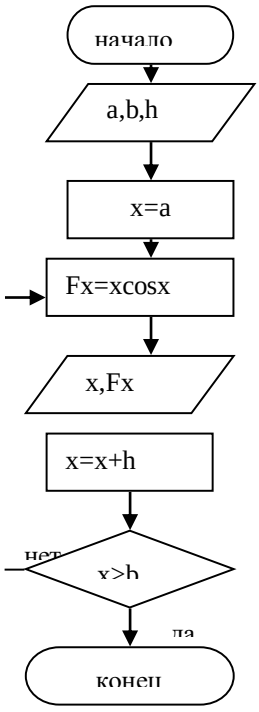
Repeat S Until < логическое выражение> ;

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.F(x)=x-sin x
2.F(x)=sin2 x
3 F(x)=2cosx-1
4.F(x)=tg x
5. F(x)=ctgx+1
6. F(x)=2 sin2x+1
7. F(x)=sin x+tg x
8. F(x)=cos x+tg x
9. F(x)=2 tg (x/2)+1
10. F(x)=2 tg (x/2)+2 cosx
11. F(x)=sin2x-cos2x
12. F(x)=7sin2x-1/2 cos x
13. F(x)=-cos 2x
14. F(x)=tg 2x-3
15. F(x)=sinx+0,5 cosx

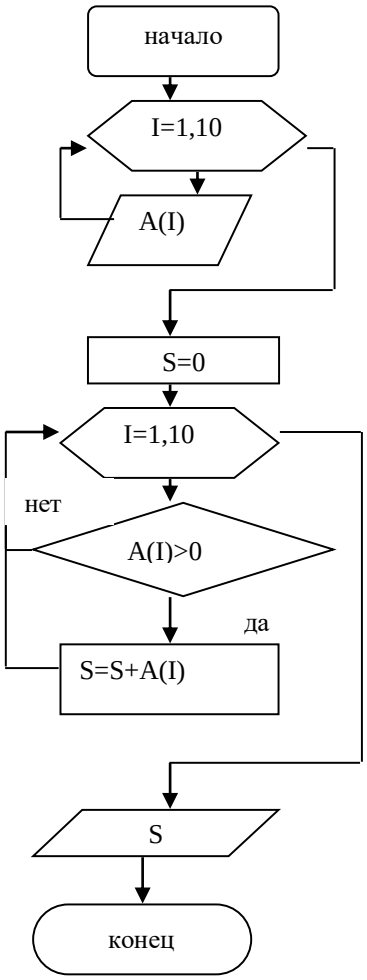
Лабораторная работа №8

Разработка программ с использованием одномерных массивов и указателей.

Вычисление суммы. При вычислении суммы конечного числа слагаемых необходимо перед циклом задать начальное значение для суммы равное нулю, S=0. Затем внутри цикла нужно накапливать сумму по формуле S=S+Y, где Y- очередное слагаемое, S- текущее значение суммы.

Пример. Вычислить сумму положительных элементов действительного массива A(10).

Блок-схема



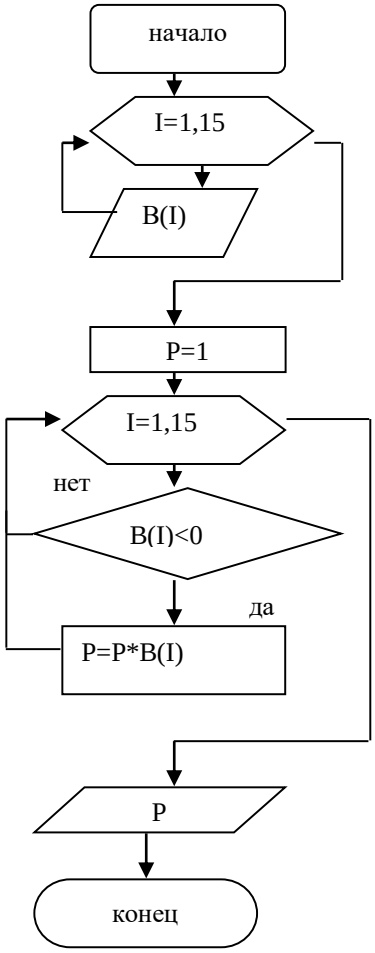
Программа

```
PROGRAM  WS;
VAR A: ARRAY[1..10] OF REAL;
I:INTEGER; S:REAL;
BEGIN
WRITELN('Введите элементы массива');
FOR I:=1 TO 10 DO
READ(A[I]);
S:=0;
FOR I:=1 TO 10 DO
IF A[I]>0 THEN S:=S+A[I];
WRITELN(S);
END.
```

Вычисление произведения. При вычислении произведения конечного числа сомножителей необходимо перед циклом задать начальное значение произведения, равное 1, P=1. Затем внутри цикла нужно вычислять произведение по формуле P=P*Y.

