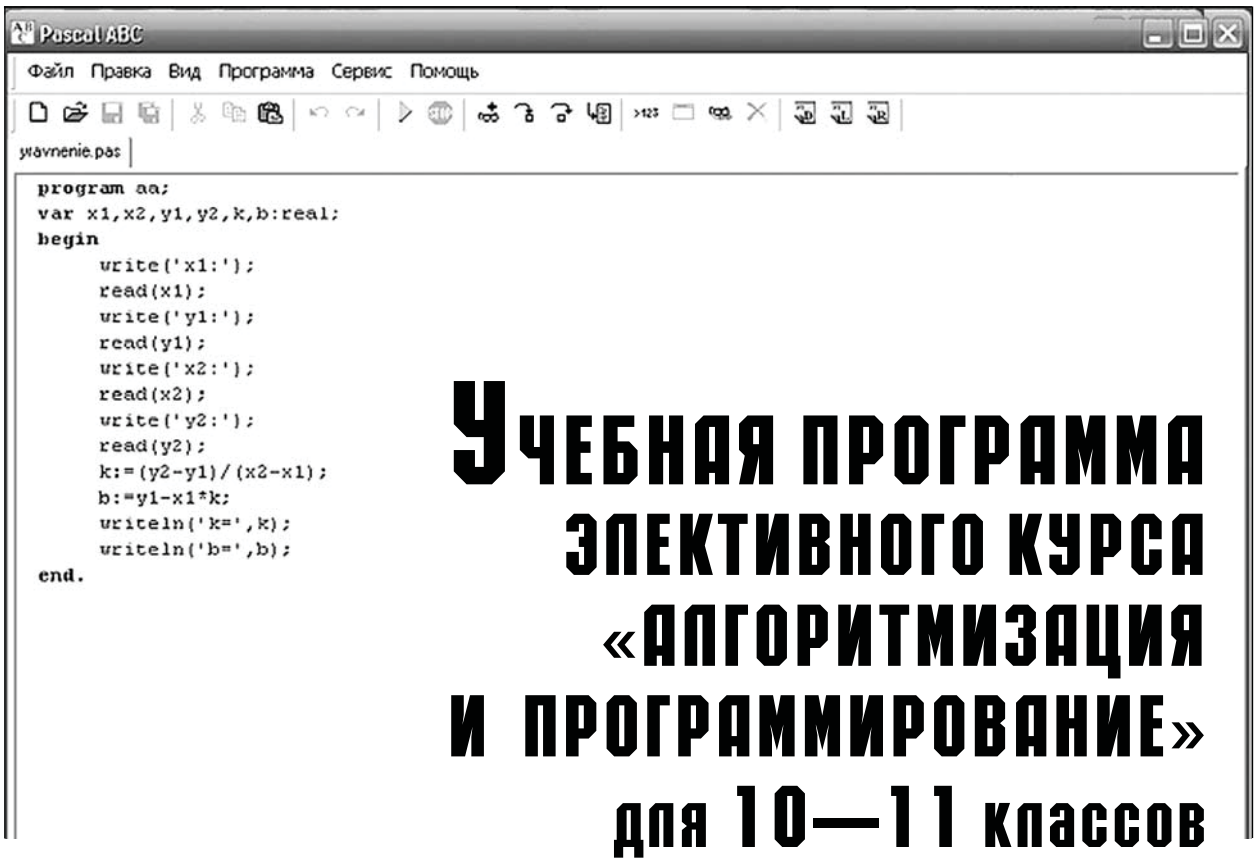
Учебная программа элективного курса «Алгоритмизация и программирование» для 10–11 классов

Государственное бюджетное образовательное учреждение
дополнительного профессионального образования
«НИЖЕГОРОДСКИЙ ИНСТИТУТ РАЗВИТИЯ ОБРАЗОВАНИЯ»



Учебная программа элективного курса «Алгоритмизация и программирование» для 10—11 классов

Нижний Новгород
Нижегородский институт развития образования
2020

УДК 004 (075.3)
ББК 74.263.2
У91

Составители:

***С. В. Кузнецова*,** учитель информатики высшей квалификационной категории МБОУ «Лицей № 7», г. Кстово;

***И. Е. Белоцерковская*,** кандидат физико-математических наук, доцент кафедры теории и методики обучения информатике ГБОУ ДПО «Нижегородский институт развития образования»;

***М. Ю. Втюрин*,** кандидат физико-математических наук, заведующий кафедрой теории и методики обучения информатике ГБОУ ДПО «Нижегородский институт развития образования»

Рекомендовано к изданию
научно-методическим экспертным советом
ГБОУ ДПО «Нижегородский институт развития образования»

У91 **Учебная** программа элективного курса «Алгоритмизация и программирование» для 10—11 классов / составители: С. В. Кузнецова, И. Е. Белоцерковская, М. Ю. Втюрин. — Нижний Новгород : Нижегородский институт развития образования, 2020. — 91 с.

ISBN 978-5-7565-0900-7

Учебная программа элективного курса «Алгоритмизация и программирование» содержит примерное учебно-тематическое планирование курса с характеристикой основных видов деятельности учащихся 10—11-х классов и всевозможных форм учебных занятий. В приложении приводится разнообразный дидактический материал.

Программа предназначена педагогам информатики, а также может быть полезна учащимся 10—11-х классов профильных образовательных организаций.

УДК 004 (075.3)
ББК 74.263.2

ISBN 978-5-7565-0900-7

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Программа элективного курса для 10—11-х классов является развитием элективного курса «Алгоритмизация и программирование» (экспертное заключение научно-методического экспертного совета ГБОУ ДПО «Нижегородский институт развития образования» № 151 от 22 октября 2013 года). С 2014 года вариантная часть учебного плана МБОУ «Лицей № 7» содержит данный элективный курс, весьма востребованный обучающимися лицея.

По данным, полученным за последние три года, можно констатировать следующее:

✓ средний балл на ЕГЭ по информатике среди обучающихся лицея составляет 84,2;

✓ 69,5 % учащихся поступили на технические факультеты, где необходимы глубокие знания по математике, физике, информатике.

Элективный курс «Алгоритмизация и программирование» имеет **техническую направленность** и **ориентирован** на обучающихся 10—11-х классов естественно-научного или технологического профилей.

По мере развития аппаратного и программного обеспечения и оснащения им школ курс информатики существенно изменился. Основное внимание стало уделяться современным информационным технологиям. Перечисленные тенденции отражены и в Федеральном государственном образовательном стандарте среднего (полного) общего образования по информатике.

Анализ существующих программ элективных курсов для учеников 10—11-х классов показал, что программы не в полной мере соответствуют образовательным потребностям в области программирования обучающихся естественно-научного и технологического профилей лицея. К изучению программирования на более высоком уровне стимулирует не только содержание контрольно-измерительных материалов ЕГЭ, но и олимпиады по информатике, значимость и уровень которых год от года возрастает.

Изучение основ алгоритмизации и программирования в курсе информатики осуществляется на структурных языках программирования, одним из которых и является *Pascal.*

С развитием новых информационных технологий, основанных на принципах объектно-ориентированного программирования, становится актуальным вопрос изучения объектно-ориентированного программирования в рамках профильных курсов. Изучение этой темы предусматривает:

✓ рассмотрение основных понятий объектно-ориентированного программирования;

✓ работу с большим количеством готовых компонентов и их собственное создание;

✓ ознакомление с базисом языка визуального проектирования необходимых для выпускников, решивших связать свою дальнейшую профессиональную деятельность с программированием.

Представленная программа элективного курса способствует решению *актуальной задачи — осуществление профессиональной ориентации обучающихся* через реализацию модели непрерывного образования «школа — вуз».

В качестве базового языка программирования выбран строгий и лаконичный *Pascal*, зарекомендовавший себя как первый язык при обучении программированию с последующим переходом к языку объектно-ориентированного программирования *Object Pascal*.

На данный момент существуют различные программы элективных курсов по алгоритмизации и программированию на *Pascal*. Особо отметим следующие программы элективных курсов:

✓ Ю. Л. Костюк, И. Л. Фукс «Основы разработки алгоритмов» (http://lbz.ru/books/232/5089/);

✓ В. И. Тишин «Информатика и математика», в 3 ч. Ч. 3 «Решение задач обработки массивов» (http://lbz.ru/books/232/7862/);

✓ Ю. Г. Просяник «Язык программирования Pascal в примерах» (https://www.google.ru/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCoQFjAA&url=http%3A%2F%2Fschoolcollection.edu.ru%2Fcatalog%2Fres% 2F92c9311d-0c0d).

Часть рассмотренных программ элективных курсов по алгоритмизации и программированию предполагают использование авторских учебно-методических пособий (Ю. Л. Костюк, В. И. Тишин). Некоторые из них опираются на другие языки программирования (например «Основы программирования на примере Visual Basic.NET» MS Corporation). Но несмотря на наличие элективных курсов по программированию, трудно найти такие, которые не только способствовали развитию навыков алгоритмизации, но рассматривали на должном уровне технологию программирования на языке *Pascal* и *Object Pascal*, учитывая индивидуальные способности обучающихся.

Для изучения языка программирования требуются особый способ и стиль мышления. Приобщение к этому стилю мышления начинается с переложения простейших математических примеров на программы, а затем следует математическое усложнение задач и соответствующее усложнение программ. При этом достаточно большое количество примеров позволяет все больше и активнее переключаться на методику программирования.