Пример. Вычислить произведение отрицательных элементов действительного массива B(15)

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 $(B_1, B_2, .. B_{15})$, значения, которых больше 2 и меньше 10.

б) Найти произведение элементов массива $(B_1, B_2, .. B_{15})$, стоящих на четных местах.

5. а) Определить количество нулевых элементов массива $(A_1, A_2, .. A_{10})$.

   б) Найти произведение положительных элементов массива $(D_1, D_2, .. D_8)$, стоящих на четных местах.
6. а) Найти сумму элементов целочисленного массива $(Z_1, Z_2, .. Z_{20})$, значения, которых кратны 3.

   б) Вычислить произведение элементов массива $(N_1, N_2, .. N_{10})$, и разделить его на 2.
7. а) Вычислить сумму элементов массива $(a_1, a_2, .. a_{15})$, значения которых больше 2 и меньше 7.

   б) Вычислить $y = \prod_{i=1}^{10} x + \sin x$
8. а) Найти количество положительных элементов массива $(Z_1, Z_2, .. Z_9)$, и разделить его на 2.

   б) Найти произведение элементов массива $(x_1, x_2, .. x_{15})$, индексы которых кратны 3.
9. а) Составить программу для вычисления среднего арифметического отрицательных элементов массива $(Z_1, Z_2, .. Z_{18})$

   б) Вычислить произведение элементов массива $(Y_1, Y_2, .. Y_3)$, модуль которых больше 5.

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

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

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

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

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

б) Вычислить $Z = \prod_{i=1}^{15} (y_i + 2) y_i \in$.

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

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

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

б) Найти произведение и количество отрицательных элементов массива $(Z_1, Z_2, \ldots Z_8)$.

15. а) Дан массив целых чисел $(J_1, J_2, \ldots J_{13})$, найти сумму нечетных элементов этого массива.

б) Вычислить $R = \left(\prod_{i=1}^{15} \right) / N$, 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 объемы 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 студентов (в группе 20 студентов). Подсчитать количество студентов, 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 студентов (в группе 20 студентов). Подсчитать количество студентов, 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 студентов. Подсчитать количество студентов, 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_индекса 2..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 B[1,1], B[2,2]< B[3,3] составляют главную диагональ матрицы. Иногда вводят понятие побочной диагонали квадратной матрицы, которую составляют элементы B[1,N], B[2,N-1], B[3,N-2], .. B[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 B= 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

2.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 студентов по итогам экзаменов по химии и информатике. Найти фамилии студентов, 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 студента');
readln(st[i]);
writeln('Введите оценку данного студента по
химии (от 2 до 5)');
readln(r[i,1]);
writeln('Введите оценку данного студента 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('Студент-отличник - ',st[i]);
end.
```

В данной программе для хранения фамилий студентов используется одномерный строковый массив st типа Fam, для хранения оценок студентов – 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 студента оценки за оба экзамена составили 5 баллов, то его фамилия будет выведена на экран с сообщением «Студент-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 в ничью 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 (помесячно). Найти среднемесячное количество выпущенной продукции для каждого завода.

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

8. Известны результаты сдачи трех экзаменов пятью студентами. Найти фамилии студентов, 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 студентами. Найти фамилии студентов, 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 студентами. Найти средний балл каждого студента 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 студентами. Определить фамилии студентов, сдавших экзамены без троек.

Лабораторная работа №14 - 16
Разработка программ с использование 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.

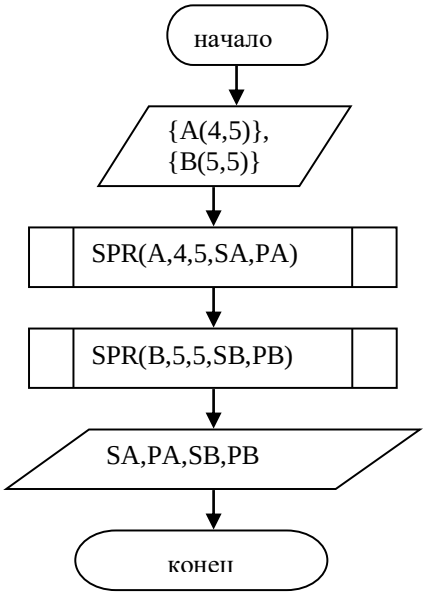
Глобальные переменные – это те переменные, которые объявлены в описании основной программы. Они доступны как в основной программе, так и во всех ее подпрограммах.

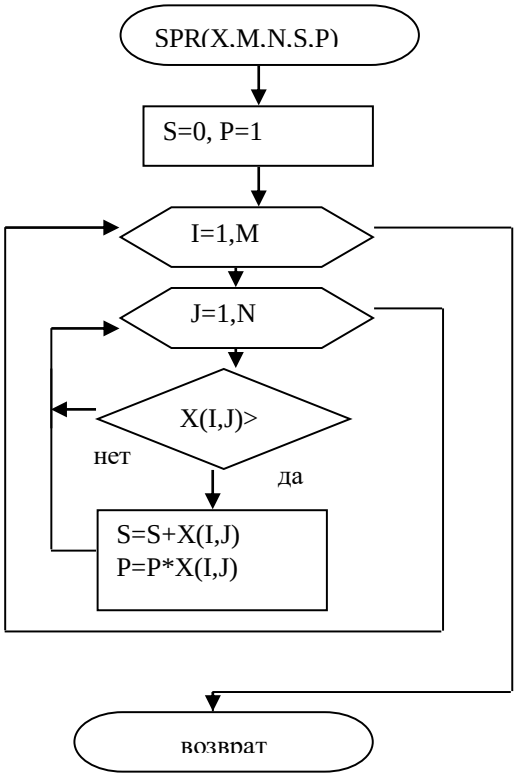
Локальные переменные- это те переменные, которые объявлены в подпрограммах. Они существуют только тогда, когда работает подпрограмма.

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).





Процедура SPR(X,M,N,S,P) вычисляет сумму и произведение положительных элементов матрицы X из M строк и N столбцов

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;
    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: $x^n = \begin{cases} 1, & n = 0 \\ x \cdot x^{n-1}, & n > 0 \end{cases}$

или $n! = \begin{cases} 1, & n = 0 \\ n \cdot (n-1)!, & n > 0 \end{cases}$

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 А(6) и В(10).

б) Вычислить сумму и количество отрицательных элементов действительных массивов Х(10) и У(5).

- а) Найти среднее арифметическое элементов целочисленных массивов С(10), D(15).

б) Найти максимальные элементы и их порядковые номера действительных массивов А(10) и В(5).

3. а) Найти сумму элементов для действительных массивов А(5,5) и В(4,6).

б) Найти сумму и количество отрицательных элементов массивов Х(5,6) и У(3,4).

4. а) Для действительных массивов А(5) и В(16) найти минимальные элементы.

б) Вычислить сумму и произведение элементов целочисленных массивов Х(3,2) и У(3,4).

5. а) Найти среднее арифметическое положительных элементов массивов Х(3,5) и У2,3).

б) Вычислить сумму и количество элементов больших 5 для действительных массивов В(20) и С(10).

6. а) Вычислить сумму элементов массивов Х(15) и У(10).

б) Вычислить произведение и сумму элементов массивов А(20) и В(10) .

7. а) Вычислить сумму элементов массивов А(20) и В(15) с нечетными номерами.

б) Вычислить произведение и количество элементов массивов Х(10) и У(12), модуль которых больше 9.

8. а) Вычислить количество элементов равных заданному числу х для массивов А(2,3) и В(3,4).

б) Вычислить сумму и произведение четных элементов массивов А(20) и В(10).

9. а) Вычислить сумму элементов массивов А(10) и В(15), значения, которых больше 2 и меньше 10.

б) Найти произведение и количество положительных элементов массива $(B_1, B_2, \ldots B_{15})$, стоящих на четных местах.

10. а) Определить количество нулевых элементов массива $(A_1, A_2, \ldots A_{10})$.

б) Найти произведения положительных и отрицательных элементов целочисленного массива $(D_1, D_2, \ldots D_8)$.

11. а) Найти сумму элементов целочисленного массива $(Z_1, Z_2, \ldots Z_{20})$, значения, которых кратны 3.

б) Вычислить количество и произведение отрицательных элементов действительной матрицы А(3,5).

12. а) Вычислить суммы элементов массивов $(a_1, a_2, \ldots a_{15})$ и $(x_1, x_2, \ldots x_{20})$, значения которых больше 2 и меньше 7.