Спектр возможных математических примеров очень широк — от арифметики до сложных задач математического анализа, комбинаторики, теории вероятностей. Тесная связь математического содержания и программирования является **особенностью Программы**. Освоение программирования на основе математических задач позволит обучающимся получить в свое распоряжение широкие возможности программирования, которое является и будет оставаться основой информационных технологий.

Новизна элективного курса «Алгоритмизация и программирование» заключается в использование во время учебного процесса дифференцированных практических задач по программирование на языке *Pascal*, для учета индивидуальных отличий в учебных возможностях обучающихся и обеспечения каждого оптимальными условиями для формирования познавательной деятельности

в процессе учебной работы, а также увеличение разноплановой проектной деятельности:

✓ по теме «Основные алгоритмические структуры» — проекты «Численные методы. Вычисление площади криволинейной фигуры», «Численные методы. Вычисление суммы сходящегося ряда»;

✓ по теме «Вспомогательные алгоритмы. Процедуры и функции» — проект «Реализация алгоритма Евклида с использованием подпрограмм»;

✓ по теме «Массивы» — проект «Численные методы. Решение систем линейных уравнений»;

✓ индивидуальный учебный проект, созданный с использованием языка объектно-ориентированного программирования *Object Pascal*.

Цели Программы:

✓ формирование у выпускников школы основ научного мировоззрения;

✓ обеспечение преемственности между общим и профессиональным образованием за счет более эффективной подготовки выпускников к освоению программ профессионального высшего образования;

✓ развитие у учащихся алгоритмического мышления, способностей к формализации;

✓ создание условий для саморазвития и самовоспитания.

Задачи Программы:

✓ овладение умениями составлять и реализовывать программы на формальном языке, соответствующих заданному описанию;

✓ создавать программы на языке программирования по их описанию;

✓ формирование умения использовать стандартные алгоритмические конструкции при программировании;

✓ сформировать умения формально исполнять алгоритмы, записанные на естественных и алгоритмических языках, в том числе на языке программирования *Pascal*;

✓ сформировать умения анализировать текст программы с точки зрения соответствия записанного алгоритма поставленной задаче и изменять его в соответствии с заданием;

✓ реализовывать сложный алгоритм с использованием современных систем программирования;

✓ привить учащимся навыки, требуемые большинством видов современной деятельности — планирование и организация своей и коллективной деятельности, коммуникабельность;

✓ развивать способность к самообучению.

При реализации Программы предполагается использование материалов работ Т. Ю. Грацианової [3].

В зависимости от объективных и субъективных условий (общего уровня подготовки учащихся, их предпочтений и предпочтений родителей/законных представителей) нами были предложены следующие две модели организации обучения в рамках элективного курса «Алгоритмизация и программирование»:

1) последовательная — по одному часу в неделю в 10—11-х классах; всего — 70 часов;

2) углубленная — по два часа в неделю в 10—11-х классах; всего — 140 часов.

Таблица 1

Распределение часов по темам курса в рамках углубленного изучения

№ п/п	Тема	Количество часов		
		Всего	Класс	
			10	11
1	Элемента теории алгоритмов	7	7	0
2	Начальные сведения о языке программирования *Pascal.* Типы данных	7	7	
3	Основные алгоритмические структуры	35	35	
4	Вспомогательные алгоритмы. Процедуры и функции	10	10	
5	Массивы	28	8	20
6	Строковый тип данных	10		10
7	Файловый тип данных	11		11
8	Указатели и динамические структуры данных	8		8
9	Основы языка *Object Pascal.* Компилятор *Free Pascal.* Среда разработки *Lazarus*	13		13
10	Индивидуальный проект с использования языка *Object Pascal* в среде разработки *Lazarus*	6		6
11	Резерв	4	2	2
	Итого	140	70	70

Таблица 2

Распределение часов по темам курса в рамках последовательного изучения

№ п/п	Тема	Количество часов		
		Всего	Класс	
			10	11
1	Элемента теории алгоритмов	5	5	
2	Начальные сведения о языке программирования *Pascal.* Типы данных	6	6	
3	Основные алгоритмические структуры	18	18	
4	Вспомогательные алгоритмы. Процедуры и функции	5	5	
5	Массивы	10		10
6	Строковый тип данных	6		6
7	Файловый тип данных	7		7
8	Основы языка *Object Pascal.* Компилятор *Free Pascal.* Среда разработки *Lazarus*	7		7
9	Индивидуальный проект с использования языка *Object Pascal* в среде разработки *Lazarus*	4		4
10	Резерв	2	1	1
	Итого	70	35	35

ПРОГРАММА ЭЛЕКТИВНОГО КУРСА

Элементы теории алгоритмов. Понятие алгоритма и его свойства. Исполнитель алгоритмов — назначение, среда исполнителя, система команд исполнителя, режимы работы. Формы представления алгоритмов — словесная, графическая. Программа как способ записи алгоритма. Типы алгоритмов — линейный, ветвящийся и циклические. Формализация понятия алгоритма. Построение алгоритма и практические вычисления. Выполнение заданного алгоритма, записанного на естественном языке.

Учащиеся должны знать:

✓ сущность понятия «алгоритм»;

✓ основные свойства алгоритма — дискретность, определенность, точность, массовость, результативность;

✓ способы записи алгоритмов — словесный и блок-схемы;

✓ основные алгоритмические конструкции — следование, ветвление, цикл;

✓ назначение языков программирования.

Учащиеся должны уметь:

✓ определять в поставленной задаче исходные данные и результаты;

✓ выбирать исполнителя для решения конкретной задачи по системе его команд;

✓ пользоваться языком блок-схем;

✓ составлять линейные, ветвящиеся и циклические алгоритмы;

✓ исполнять алгоритм, записанный на естественном языке;

✓ исполнять алгоритм для конкретного исполнителя с фиксированным набором команд;

✓ создавать линейный алгоритм для формального исполнителя.

Компьютерный практикум:

✓ запись алгоритма на естественном языке;

✓ составление блок-схемы по заданному алгоритму;

✓ работа в средах учебных исполнителей («Чертежник», «Робот» и другое);

✓ практические вычисления по заданному алгоритму.

Начальные сведения о языке программирования Pascal. Типы данных. Начальные сведения о языке программирования. Понятие транслятора — «интерпретатора», «компилятора». Программа как способ записи алгоритма. Интегрированная инструментальная оболочка Турбо-Паскаля (ИИО ТП). Структура программы на языке *Pascal*. Алфавит и классификация данных языка Турбо-Паскаль — алфавит, идентификатор, константы, переменные, арифметические выражения, логические выражения, стандартные функции. Оператор (процедура) вывода данных. Оператор (процедура) вывода.

Текстовая графика. Целый тип данных. Арифметические выражения. Оператор присваивания. Вещественный тип данных. Запись математических выражений. Процедуры перечисляемого аргумента. Логические операции, применяемые к переменным логического типа. Логический тип данных. Вычислительные алгоритмы. Форматированный вывод данных. Переменные ограниченного типа. Приведение типов переменных. Абсолютные переменные.

Учащиеся должны знать:

✓ назначение языков программирования;

✓ назначение транслятора;

✓ разницу между компилятором и интерпретатором;

✓ основные понятия языка — алфавит (буквы, цифры, специальные символы), оператор;

✓ назначение и синтаксис идентификатора;

✓ структуру программы на языке *Pascal*;

✓ определение программы как способа записи алгоритма;

✓ определение типа, определение переменных этого типа, констант;

✓ стандартные функции;

✓ процедуры перечисляемого аргумента;

✓ логические операции, применяемые к переменным логического типа;

✓ синтаксис изучаемых операторов;

✓ возможности создания изображений, состоящих из текстовой графики;

✓ выполнение каждого оператора.

Учащиеся должны уметь:

✓ записывать и читать арифметические выражения, переходя от математической формы записи к записи на языке программирования и обратно;

✓ объяснять различие между транслятором, интерпретатором и компилятором;

✓ работать в ИИО ТП;

✓ пояснять необходимость разнообразия типов данных в языке программирования;

✓ выбрать необходимый тип;

✓ создавать изображения, используя текстовую графику;

✓ пояснять необходимость разнообразия типов данных в языке программирования;

✓ выбирать необходимый тип;

✓ реализовать несложные алгоритмы с использованием операторов языка *Pascal*.

Компьютерный практикум:

✓ работа с учебным исполнителем алгоритмов;

✓ отработка процедуры вывода на экран;

✓ отработка процедуры ввода данных с клавиатуры на примере диалоговых приложений;

✓ отработка задач на вычисление логических выражений;

✓ отработка задач на логический тип данных;

✓ создание изображений, состоящих из текстовой графики.

Основные алгоритмические структуры. Линейный алгоритм. Программы, реализующие линейные алгоритмы с целочисленной арифметикой. Полное ветвление. Неполное ветвление. Вложенное ветвление. Разветвляющие программы. Ветвление по ряду условий (оператор *case*). Понятие циклического алгоритма. Структура цикла с параметром. Вывод в табличном виде с использованием псевдографики — значение кодов *ASC*II альтернативной таблицы для построения линий, таблиц и рамок. Теоретико-числовые алгоритмы — кратность чисел (параметр цикла — число), целочисленная арифметика (параметр цикла — счетчик), последовательность чисел, нахождения суммы ряда, трассировка. Структура цикла с предусловием. Теоретико-числовые алгоритмы — совершенные числа, кратность чисел, целочисленная арифметика, приближенное вычисление суммы бесконечного ряда, возведение числа в указанную целую степень. Структура цикла с постусловием. Вложенные циклы. Относительность выбора операторов *while* и *repeat*.

Учащиеся должны знать:

✓ структуру полного и неполного ветвления;

✓ структуру вложенного ветвления;

✓ структуру оператора выбора;

✓ алгоритм вычисления значения функции;

✓ структуру цикла с заданным числом повторений (цикл с параметром);

✓ алгоритм вывода информации в табличном виде;

✓ алгоритм нахождения цифры заданного числа;

✓ алгоритм вывода формул последовательности чисел;
✓ алгоритмы накопления (подсчет сумм, произведений, организация счетчиков);
✓ структуру цикла с предусловием;
✓ цикл не выполняется ни разу, если с самого начала значение логического выражения ложно;
✓ алгоритм нахождения цифры заданного числа;
✓ алгоритм удаления цифры из числа;
✓ алгоритм вставки цифры в число;
✓ понятие вложенного цикла.

Учащиеся должны уметь:
✓ реализовать несложные линейные алгоритмы с использованием операторов языка *Pascal*;
✓ составлять алгоритмы и реализовывать на компьютере с использованием полного и неполного ветвления;
✓ составлять алгоритмы и реализовывать на компьютере с использованием вложенного ветвления;
✓ составлять алгоритмы и реализовывать на компьютере с использованием оператора выбора;
✓ выводить таблицу с информацией, используя псевдографику;
✓ находить количество, сумму и произведение чисел по заданным критериям;
✓ решать задачи на кратность чисел (параметр цикла — число), на целочисленную арифметику (параметр цикла — счетчик);
✓ решать задачи на вывод формулы последовательности чисел;
✓ находить количество, сумму и произведение цифр числа по заданным критериям;
✓ составлять алгоритмы и реализовывать программы на компьютере удаления цифры из числа;
✓ составлять алгоритмы и реализовывать программы на компьютере вставки цифры в число;
✓ составлять алгоритм с использованием вложенного цикла.

Компьютерный практикум:
✓ отработка линейных программ, используя математические задачи;
✓ отработка задач
с использованием полного и неполного ветвления;
на вложенное ветвление;
на вывод информации в табличном виде;
на составление и вывод в табличном виде таблиц истинности логических выражений;
на кратность чисел;
на вычисление суммы ряда;
на возведения числа в указанную целую степень;
нахождения цифры заданного числа;
на разложение числа в произведение простых множителей;
удаления цифры из числа;

на нахождение цифр (их суммы, произведения) данного числа после перевода его в систему счисления с основанием *p*;
вставки цифры в число.

Вспомогательные алгоритмы. Процедуры и функции о назначении процедур и функций. Общая структура процедур и функций; параметры процедур и функций (параметры-значения и параметры-переменные). Взаимодействие основной программы с процедурами и функциями. Формальные и фактические параметры. Область действия переменных. Локализация имен. Рекурсивные алгоритмы.

Учащиеся должны знать:
✓ назначение процедур и функций;
✓ структуру процедур и функций;
✓ связь формальных и фактических параметров;
✓ отличие параметров-значений от параметров-переменных.

Учащиеся должны уметь:
✓ организовывать программы с использованием процедур и функций;
✓ использовать рекурсивные алгоритмы при решении задач.

Компьютерный практикум — отработка задач на
✓ вычисление степени числа;
✓ вычисление факториала числа;
✓ вычисление суммы ряда;
✓ рекурсию.

Массивы. Символьный тип данных — константы, переменные, функции. Перечисляемый тип данных — определение типа, константы, переменные, операции. Ограниченный тип данных.

Простейшая структура данных — массив. Понятие массива, задание типа. Использование числовых массивов в задачах. Простейшие операции над одномерными массивами. Упорядочивание (сортировка) массивов — алгоритм сортировки выбором, обменом. Генератор случайных чисел. Использование случайных значений для задания значений массива.

Учащиеся должны знать:
✓ определение типа, определение переменных этого типа, констант;
✓ стандартные функции;
✓ синтаксис изучаемых операторов;
✓ выполнение каждого оператора.

Учащиеся должны уметь:
✓ пояснять необходимость разнообразия типов данных в языке программирования;
✓ выбрать необходимый тип;
✓ реализовать несложные алгоритмы с использованием изученных операторов языка *Pascal*.

Компьютерный практикум — отработка задач:
✓ на вычисление суммы элементов массива;
✓ на целочисленную арифметику;

✓ с использованием алгоритма нахождения среднего значения, максимального и минимального значения элементов одномерного массива;

✓ с использованием алгоритма нахождения и замены элементов одномерного массива по заданным критериям;

✓ с использованием алгоритма вставки, удаления элементов одномерного массива;

✓ с использованием алгоритмов сортировки элементов одномерного массива;

✓ с использованием выше перечисленных алгоритмов над двумерным массивом.

Строковый тип данных. Строковые константы и переменные. Операции над строками — присваивания, сцепления и отношения.

Встроенные функции для обработки строк. Программирование алгоритмов с использованием строк.

Учащиеся должны знать:

✓ понятия строковых констант и переменных;

✓ основные операции над строками: присваивания, сцепления и отношения;

✓ встроенные функции для обработки строк;

✓ алгоритм создания массива из слов предложения.

Учащиеся должны уметь:

✓ использовать основные операции над строками при решении конкретной задачи;

✓ использовать встроенные функции для обработки строк;

✓ использовать алгоритм создания массива из слов предложения при решении задач.

Компьютерный практикум — отработка задач с использованием

✓ встроенных функций для обработки строк;

✓ алгоритма создания массива из слов предложения.

Файловый тип данных. Типизированные файлы, стандартные процедуры работы с типизированными файлами. Основные операции с файлами — чтение из файла, запись в файл, добавление данных в файл, запись и чтение файла, прямая выборка элементов из файла.

Текстовые файлы. Процедуры работы с текстовыми файлами. Программы обработки текстовых файлов.

Учащиеся должны знать:

✓ возможности использования данных типа файл;

✓ знать стандартные процедуры работы с типизированными файлами, с текстовыми файлами;

✓ отличие текстового файла от типизированного, элементами которого являются данные символьного типа.

Учащиеся должны уметь:

✓ объяснить необходимость использования данных типа файл при решении конкретной задачи;

✓ читать элементы файла;

✓ записывать данные в файл;
✓ понимать разницу между входным и выходным файлами.

Компьютерный практикум — отработка задач с использованием
✓ типизированных файлов;
✓ текстовых файлов.

Указатели и динамические структуры данных. Введение — о статических и динамических переменных. Ссылки и указатели. Линейные списки (основные операции) — создание и просмотр списка; удаление элемента из списка.

Стек — введение понятия, основные операции со стеком.

Очереди — введение понятия, основные операции над очередью.

Учащиеся должны знать:
✓ понятия списка, стека, очереди, дерева;
✓ основные операции над списками, со стеками, над очередью.

Учащиеся должны уметь — при составлении алгоритма заданной задачи использовать статические и динамические переменные.

Компьютерный практикум — отработка задач с использованием
✓ стека, очереди, списка;
✓ алгоритмов динамических и статистических структур.

Основы языка Object Pascal. Компилятор Free Pascal. Среда разработки Lazarus. Среда визуального программирования *Lazarus*. Интегрированная среда разработки приложений. Окна, формы и объекты, процедуры и функции. Создание интерфейса проекта. Графические методы и процедуры.

Учащиеся должны знать:
✓ структуру программы, элементы языка, модуль и основные его компоненты;
✓ основные понятия объектно-ориентированного языка;
✓ графические методы и процедуры.

Учащиеся должны уметь:
✓ добавлять объекты и подключать процедуру событий к объектам;
✓ использовать объекты для решения содержательных задач;
✓ использовать компонент *StringGrid* и поле *Memo* для работы с массивами.
✓ создавать интерфейс проекта;
✓ использовать свойство объекта *Canvas* для рисования графики.

Компьютерный практикум — проект «Дорога». Рассматривается модель, включающая объект — дорогу и несколько объектов — автомашин.

Индивидуальный проект с использованием языка Object Pascal в среде разработки Lazarus

Постановка задачи, разработка алгоритма, а затем программы в системе программирования *TP*, решающей поставленную задачу. Оформление документации. Защита проектной работы.

Учащиеся должны знать:
✓ типы информационных моделей;
✓ основные этапы разработки и исследования моделей на компьютере;

✓ способы пошаговой детализации программ;
✓ методы отладки программ.

Учащиеся должны уметь:
✓ исследовать предложенные модели;
✓ оптимизировать модели;
✓ строить информационные модели управления объектами;
✓ использовать изученный аппарат языка программирования для решения поставленной задачи;
✓ пользоваться текстовым редактором, выполнять основные операции при оформлении документации проектной работы.

Компьютерный практикум (примерные темы проектов):
✓ «Калькулятор»;
✓ «Заставка Часы»;
✓ «Вычисление арифметических выражений»;
✓ «Словарь»;
✓ «Заполнение готовых форм с помощью информации из базы данных».