б) Найти сумму и произведение положительных элементов массивов А(3,5) и В(5,5) .

13. а) Вычислить R = $S_1 + S_2 + S_3$, где S1= $\sum_{i=1}^{10} Ai$, S2 = $\sum_{i=1}^{15} Ci$, S3 = $\sum_{i=1}^{20} Di$

б) Составить программу для вычисления среднего арифметического отрицательных и среднего геометрического положительных элементов массивов $(Y_1, Y_2, \ldots Y_3)$ и $(Z_1, Z_2, \ldots Z_{18})$.

14. а). Вычислить произведение элементов массива $(C_1, C_2, \ldots C_{10})$, удовлетворяющих условию $C_i \prec i$.

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

15. а) Вычислить $A = \frac{3K + 2N^2}{\sqrt{4P}}$, где 

б) Найти сумму и количество положительных элементов массивов А(3,5) и В(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.

Вопросы к экзамену
По дисциплине Алгоритмизация и программирование

1. Понятие алгоритма и программы. Способы записи алгоритмов.
2. Общая характеристика языков программирования и их классификация.
3. Понятие о системе программирования. Трансляция программ.
4. Основные конструкции языка программирования. Язык программирования Паскаль. Структура программы
5. Понятие величины. Оператор присваивания. Объявление переменных.
6. Линейные программы. Ввод и вывод данных в языке Паскаль.
7. Простые типы данных и операции над ними (Паскаль).
8. Разветвляющиеся алгоритмы и программы. Реализация в языке Паскаль. Примеры.
9. Оператор выбора (варианта). Реализация в языках Паскаль.
10. Циклические алгоритмы и программы. Реализация в языке Паскаль
11. Понятие подпрограммы. Процедуры и функции.
12. Реализация процедур и функций в Паскале.
13. Массивы. Ввод и вывод элементов массива.
14. Типовые алгоритмы обработки массивов. Примеры.
15. Сортировка элементов массива.
16. Поиск элементов в массиве.
17. Приведите структуру программы на языке Паскаль.
18. Константы и переменные. Где и как объявляются. Их отличия.
19. Перечислите базовые типы данных в языке Паскаль. Формы представления вещественных чисел. Какие арифметические операции невозможны над данными вещественного типа?
20. Приведите таблицы стандартных функций возвращающих целый и вещественный результат.
21. Приведите форматы оператора ввода и оператора присваивания.
22. Приведите примеры операторов ввода и присваивания. Приведите формат оператора вывода. Приведите примеры оператора. Для чего нужно указание формата числа в операторе вывода. Приведите примеры. Отличия операторов Writeln и Write.
23. Приведите форматы полного и неполного условного оператора. Приведите примеры.
24. Приведите форматы операторов цикла While и Repeat. Приведите пример оператора. Их отличия.
25. Приведите форматы оператора цикла For. Приведите примеры оператора.
26. Что называется массивом? Как объявить одномерный массив? Приведите фрагмент ввода одномерного массива с клавиатуры. Приведите фрагмент формирования одномерного массива случайными числами и укажите, какой диапазон чисел будет использован. Приведите фрагмент вывода одномерного массива в строку.
27. Программирование - основные понятия (программа, трансляторы, интерпретаторы, компиляторы). Требования, предъявляемые к программе.
28. Анализ и проектирование- разработка комплекса алгоритмов. Фаза анализа, проектирования, реализации программы.
29. Этапы разработки программы: спецификация, разработка алгоритма, кодирование.
30. Отладка программы.
31. Обработка ошибок. Типы ошибок. Ошибки при выполнении программы.
32. Режимы отладки. Окна отладки
33. Тестирование. Уровни тестирования ПО: модульное, интеграционное, системное. Статическое и динамическое тестирование.
34. Компиляция.
35. Сопровождение ПО. Структура ИТ- сопровождения.
36. Структурное программирование. Понятие подпрограммы. Нисходящее, восходящее проектирование. Процедуры и функции VB. Примеры.
37. Двумерные массивы. Трехмерные массивы. Синтаксис. Примеры.

38. Одномерные массивы. Понятия: массив, структуры данных, одномерный, двумерный массив. Объявление. Пример описания одномерного массива.
39. Программирование ветвлений. Понятия: условное (логическое выражение), простое условие, сложное условие. Примеры
40. Оператор If. Однострочная и многострочная запись условного оператора. Примеры
41. Программирование повторений. Формат оператора GoTo Метка, For … Next. Примеры.
42. Программирование повторений. Вложенные циклы. Объявление. Примеры.