РЕКОМЕНДУЕМОЕ ПОУРОЧНОЕ ПЛАНИРОВАНИЕ КУРСА

Таблица 3

10 класс
(при углубленной модели обучения)

№ п/п	Тема урока
	Элементы теории алгоритмов (7 часов)
1	Введение в курс. Алгоритм и его свойства
2	Исполнитель алгоритмов — назначение, среда исполнителя, система команд исполнителя, режимы работы. Формы представления алгоритмов. Блок-схема
3	Типы алгоритмов
4	Построение алгоритма и практические вычисления
5	Построение алгоритма и практические вычисления
6	Выполнение заданного алгоритма, записанного на естественном языке
7	*Контрольная работа по теме «Элементы теории алгоритмов»*
	Начальные сведения о языке программирования Pascal. Типы данных (7 часов)
8	Начальные сведения о языке программирования. Понятие транслятора (интерпретатора, компилятора). Введение в Паскаль. Из истории. Общая структура программ в Паскале. Интегрированная инструментальная оболочка. Создание программ в ИИО ТП

Продолжение табл. 3

№ п/п	Тема урока
9	Хранение данных. Константы. Переменные. Правила организации идентификаторов. Выражения (арифметические, логические и стандартные функции). Оператор присваивания
10	Решение задач на запись математических выражений на языке Паскаль и расстановка приоритетов выполнения операций. Оператор присваивания
11	Целочисленные типы переменных. Вещественные типы переменных. Логический тип данных
12	Процедуры перечисляемого аргумента. Логические операции, применяемые к переменным логического типа. Решение задач на вычисление логических выражений
13	Переменные ограниченного типа. Приведение типов переменных. Абсолютные переменные. Решение задач
14	*Контрольная работа по теме «Типы данных»*
	Основные алгоритмические структуры (35 часов)
15	Линейная программа. Решение задач. Практическая работа № 1
16	Линейная программа. Решение задач. Практическая работа № 1
17	Разветвляющийся алгоритм. Полное ветвление. Неполное ветвление
18	Решение задач с использованием полного и неполного ветвления. Практическая работа № 2
19	Оператор выбора case. Решение задач с использованием оператора выбора. Практическая работа № 2
20	Вложенное ветвление
21	Решение задач на вложенное ветвление
22	Проект «Программирование обучающих тестов»
23	Самостоятельная работа по теме «Линейные и разветвляющиеся программы» (отладка программ)
24	Циклы. Структура цикла с параметром
25	Нахождение суммы и количество чисел в заданном ряду
26	Решение задач на кратность чисел (параметр цикла — число)
27	Решение задач на кратность чисел (параметр цикла — счетчик)
28	Работа с окнами. Метод пошагового выполнения программ
29	Самостоятельная работа по теме «Цикл с параметром»
30	Последовательность чисел. Алгоритм вывода формулы последовательности чисел
31	Решение задач вычисление суммы элементов сходящегося ряда

Продолжение табл. 3

№ п/п	Тема урока
32	Решение задач на составление и вывод в табличном виде таблиц истинности логических выражений
33	Самостоятельная работа на последовательность чисел и составление таблиц истинности логических выражений
34	Цикл с предусловием. Общая структура. Синтаксис оператора
35	Цикл с предусловием. Нахождение количества цифр числа. Нахождение наименьшей (наибольшей) цифры числа и ее позиции
36	Алгоритм выделения цифры из заданного числа
37	Решение задач на нахождение цифр (их суммы, произведения) данного числа после перевода его в систему счисления с основанием *p*
38	Алгоритм получения из заданного числа палиндрома
39	Самостоятельная работа на цикл с предусловием
40	Цикл с постусловием. Общая структура. Синтаксис оператора
41	Цикл с постусловием. Решение задач на возведения числа в указанную целую степень
42	Решение задач на разложение числа в произведение простых множителей. Алгоритм Евклида
43	Самостоятельная работа на цикл с постусловием
44	Вложенные циклы. Относительность выбора операторов *while* и *repeat*
45	Вложенные циклы. Относительность выбора операторов *while* и *repeat*
46	Проект «Численные методы. Вычисление площади криволинейной фигуры»
47	Проект «Численные методы. Вычисление суммы сходящегося ряда»
48	Практическая работа № 3
49	Практическая работа № 3
	Вспомогательные алгоритмы. Процедуры и функции (10 часов)
50	Вспомогательные алгоритмы. Подпрограммы
51	Процедуры. Описание процедуры. Формальные и фактические параметры. Глобальные и локальные переменные. Вычисление суммы ряда
52	Решение задач на вычисление степени числа и замены местами целых значений переменных
53	Функции. Описание функций в программе. Функция для вычисления факториала числа
54	Решение задач с использованием функций
55	Рекурсивные алгоритмы. Решение задач с рекурсивной формулировкой и задач, из постановки которых можно извлечь рекурсию
56	Проект «Реализация алгоритма Евклида с использованием подпрограмм»
57	Решение задач, которые можно решить как частный случай обобщенной, и задач, в которых можно использовать характеристику или свойство функции. Практическая работа № 4

Окончание табл. 3

№ п/п	Тема урока
58	*Контрольная работа по теме «Вспомогательные алгоритмы»*
59	*Контрольная работа по теме «Вспомогательные алгоритмы»*
	Массивы (8 часов)
60	Одномерный массив. Ввод и вывод элементов массива при помощи процедуры
61	Одномерный массив. Нахождение суммы и среднего значения элементов массива
62	Одномерный массив. Нахождение элементов по заданным критериям. Нахождение номеров элементов, обладающих заданным свойством
63	Одномерный массив. Изменение значений некоторых элементов массива
64	Одномерный массив. Решение задач на нахождение элементов по заданным критериям и замену. Практическая работа № 5
65	Одномерный массив. Создание нового массива. Практическая работа № 5
66	Работа с несколькими массивами
67	*Контрольная работа по теме «Одномерный массив» (составление блок-схемы)*
68	*Контрольная работа по теме «Одномерный массив» (написание программы)*
69	Резерв
70	Резерв

Таблица 4

10 класс
(при последовательной модели обучения)

№ п/п	Тема урока
	Элементы теории алгоритмов (5 часов)
1	Введение в курс. Алгоритм и его свойства. Исполнитель алгоритмов: назначение, среда исполнителя, система команд исполнителя, режимы работы
2	Формы представления алгоритмов. Блок-схема. Типы алгоритмов
3	Построение алгоритма и практические вычисления
4	Выполнение заданного алгоритма, записанного на естественном языке
5	*Контрольная работа по теме «Элементы теории алгоритмов»*

Продолжение табл. 4

№ п/п	Тема урока
	Начальные сведения о языке программирования Pascal *Типы данных* *(6 часов)*
6	Начальные сведения о языке программирования. Понятие транслятора (интерпретатора, компилятора). Введение в Паскаль. Из истории. Общая структура программ в Паскале. Интегрированная инструментальная оболочка. Создание программ в ИИО ТП
7	Хранение данных. Константы. Переменные. Правила организации идентификаторов. Выражения (арифметические, логические и стандартные функции). Оператор присваивания
8	Решение задач на запись математических выражений на языке Паскаль и расстановка приоритетов выполнения операций. Оператор присваивания
9	Целочисленные типы переменных. Вещественные типы переменных. Логический тип данных
10	Процедуры перечисляемого аргумента. Логические операции, применяемые к переменным логического типа. Решение задач на вычисление логических выражений
11	*Контрольная работа по теме «Типы данных»*
	Основные алгоритмические структуры *(18 часов)*
12	Линейная программа. Решение задач. Практическая работа № 1
13	Разветвляющийся алгоритм. Полное ветвление. Неполное ветвление
14	Решение задач с использованием полного и неполного ветвления. Практическая работа № 2
15	Самостоятельная работа по теме «Линейные и разветвляющиеся программы» (составление блок-схемы, написание программ)
16	Циклы. Структура цикла с параметром
17	Нахождение суммы и количество чисел в заданном ряду
18	Решение задач на кратность чисел
19	Самостоятельная работа по теме «Цикл с параметром». Последовательность чисел. Алгоритм вывода формулы последовательности чисел
20	Решение задач вычисление суммы элементов сходящегося ряда
21	Цикл с предусловием. Общая структура. Синтаксис оператора
22	Цикл с предусловием. Нахождение количества цифр числа. Нахождение наименьшей (наибольшей) цифры числа и ее позиции
23	Алгоритм выделения цифры из заданного числа
24	Цикл с постусловием. Общая структура. Синтаксис оператора

Окончание табл. 4

№ п/п	Тема урока
25	Цикл с постусловием. Решение задач на возведения числа в указанную целую степень
26	Самостоятельная работа на цикл с предусловием и цикл с постусловием. Практическая работа № 3
27	Вложенные циклы. Практическая работа № 3
28	Решение задач по теме «Основные алгоритмические структуры»
29	*Контрольная работа по теме «Основные алгоритмические структуры»*
	Вспомогательные алгоритмы. Процедуры и функции (5 часов)
30	Процедуры. Описание процедуры. Функции. Описание функций в программе. Формальные и фактические параметры. Глобальные и локальные переменные
31	Вычисление суммы ряда. Решение задач на вычисление степени числа
32	Функция для вычисления факториала числа
33	Решение задач с использованием функций. Практическая работа № 4
34	*Контрольная работа по теме «Вспомогательные алгоритмы»*
35	Резерв

Таблица 5

11 класс
(при углубленной модели обучения)

№ п/п	Тема урока
	Массивы (20 часов)
1	Одномерный массив. Удаление одного элемента из одномерного массива
2	Одномерный массив. Удаление нескольких элементов из одномерного массива
3	Одномерный массив. Вставка одного элемента в одномерный массив
4	Одномерный массив. Вставка нескольких элементов в одномерный массив
5	Самостоятельная работа на удаление и вставку элементов
6	Одномерный массив. Сортировка по возрастанию (обменом, выбором)
7	Одномерный массив. Сортировка по убыванию. Решение задач
8	Одномерный массив. Сортировка. Удаление и вставка элементов. Решение задач
9	Одномерный массив. Сортировка. Удаление и вставка элементов. Решение задач

Продолжение табл. 5

№ п/п	Тема урока
10	Создание одномерного массива из средних значений по строчкам
11	Контрольная работа на сортировку, удаление и вставку элементов
12	Двумерные массивы. Описание двумерного массива. Способы заполнения. Работа с элементами
13	Двумерные массивы. Нахождение суммы элементов и среднего значения по строчкам
14	Двумерные массивы. Нахождение количества элементов с данным свойством
15	Двумерные массивы. Работа с несколькими массивами. Изменение значений некоторых элементов, обладающих заданным свойством
16	Заполнение двумерного массива по указанному правилу. Перестановка элементов массива
17	Вставка и удаление строк и столбцов двумерного массива
18	Проект «Численные методы. Решение систем линейных уравнений»
19	Практическая работа № 5
20	*Контрольная работа на двумерные массивы (написание программы)*
	Строковый тип данных (10 часов)
21	Строковый тип данных. Операции над строками
22	Операции над строками (присваивания, сцепления, отношения)
23	Стандартные процедуры и функции (удаление, вставка, копирование)
24	Стандартные процедуры и функции (длина строки, поиск подстроки)
25	Числа и строки
26	Программирование алгоритмов с использованием строк
27	Программирование алгоритмов с использованием строк
28	Практическая работа № 6
29	*Контрольная работа на обработку строк*
30	*Контрольная работа на обработку строк*
	Файловый тип данных (11 часов)
31	Файловый тип данных. Описание файлов
32	Операции для работы с файлами последовательного доступа
33	Программирование алгоритмов на работу с файлами (чтение из файла)
34	Программирование алгоритмов на работу с файлами (запись в файл)
35	Программирование алгоритмов на работу с типизированными файлами
36	Программирование алгоритмов на работу с типизированными файлами
37	Самостоятельная работа на типизированные файлы
38	Текстовые файлы
39	Обработка текстовых файлов

Окончание табл. 5

№ п/п	Тема урока
40	*Контрольная работа на работу с файлами*
41	*Контрольная работа на работу с файлами*
Основы языка Object Pascal. Компилятор Free Pascal. Среда разработки Lazarus (13 часов)	
42	Использование среды *Lazarus*. Анатомия проекта
43	Работа с компонентами
44	Основы кода
45	Символы и строки. Стандартные строковые функции и сообщения
46	Логические типы, конструкции и компонент
47	Числа
48	Подпрограммы
49	Циклы и переключатель *case*
50	Массивы простые, многомерные и динамические
51	Диалоги. Организация меню и панелей инструментов
52	Модули
53	Тестирование и отладка
54	*Проект «Телефонный справочник»*
Указатели и динамические структуры данных (8 часов)	
55	Введение: о статических и динамических переменных
56	Ссылки и указатели
57	Линейные списки
58	Стек
59	Очередь
60	Программирование алгоритмов с использованием динамических структур данных
61	Программирование алгоритмов с использованием динамических структур данных
62	Самостоятельная работа по теме «Динамические структуры данных»
Индивидуальный проект с использованием языка Object Pascal в среде разработки Lazarus (6 часов)	
63	Индивидуальный проект с использованием среды *Lazarus*
64	Индивидуальный проект с использованием среды *Lazarus*
65	Индивидуальный проект с использованием среды *Lazarus*
66	Индивидуальный проект с использованием среды *Lazarus*
67	Индивидуальный проект с использованием среды *Lazarus*
68	*Защита проекта*
Резерв (2 часа)	

Таблица 6

11 класс
(при последовательной модели обучения)

№ п/п	Тема урока
	Массивы (10 часов)
1	Одномерный массив. Ввод и вывод элементов массива при помощи процедуры
2	Одномерный массив. Нахождение суммы и среднего значения элементов массива
3	Одномерный массив. Нахождение элементов по заданным критериям. Нахождение номеров элементов, обладающих заданным свойством
4	Одномерный массив. Удаление и вставка элемента из одномерного массива
5	Одномерный массив. Сортировка по возрастанию (обменом, выбором)
6	Одномерный массив. Сортировка. Удаление и вставка элементов. Решение задач
7	Самостоятельная работа на сортировку, удаление и вставку элементов
8	Двумерные массивы. Описание двумерного массива. Способы заполнения. Работа с элементами
9	Двумерные массивы. Нахождение суммы элементов и среднего значения по строчкам
10	*Практическая работа № 5*
	Строковый тип данных (6 часов)
11	Строковый тип данных. Операции над строками
12	Операции над строками (присваивания, сцепления, отношения)
13	Стандартные процедуры и функции (удаление, вставка, копирование)
14	Стандартные процедуры и функции (длина строки, поиск подстроки)
15	Практическая работа № 6
16	*Контрольная работа на обработку строк*
	Файловый тип данных (7 часов)
17	Файловый тип данных. Описание файлов
18	Операции для работы с файлами последовательного доступа
19	Программирование алгоритмов на работу с файлами (чтение из файла)
20	Программирование алгоритмов на работу с файлами (запись в файл)
21	Текстовые файлы
22	Обработка текстовых файлов
23	*Контрольная работа на работу с файлами*

Окончание табл. 6

№ п/п	Тема урока
Основы языка Object Pascal Компилятор Free Pascal. Среда разработки Lazarus (7 часов)	
24	Использование среды *Lazarus*. Анатомия проекта
25	Работа с компонентами
26	Основы кода
27	Символы и строки. Числа
28	Циклы и переключатель *case*
29	Массивы простые, многомерные и динамические
30	*Проект «Экранная заставка»*
Проектная работа (4 часов)	
31	Индивидуальный проект с использованием среды *Lazarus*
32	Индивидуальный проект с использованием среды *Lazarus*
33	Индивидуальный проект с использованием среды *Lazarus*
34	*Защита проекта*
Резерв (1 час)	

Практическая работа № 1

Цели работы:

✓ выработать практические навыки использования системы программирования *Pascal;*

✓ научиться создавать, вводить, отправлять на выполнение и исправлять простейшие программы на языке *Pascal*;

✓ познакомиться с диагностическими сообщениями компилятора об ошибках на примере программ, реализующих линейные алгоритмы.

Общие сведения. Линейным называется алгоритм, в котором результат получается путем однократного выполнения заданной последовательности действий при любых значениях исходных данных. Операторы программы выполняются последовательно один за другим в соответствии с их расположением в программе.

Перед выполнением работы необходимо ознакомиться с теоретическим материалом по темам «Описание языка Паскаль», «Простые операторы. Ввод/вывод данных».

Пример. Определить расстояние на плоскости между двумя точками с заданными координатами $M_1\ (x_1,\ y_1)$ и $M_2\ (x_2,\ y_2)$.

Решение задачи. В этом примере проведем полный разбор решения задачи.

Математическая модель: расстояние на плоскости между двумя точками $M_1\ (x_1,\ y_1)$ и $M_2\ (x_2,\ y_2)$ высчитывается по формуле:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.$$

Составим блок-схему алгоритма, а затем уточним содержимое блоков «Вычисление расстояния» и «Вывод расстояния» (рис.1):

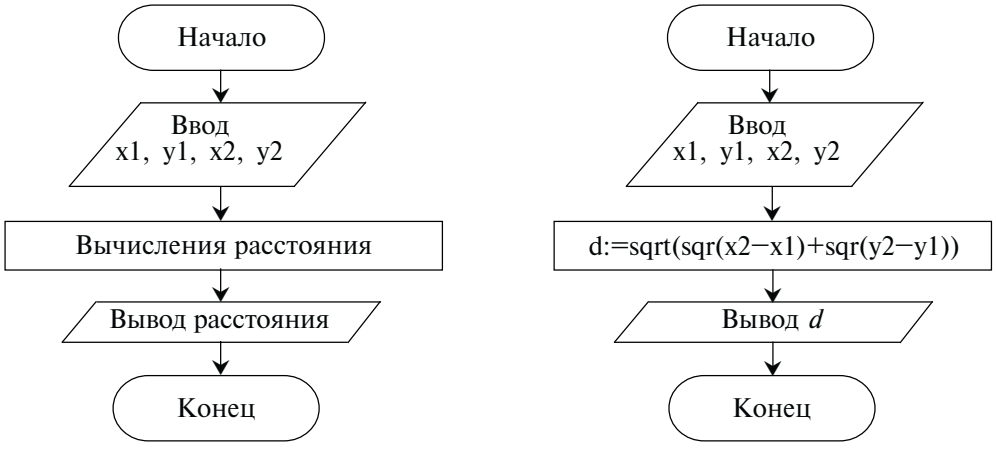


Рис. 1. Блок-схема алгоритма

Дальнейшая детализация не требуется. Переводим блок-схему на язык *Pascal*, доработав программу, чтобы улучшить ее интерфейс:

```
program example1;
var x1, x2, y1, y2: Integer;
 d:Real;
begin
 Writeln (эта программа вычисляет расстояние между двумя точками на плос-
кости');
 Writeln («введите координаты двух точек:»);
 Write('x1= '); Readln(x1);
 Write('y1= '); Readln(y1);
 Write('x2= '); Readln(x2);
 Write('y2 = '); Readln(y2);
 d:=sqrt(sqr(x2-x1)+sqr(y2-y1));
 Writeln('d= ',d:6:2);
 Writeln(нажмите «enter» для завершения работы программы);
 Readln;
end.
```

Варианты заданий

1. Даны x, y. Составить программу вычисления значения выражения:

a) $\frac{|x|-|y|}{1+|xy|}$; **b)** $\frac{\sqrt{x^2+y^2}}{xy}$; **c)** $\frac{x-y}{|x|-|y|}$; **d)** $\frac{\sqrt{|x|+|y|}}{\sqrt{x^2+1}}$.