43. Цикл с условием. Форматы операторов с проверкой условия в начале цикла. Примеры.
44. Цикл с условием. Форматы операторов с проверкой условия в конце цикла. Примеры.
45. Экспертные системы. Понятие, архитектура ЭС, Черты экспертных систем. Характеристики ЭС.
46. Компоненты ЭС. База знаний. Машина вывода.
47. Предпосылки создания ЭС, выбор предметной области. Основное отличие от других ПК. Люди, причастные к ЭС.
48. Классификация ЭС по функциональному назначению, по связи с реальным временем. Примеры.
49. Классификация по степени интеграции с другими программами. Структура автономной ЭС: редактор БЗ, База знаний, программа вывода, программа объяснения, интерфейс.
50. Принципиальная схема работы экспертной системы. Этапы разработки ЭС.
51. Программные средства
52. Логическое программирование
53. Хакер. Защита информации.
54. Информационная безопасность.
55. Виды алгоритмов. Блок-схема

СЕВЕРО-КАВКАЗСКАЯ ГОСУДАРСТВЕННАЯ АКАДЕМИЯ

Кафедра Общей информатики

20____ – 20____ учебный год

Экзаменационный билет № 1

по дисциплине: Алгоритмизация и программирование

для обучающихся направления подготовки 09.03.04 - Программная инженерия

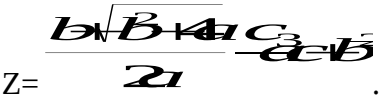
1. Понятие алгоритма и программы. Способы записи алгоритмов.
2. Одномерные массивы. Понятия: массив, структуры данных, одномерный, двумерный массив. Объявление. Пример описания одномерного массива.
3. Задача: Задан одномерный массив A[1..20]. Найти сумму максимального и минимального элементов.

Зав. кафедрой Эльканова Л.М.

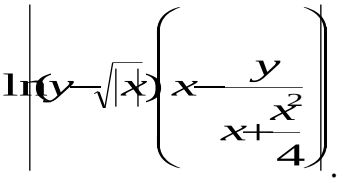
Задачи к экзамену

ТЕМА: АЛГОРИТМЫ ЛИНЕЙНОЙ СТРУКТУРЫ

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

Z=  .

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

Z=  .

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

Z=  .

ТЕМА: АЛГОРИТМЫ РАЗВЕТВЛЯЮЩЕЙСЯ СТРУКТУРЫ

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

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

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

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

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

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

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)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×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 журнал и статей. Выдавать информацию о публикациях, удовлетворяющих тому или иному критерию, например, изданных с 2000 года.

ТЕМА: МНОЖЕСТВА.

41. Дана матрица размера M×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 x 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 10x20 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 Алгоритмизация и программирования

1. Основные элементы программирования алгоритма.
2. Свойства алгоритмов. Виды алгоритмов.
3. Алгоритмизация и программирование, языки программирования высокого уровня.
4. Назначение алгоритмического языка PASCAL.
5. Основные символы языка.
6. Простейшие конструкции.
7. Структура программного модуля.
8. Классификация операторов
9. Операторы языка Паскаль.
10. Организация программ линейной структуры.
11. Оператор перехода. Условный оператор.
12. Организация программ разветвляющейся структуры.
13. Оператор выбора.
14. Операторы цикла.
15. Циклы с заданным и неявным числом повторений.
16. Одномерные массивы.
17. Вычисление суммы и произведения.
18. Нахождение наибольшего и наименьшего значений.
19. Вложенные циклы.
20. Двумерные массивы.
21. Оформление подпрограмм и обращение к ним.
22. Подпрограмма-функция. Подпрограмма-процедура.
23. Переменные типы данных.
24. Основные понятия и средства компьютерной графики в Паскале.
25. Логика в информатике.
26. Хакер. Защита информации. Информационная безопасность. Блок-схема.

Задания к контрольной работе по дисциплине Алгоритмизация и программирования

Вариант 1

1. Основные элементы программирования
2. Программные средства
3. **Задание № 1**

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

Z= 

Вариант 2

1. Требования, предъявляемые к программе
2. Нисходящее проектирование
3. **Задание № 2**

Вычислить количество и сумму отрицательных элементов массива X(5,4).

Вариант 3

1. Анализ и проектирование — разработка комплекса алгоритмов. Кодирование и компиляция. Тестирование и отладка.
2. Восходящее проектирование
3. **Задание № 3**