2. Составить программу для решения следующих задач:

a) дана длина ребра куба. Найти объем куба и площадь его боковой поверхности;

b) известна длина окружности. Найти площадь круга, ограниченного этой окружностью;

c) вычислить высоту треугольника, опущенную на сторону a, по известным значениям длин его сторон a, b, c;

d) по данным сторонам прямоугольника вычислить его периметр, площадь и длину диагонали;

3. Вывести значение *true*, если приведенное высказывание для предложенных исходных данных является истинным, и значение *false* в противном случае (все числа, для которых не указано иное, являются действительными):

a) данное число x принадлежит отрезку $[-a, a]$;

b) данное число x не принадлежит интервалу (a, b);

c) данное целое число x является нечетным;

d) данное число x является корнем уравнения: $ax^2 + bx + c = 0$;

Дополнительные задания

1. Ученик начал решать задачи данного урока программирования, когда электронные часы показывали h_1 часов и $\min_1$ минут, а закончил, когда было h_2 часов и $\min_2$ минут. Составьте программу, позволяющую определить, сколько времени (в часах и минутах) ученик решал эти задачи.

2. Дано действительное число *a*. Не пользуясь никакими другими операциями, кроме умножения, получить:

a) a_4 за две операции;
b) a_6 за три операции;
c) a_7 за четыре операции;
d) a_8 за три операции.

Контрольные вопросы

1. Каковы назначение и возможности системы программирования?
2. Как запустить программу на трансляцию и выполнение?
3. Как обозначается начало и конец программы?
4. Из каких разделов состоит программа на языке *Pascal*?
5. Как в языке *Pascal* осуществляется вывод на экран?
6. Для чего предназначен оператор присваивания?
7. Как вывести на экран значение переменной?

Практическая работа № 2

Цели работы:

✓ научиться решать задачи на разветвляющиеся алгоритмы;
✓ научиться использовать в программах условных операторов *if*, *case*.

Общие сведения. Алгоритм называется разветвляющимся, если он содержит несколько ветвей, отличающихся друг от друга содержанием вычислений. Выход вычислительного процесса на ту или иную ветвь алгоритма определяется исходными данными задачи. Перед выполнением работы необходимо ознакомиться с правилами записи логических выражений, операций сравнения, операторов *if, case*.

Пример. Дано действительное *x*. Для функции *f*, график которой представлен на рисунке 2, вычислить *f(x)*.

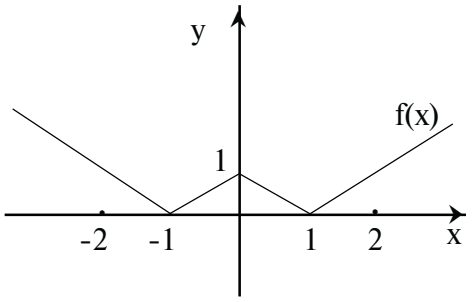


Рис. 2. График функции

Решение задачи

Математическая модель: функция вычисляется по следующей формуле:

$$f(x)=\begin{cases} -x-1, x<-1 \\ x-1, -1\le x<0 \\ -x+1, 0\le x<1 \\ x+1, x\ge 1 \end{cases}.$$

Составим схему алгоритма, детализировав все блоки (см. рис. 2). Дальнейшая детализация не требуется. Переводим алгоритм на язык *Pascal*:

```
Program example1;
var x, f:Real;
begin
  Write('Введите x: '); Readln(x);
 if x<-1 then f:= -x-1 else
   if (x>=-1) and (x<0) then f:= x-1 else
     if (x>=0) and (x<1) then f:= -x+1 else f:= x+1;
 Writeln('F= ',f:6:2);
 Readln;
end.
```

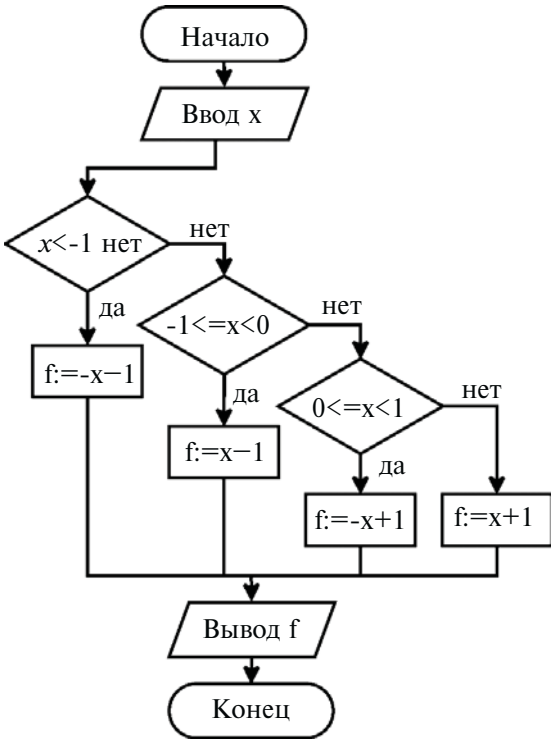


Рис. 3. Блок-схема ветвления в четырех направлениях

Варианты заданий

1. Используя оператор *if,* вычислить заданное выражение для данных типа *Integer:*

a) $f(x)=\begin{cases} x^2+3, x<0 \\ 6\sqrt{x}, 0\le x\le 5 \\ -x+9, x>5 \end{cases}$;

b) $f(x)=\begin{cases} 3|x|, x<-2 \\ 9x, -2\le x\le 2 \\ \sin x, x>2 \end{cases}$;

c) $f(x)=\begin{cases} \sin^2 x-\cos^2 x, x<0 \\ \ln(3x+2), 0\le x\le 2 \\ x^2-x^3, x>2 \end{cases}$;

d) $f(x)=\begin{cases} |2x-2|, x<-2 \\ \sin x, -2\le x\le 5 \\ x^4, x>5 \end{cases}$.

2. Найти алгоритм решения задачи и реализовать его с помощью оператора (операторов) *if-then-else*:

A. Составить программу, реализующую эпизод сказки: машина спрашивает, куда пойдет герой. В зависимости от ответа (налево — (-1), прямо — 0, направо — 1) печатает, что произойдет с героем.

B. Морской бой. Машина задумывает два числа от 0 до 9. Игрок пытается их угадать, вводя свои два числа. Если они совпали (в любом сочетании), то игрок выиграл.

C. В Атлантическом океане терпит бедствие пассажирский теплоход «Посудина». Все пассажиры будут спасены, если на помощь успеют два судна. Судно продержится на плаву t часов. Скорость судов-спасателей — 40 узлов.

Составить программу, определяющую спасутся ли пассажиры. Известны расстояния от трех судов-спасателей до тонущего судна.

D. Через старый мост движется поток автомашин. Одновременно на мосту могут находиться 3 машины. Если на мост въедут 3 легковых или 2 легковых и грузовик — мост выдержит. Если 2 грузовика и легковая или 3 грузовика — рухнет.

3. Используя оператор выбора, составить программы решения следующих задач:

a) по номеру дня недели вывести на печать рабочий это день или выходной, считая выходными субботу и воскресенье;

b) по номеру месяца указать, к какому времени года он относится;

c) по номеру месяца вывести на печать количество дней в нем;

d) единицы массы пронумерованы следующим образом:

1 — килограмм,
2 — миллиграмм,
3 — грамм,
4 — тонна.

Дан номер единицы массы и масса тела M в этих единицах (M — вещественное число). Вывести массу данного тела в килограммах.

Дополнительные задания

1. Даны действительные числа *a, b, c, x, y*. Выяснить, пройдет ли кирпич с ребрами *a,b,c* в прямоугольное отверстие со сторонами x и y. Просовывать кирпич в отверстие разрешается только так, чтобы каждое из его ребер было параллельно или перпендикулярно каждой из сторон отверстия.

2. Сможет ли шар радиуса R пройти в ромбообразное отверстие со стороной P и острым углом Q?

Контрольные вопросы

1. Какие операторы используются для программирования развилок?

2. Как выполняются операторы условного перехода?

3. Какую из функций: sin(x), abs(x), trunc(x) можно заменить условным оператором *if* $x < 0$ *then* $x := -x$?

4. Если выбор вариантов осуществляется из конечного числа элементов выбора, то лучше взять для этого оператор *if* или *case*?

5. Как заменить оператор case операторами *if*?

6. В чем преимущество оператора *case* от последовательности «коротких» операторов *if*?

7. Какой тип переменной можно использовать в качестве ключа оператора *case*?

Практическая работа № 3

Цели работы:

✓ закрепить практические навыки работы с системой программирования;

✓ научиться составлять программы с использованием циклических структур и выбирать для этого нужный оператор цикла.

Общие сведения. Алгоритм называется циклическим, если он содержит многократное выполнение одних и тех же операторов при различных значениях промежуточных данных.

Перед выполнением работы необходимо изучить различные схемы организации циклов и операторы *for, while, repeat* .

В этом разделе и последующих мы не будем приводить подробное решение задач, а ограничимся текстами программ с пояснениями.

Пример. На промежутке от 1 до *M* найти все числа Армстронга.

> Натуральное число из *n* цифр называется числом Армстронга, если сумма его цифр, возведенных в степень *n*, равна самому числу. Например, число 153 (153 = 13 + 53 + 33)

Решение. После организации ввода данных программа будет содержать цикл с параметром *i* (от 1 до *M*) с двумя вложенными циклами. Первый предназначен для подсчета количества цифр *n*; второй — для вычисления суммы *s* степеней цифр числа *i*. Если числа *i* и *s* равны, то *i* — число Армстронга, его необходимо вывести на экран:

```
PROGRAM Primer_1;
var i,k,s,p,n,M: Integer;
begin
 Write('Введите M '); Readln(M);
 for i:=1 to M do
  begin
    s:=0; k:=i; n:=0;
    while k<>0 do
       begin k:=k div 10; n:=n+1 end;
    k:=i;
    While k<>0 do
    begin p:=k mod 10; k:=k div 10;
       if p<>0 then s:=s+ Round(Exp(n*Ln(p)))
    end;
  if s=i then Writeln(i);
 end;
 Readln;
end.
```

Варианты заданий

1. Целочисленная арифметика:

a) найти количество натуральных двузначных чисел, каждое из которых делится на 3 и на 13;

b) найти количество натуральных четырехзначных чисел, каждое из которых не делится ни на 2, ни на 3;

c) найти количество натуральных чисел, не превосходящих 1000, каждое из которых кратно 25 и не кратно 3;

d) найти те натуральные числа, не превосходящие x, которые при делении на 10 дают в остатке 5.

2. Найти алгоритм решения задачи и реализовать его в виде Паскаль-программы:

a) начальный вклад в банк составляет a рублей. Через сколько лет он станет больше b рублей? Каждый год вклад увеличивается на 3 %;

b) ежегодный прирост рыбы в пруду составляет 15 %. Запасы рыбы оценены в A тонн. Ежегодный план отлова — B тонн. Подсчитать, сколько лет можно выдерживать заданный план;

c) каждая бактерия делится на две в течение одной минуты. В начальный момент имеется A бактерий. Сколько времени потребуется, чтобы количество бактерий превзошло X?

d) определить количество посетителей салона, которых успеет обслужить мастер-стилист, если его рабочий день составляет t часов и известна продолжительность (в минутах) обслуживания каждого посетителя очереди (вводится пользователем).

3. Составить программу для решения следующих задач:

a) вычислить количество точек с целочисленными координатами, попадающими в круг радиуса R ($R > 0$) с центром в начале координат;

b) найти все натуральные числа от 1 до N, представимые в виде суммы кубов двух натуральных чисел;

c) найти все натуральные числа от 1 до N, представимые в виде суммы квадратов трех натуральных чисел;

d) даны натуральные M, N ($M < N$). Найти числитель и знаменатель несократимой правильной дроби p/q такой, что $p/q = m/n$.

Контрольные вопросы

1. Как записывается и как работает оператор *for*?

2. Для организации каких циклов применим оператор *for*?

3. В чем отличие оператора *while* от оператора *repeat*?

4. Как программируются циклические алгоритмы с явно заданным числом повторений цикла?

5. Напишите пример оператора цикла, который не выполняется ни разу.

6. С какими ограничениями реализована конструкция цикла со счетчиком?

7. Замените оператор «*repeat A until B*» равносильным фрагментом программы с оператором *while*.

Практическая работа № 4

Цели работы:

✓ научиться создавать и вызывать в Паскаль-программах процедуры и функции;

✓ уметь использовать рекурсивные функции.

Общие сведения. Процедуры и функции являются строительными блоками для программ и обеспечивают их модульность, поэтому представляют собой важнейшее средство повышения эффективности программирования.

Перед выполнением работы необходимо ознакомиться с правилами оформления и вызова процедур и функций в языке программирования *Pascal*.

Пример 1. Вычислить значения функции f(x) = 2 cos x + 3, при x = {1; 4; 7,5; 20}. Вывести результаты в два столбца:

✓ в первом — значения x,

✓ во втором — значения $f(x)$.

Вычисления провести двумя способами: с помощью функции и процедуры.

Решение. Аргумент и результат функции — действительные числа, поэтому используем тип *Real*. В теле функции будет только оператор присваивания — для вычисления значения выражения.

Процедура отличается строкой заголовка. Для передачи в основную программу результатов вычислений добавим параметр-переменную *fx*. Чтобы вывести результаты в виде таблицы, используем форматный вывод:

```
program proc_1;
function f (x: Real):Real;
begin
  f:=2*cos(x)+3
end;
procedure proc_f (x: Real; var fx: real);
begin
  fx:=2*cos(x)+3
end;
  var x, fx: real;
begin
 Writeln('с использованием процедуры:');
Writeln(' x f(x)');
x:=1; proc_f (x,fx); Writeln (x:6:2, fx:6:2);
x:=4; proc_f (x,fx); Writeln (x:6:2, fx:6:2);
x:=7.5; proc_f (x,fx); Writeln (x:6:2, fx:6:2);
x:=20; proc_f (x,fx); Writeln (x:6:2, fx:6:2);
  Readln;
Writeln('с использованием функции:');
Writeln(' x f(x)');
Writeln(1:6, f (1):6:2);
Writeln(4:6, f (4):6:2);
Writeln(7.5:6:2, f (7.5):6:2);
Writeln(20:6, f (20):6:2);
  Readln;
end.
```

Пример 2. Создать рекурсивную функцию поиска *i*-го члена последовательности, заданной рекуррентной формулой $A_1 = \text{const}_1$, $A_2 = \text{const}_2$, $A_i = 3_{Ai-2} - -A_{i-1}$. Вывести через пробел значения рекурсивной функции при значениях аргумента от 1 до 10 включительно.

Решение. По условию задачи аргумент может принимать только целые значения, поэтому функция имеет параметр-значение типа *Integer*. Выход из рекурсии в данном случае осуществляется при двух значениях аргумента (при $i = 1$, $i = 2$), поэтому в рекурсивной функции необходимы два вложенных условных оператора или же оператор выбора *case*.

В приведенном примере использованы операторы *if*, но попробуйте самостоятельно записать решение с помощью оператора выбора.

В основной программе значения аргумента — целые последовательные числа, поэтому следует воспользоваться оператором цикла с параметром *for*:

```
program proc_2;
function A (i: Integer): Integer;
begin
  if i=1 then A:=1 else
  if i=2 then A:=3 else A:=3*A(i-2)-A(i-1)
end;
  var i: Integer;
begin
  for i:=1 to 10 do Write (A(i),' ');
  Readln
end.
```

Варианты заданий

1. Составить программу для решения задачи с применением функции и процедуры пользователя.

A. В правильном треугольнике проведена средняя линия. Найти площадь образовавшейся трапеции, дважды используя функцию вычисления площади правильного треугольника по формуле:

$$s = \frac{a^2\sqrt{3}}{4}.$$

B. Для правильного треугольника со стороной a построены вписанная и описанная окружности. Найти площадь образовавшегося кольца, используя функцию вычисления площади круга $S = \pi R^2$.

Для нахождения радиусов окружностей воспользуйтесь формулами:

$$R = \frac{a\sqrt{3}}{3},\ r = \frac{a\sqrt{3}}{6}.$$

C. Тариф предусматривает оплату телефонных разговоров следующим образом: при продолжительности разговора меньше P минут стоимость одной минуты составляет S_1 копеек, в противном случае — S_2 коп/мин (S_1, S_2, P — константы). Используя функцию вычисления стоимости одного разговора, найти суммарную стоимость трех звонков известной продолжительности.

D. На товар дважды была сделана скидка — на p_1, а затем на p_2 процентов. Первоначальная стоимость товара составляла S рублей. Используя функцию вычисления стоимости товара с учетом скидки на P процентов, найти стоимость товара после двойной скидки:

$$C = \frac{100 - P}{100} \cdot S.$$

2. Вывести значения рекурсивной функции при значениях аргумента от 1 до 10 включительно.

A. Найти член последовательности, заданной формулой: $D_i = 7 + D_{i-1}$ при $i > 1$, где D_1 определяется пользователем;

B. Найти член последовательности, заданной формулой: $A_i = A_{i-1} - A_{i-2}$ при $i > 2$. Значения первого и второго членов последовательности вводятся пользователем;

C. Найти член последовательности, заданной следующим образом: $y_1 = 0$; $y_2 = 10$; $y_n = 2 \cdot y_{n-1} - y_{n-2}$, где $n > 2$;

D. Найти член последовательности, заданной формулой $B_i = 4 \cdot B_{i-1}$, при $i > 1$. Значения первого члена последовательности вводится пользователем.

Дополнительные задания

1. Для чисел *a, b, c* найти значение выражения min *(a, ab)* + min *(a, ac)* + 1 с использованием функции вычисления минимального из двух чисел.

2. Дан квадрат со стороной *a*. Диагональ этого квадрата является стороной второго квадрата; диагональ второго квадрата — стороной третьего. Найти длину стороны третьего квадрата, используя функцию вычисления длины диагонали квадрата по его стороне:

$$d = a\sqrt{2}.$$

Контрольные вопросы

1. В чем заключается принципиальное отличие процедур от функций?

2. Чем отличается вызов функции от вызова процедуры?

3. Какой переменной присваивается значение в процедуре и в функции?

4. Какие переменные в языке *Pascal* называются локальными, а какие — глобальными?

5. Какие процедуры и функции называют рекурсивными?