В целочисленном массиве A (30) найти наименьший из элементов с нечетными номерами.

Вариант 4

1. Назначение алгоритмического языка PASCAL.
2. Функциональное программирование
3. **Задание № 4**

Найти произведение элементов массива , индексы которых кратны 3.

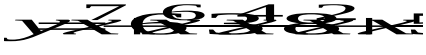
Вариант 5

1. Pascal
2. Переменные типы данных.
3. **Задание № 5**

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

Вариант 6

1. Игра в планирование
2. Хакер. Защита информации. Информационная безопасность. Блок-схема.
3. **Задание № 6**

Составить блок-схему и программу для вычисления значения многочлена  в точке x=0.5.

Вариант 7

1. Планирование итераций
2. Логика в информатике.
3. **Задание № 7**

   Разработайте приложение для определения общей стоимости покупки (в рублях и долларах). Название продуктов выбираются из списка, цены, количество, курс доллара вводит пользователь.

Вариант 8

1. Организация программ разветвляющейся структуры.
2. Характеристики экспертной системы.
3. **Задание № 8**

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

Тестовые задания
по дисциплине Алгоритмизация и программирования

Формируемая компетенция ОПК-6

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. **В каком файле может храниться рисунок?**
TEST.EXE;
ZADAN.TXT;
COMMAND.COM;
CREML.BMP.
6. **Файлом называется:**
набор данных для решения задачи;
поименованная область на диске или другом машинном носителе;
программа на языке программирования для решения задачи;
нет верного ответа.
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. Редактирование текста представляет собой:
процедуру сохранения текста на диске в виде текстового файла
процедуру считывания с внешнего запоминающего устройства ранее созданного текста
процесс внесения изменений в имеющийся текст
процесс передачи текстовой информации по компьютерной сети

25. Какие функции выполняет операционная система?
обеспечение организации и хранения файлов
подключения устройств ввода/вывода
организация обмена данными между компьютером и различными периферийными устройствами
организация диалога с пользователем, управления аппаратурой и ресурсами компьютера

26. В состав ОС не входит ...
BIOS
программа-загрузчик
драйверы
ядро ОС

27. Графическим редактором называется программа, предназначенная для ...
создания графического образа текста
редактирования вида и начертания шрифта
работы с графическим изображением
построения диаграмм.

28. Линейным называется алгоритм, в котором
_____

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

30. 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

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

32. В графических схемах алгоритмов стрелки направлений на линиях потоков
необходимо рисовать, если направление потока сверху вниз и слева направо
рисовать не нужно
необходимо рисовать, если направление потока снизу вверх и справа налево
можно рисовать или не рисовать

33. Инструкция для компьютера по выполнению задания, написанная на компьютерном языке **называется**
_____

34. Укажите последовательность действий выполняемых при сохранении готовой программы:

Нажать "Файл"

Нажать "Сохранить"

Выбрать "Сохранить как"

Выбрать место сохранения и имя файла

Темы рефератов

по дисциплине Алгоритмизация и программирования

1. Основные алгоритмические конструкции. Составление алгоритмов
2. Языки и системы программирования
3. Методы программирования
4. Типы данных. Объявление переменных и констант в программе
5. Выражения и функции языка Pascal
6. Программирование ветвлений в Pascal
7. Создание инсталляционного пакета
8. Программирование повторений
9. Отладка программ
10. Графические возможности языка Pascal
11. Перемещение объектов. Анимация
12. Многодокументный интерфейс. Создание меню пользователя
13. Модульное программирование
14. Создание панели инструментов
15. Подпрограммы. Процедуры и функции
16. Организация ввода-вывода данных
17. Современные системы разработки эффективных программ на языке высокого уровня
18. Формы представления алгоритмов: естественный язык, блок-схема, формальный язык.

5. Методические материалы, определяющие процедуры оценивания компетенции

5.1 Критерии оценивания качества устного ответа (коллоквиума)

- оценка **«отлично»** выставляется обучающемуся, 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 оценивания тестирования

При тестировании все верные ответы берутся за 100%.
90%-100% отлично
70%-90% хорошо
50%-70% удовлетворительно
менее 50% неудовлетворительно

5.3 Критерии оценивания результатов освоения дисциплины на зачете

- оценка **«отлично»** выставляется обучающемуся, если даны исчерпывающие и обоснованные ответы на все поставленные вопросы, правильно решены практические задания, при ответах выделялось главное, все теоретические положения умело увязывались с требованиями руководящих документов, ответы были четкими и краткими, а мысли излагались в логической последовательности, показано умение самостоятельно анализировать факты, события явления, процессы в их взаимосвязи и диалектическом развитии.
- оценка **«хорошо»** выставляется обучающемуся, если даны полные, достаточно обоснованные ответы на поставленные вопросы, правильно решены практические задания; при ответах не всегда выделялось главное, отдельные положения недостаточно увязывались с требованиями руководящих документов; ответы в основном были краткими, но не всегда четкими и по существу;
- оценка **«удовлетворительно»** выставляется обучающемуся, если даны в основном правильные ответы на все поставленные вопросы, но без должной глубины и обоснования; на уточняющие вопросы даны правильные ответы; при ответах не выделялось главное; ответы были многословными, нечеткими и без должной логической последовательности; на отдельные дополнительные вопросы не даны положительные ответы;
- оценка **«неудовлетворительно»** выставляется обучающемуся, если даны неправильные ответы на большинство вопросов; обучающийся путается в определениях и понятиях; не владеет практическими навыками решения задач.

5.4 Критерии оценивания 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.

5.5 Критерии оценивания практических работ

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

Критерии выполнения отчета на min балл Практическая 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.

5.6 Критерии оценки выполнения рефератов:

– оценка **«отлично»** выставляется обучающемуся, в случае, если теоретическое содержание темы изложено в полном объеме, сформированы необходимые практические навыки оформления материала в соответствии с требованиями, предъявляемыми к оформлению работ;

– оценка **«хорошо»** выставляется обучающемуся, в случае, если теоретическое содержание темы изложено в полном объеме, некоторые практические навыки оформления материала сформированы недостаточно;

– оценка **«удовлетворительно»** выставляется обучающемуся, в случае, если теоретическое содержание темы изложено не в полном объеме, но пробелы не носят существенного характера, некоторые практические навыки оформления материала сформированы недостаточно;

– оценка **«неудовлетворительно»** ставится в случае, если теоретическое содержание курса не освоено, необходимые практические навыки работы не сформированы, выполненные учебные задания содержат грубые ошибки.

5.7 Критерии оценки контрольных работ

- оценка **«отлично»** выставляется обучающемуся, если даны исчерпывающие и обоснованные ответы на все поставленные вопросы, правильно решены практические задания, при ответах выделялось главное, все теоретические положения умело увязывались с требованиями руководящих документов, ответы были четкими и краткими, а мысли излагались в логической последовательности, показано умение самостоятельно анализировать факты, события явления, процессы в их взаимосвязи и диалектическом развитии.

- оценка **«хорошо»** выставляется обучающемуся, если даны полные, достаточно обоснованные ответы на поставленные вопросы, правильно решены практические задания; при ответах не всегда выделялось главное, отдельные положения недостаточно увязывались с требованиями руководящих документов; ответы в основном были краткими, но не всегда четкими и по существу;

- оценка **«удовлетворительно»** выставляется обучающемуся, если даны в основном правильные ответы на все поставленные вопросы, но без должной глубины и обоснования; на уточняющие вопросы даны правильные ответы; при ответах не выделялось главное; ответы были многословными, нечеткими и без должной логической последовательности; на

отдельные дополнительные вопросы не даны положительные ответы;
- оценка **«неудовлетворительно»** выставляется обучающемуся, если даны неправильные ответы на большинство вопросов; обучающийся путается в определениях и понятиях; не владеет практическими навыками решения задач.