6. Как отличить «обычную» функцию от рекурсивной?

7. Если значения процедуры и функции являются операндами некоторого выражения, то которую из них можно вставить непосредственно в это выражение?

Практическая работа № 5

Цели работы:

✓ овладеть основными приемами работы с одномерными и двумерными массивами;

✓ уметь различать в двумерном массиве обработку строк и столбцов, а также отличать нахождение первых и последних элементов последовательности, обладающих некоторым свойством.

Общие сведения. Табличное представление информации одно из самых распространенных, поэтому массивы широко применяются в прикладных программах.

Перед выполнением работы необходимо ознакомиться с теоретическим материалом по теме «Обработка массивов».

Пример 1. Составить программу, позволяющую в одномерном массиве, состоящем из *N* вещественных элементов; вычислить сумму положительных элементов.

Решение. При написании процедур ввода и вывода следует обратить внимание на то, что элементы — вещественные числа, поэтому необходимо позаботиться о верной обработке дробной части.

Вычисление суммы оформим в виде функции с одним аргументом — массивом.

Локальными переменными функции будут индексная переменная *i* и дополнительная переменная *s* для хранения текущей суммы элементов.

В начале тела функции обязательно обнуление *s*.

Каждый элемент массива сравним с нулем. Если значение положительно, добавим его к искомой сумме *s*.

В конце функции запишем значение переменной *s* в результирующую переменную:

```
program massiv_1;
const N=10;

type mas=array [1..N] of Real;
procedure Vvodmas(var A:mas);
  var i:Integer;
begin
  for i:=1 to N do A[i]:=-50+Random(101)+random;
end;

procedure Vivodmas(A:mas);
  var i:Integer;
begin
  for i:=1 to N do Write(A[i]:8:2);
Writeln
end;

function Summa(A:mas):real;
  var i: Integer; s:real;
begin
  s:=0;
  for i:=1 to N do if A[i]>0 then s:=s+A[i];
 Summa:=s;
end;

var A: mas;
begin
  Randomize; Vvodmas(A);
```

```
  Writeln('Исходный массив:'); Vivodmas(A);
  Writeln('Ответ: ', Summa(A):0:2);
  Readln
end.
```

Пример 2. В двумерном массиве, состоящем из целочисленных элементов, в каждом столбце поменять местами наибольший по модулю и последний не принадлежащий интервалу *(a, b)* элементы массива.

Решение. Преобразования необходимо провести в каждом столбце массива, поэтому параметр внешнего цикла в процедуре обработки — номер столбца *j*, а вложенного — номер строки *i*.

Для перестановки двух элементов в столбце массива необходимо найти номера их строк *n1* и *n2*, а затем поменять местами значения элементов с использованием промежуточной переменной *p*.

Чтобы найти наибольший по модулю элемент столбца, введем дополнительную переменную max, которая будет хранить максимальное по модулю значение в текущем столбце массива на данный момент.

Можно решить задачу без использования переменной max. Подумайте, как это сделать.

Программа должна корректно работать с любыми входными данными, а значит и в тех случаях, когда некоторые или даже все столбцы массива содержат только элементы из интервала *(a, b)* и обмен значений в некоторых столбцах или во всем массиве не нужен:

```
const n=10; m=7;
type mas=array [1..n,1..m] of Integer;

procedure Vvodmas(var D:mas);
  var i,j:Integer;
begin
for i:=1 to n do
  for j:=1 to m do
  D[i,j]:=-50+Random(101);
end;

procedure Vivodmas(D:mas);
  var i,j:Integer;
begin
  for i:=1 to n do
  begin
  for j:=1 to m do Write(D[i,j]:4);
  Writeln;
  end;
end;

procedure Obmen(a,b: real; var D:mas);
  var i,j,p,n1,n2,max: Integer;
begin
  for j:=1 to m do
  begin
```

```
      n1:=1; max:=abs(D[1,j]);{считаем первый элемент столбца наибольшим по
модулю}
      for i:=2 to n do
      if abs(D[i,j])>max then {обнаружен больший элемент}
      begin n1:=i; max:=abs(D[i,j]) end;

      i:=n; {перебираем элементы столбца, начиная с последнего}
      while (i>=1)and (D[i,j]>a)and(D[i,j]<b) do i:=i-1;
      n2:=i;
      if n2<>0 then {если элемент, не принадлежащий интервалу (a,b), был най-
ден}
      begin
        p:=D[n1,j]; D[n1,j]:=D[n2,j]; D[n2,j]:=p; {обмен значений}
      end;
    end;
  end;

  var D: mas; a,b:Real;
  begin
    Randomize; Vvodmas(D);
    Writeln('Исходный массив:'); Vivodmas(D);
    Write('Введите через пробел концы интервала (a,b): '); Readln(a,b);
    Obmen(a,b,D);
    Writeln('Ответ:'); Vivodmas(D);
    Readln
  end.
```

Варианты заданий

1. Составить программу, позволяющую в одномерном массиве, состоящем из N вещественных элементов, вычислить:

a) сумму модулей отрицательных элементов массива;

b) количество элементов массива, не принадлежащих интервалу (a, b);

c) наименьший из элементов массива, принадлежащих отрезку $[a, b]$;

d) количество элементов массива, равных первому элементу.

2. В двумерном массиве, состоящем из целочисленных элементов, поменять местами:

a) в каждом столбце наибольший по модулю и последний положительный элементы;

b) в каждом столбце первый и последний отрицательные элементы;

c) в каждой строке наибольший и наименьший элементы;

d) в каждом столбце первый принадлежащий отрезку $[a, b]$ и первый отрицательный элементы.

Дополнительные задания

1. Определить в одномерном массиве число соседств из двух чисел разного знака.

2. Дан двумерный массив целых чисел. Поменять местами строку, содержащую максимум массива, со строкой, содержащей его минимум.

Контрольные вопросы

1. Как описываются в языке *Pascal* одно- и двумерные массивы?

2. Может ли массив содержать разнотипные данные?

3. В каком порядке указывают индексы при обращении к элементам двумерного массива?

4. Привести пример массива *var A: array [1..3, 20..24] of real.*

5. Можно ли при обработке двумерных массивов использовать однократные циклы? Если да, то приведите примеры.

6. Если в одномерном массиве проверяется «похожесть» его первой и второй части, то в каких границах надо писать оператор *for* для прохождения этого массива?

7. Каким образом надо находить первый и последний элементы одномерного массива, обладающие некоторым свойством (отрицательный, наибольший, входящий в интервал и прочее)?

Практическая работа № 6

Цели работы:

✓ освоить строковые операции, процедуры и функции;

✓ научиться создавать программы обработки текстовых данных, представляющих последовательность символов и строк, состоящих из слов, разделенных одним или несколькими пробелами.

Общие сведения. Для обработки текстовой информации можно использовать те же методы, что применяют для одномерных массивов, так как структура строкового типа схожа с массивом. Однако для упрощения написания программ по работе с текстами разработаны стандартные строковые процедуры и функции. Поэтому основной задачей этой лабораторной работы является освоение строковых операций, функций *Length, Pos, Copy* и процедур *Delete, Insert.*

Перед выполнением работы необходимо ознакомиться с теоретическим материалом по теме «Обработка литерных величин. Данные типа *Char* и *String*».

Пример 1. Составить программу обработки данной строки, позволяющую выписать все знаки сравнения и все скобки, сохранив их последовательность.

Решение. Все знаки сравнения и скобки перечислим в строковой константе *srav_sk*. В теле программы последовательно рассмотрим все символы введенной строки *s*, проверяя каждый на вхождение в строку *srav_sk*, выводя на экран содержащиеся в строковой константе символы строки *s*:

```
program string_1;
const srav_sk='<>=(){}[]';
var s:String; i:Integer;
begin
  Writeln('Введите строку:'); Readln(s);
  for i:=1 to Length(s) do
  if Pos(s[i],srav_sk)<>0 then write(s[i]);
  Readln
end.
```

Пример 2. Дана строка, состоящая из слов, разделенных пробелами (одним или несколькими). Составить программу обработки данной строки, выводящую через запятую слова четной длины, которые при обмене местами левой и правой половины дают то же слово (например, «папа», «мама», «канкан»).

Решение. В строке, последовательно перебирая элементы, будем выделять начало *ns* и конец *ks* каждого слова; копировать слово в другую строковую переменную; проверять четность длины и равенство частей строки «*slovo*». Слова, для которых выполнено условие задания, будем склеивать в переменной *otv*.

После добавления к ответу последнего слова в конце *otv* будет лишняя запятая, которую перед выводом на экран необходимо удалить:

```
program string_2;
var s, slovo, otv:String; ns, ks, i, m: Integer;
begin
  Writeln ('Введите строку:'); Readln(s);
  i:=1; ns:=1; otv:=''; s:=s+' ';
  while i<=Length(s) do
  begin
    while (i<=Length(s)) and (s[i]<>' ') do i:=i+1; {ищем очередной пробел}
    ks:=i; slovo:=Copy(s, ns, ks-ns); {выделяем слово}
    m:=Length(slovo) div 2; {половина длины слова}
    if (Length(slovo) mod 2=0) and (Copy(slovo,1,m)=Copy(slovo,m+1,m))
      {если слово имеет четную длину}
      then otv:= otv+slovo+ ','; {добавляем слово к ответу}
    while (i<=Length(s)) and (s[i]=' ') do i:=i+1; {пропускаем пробелы}
    ns:=i; {начало следующего слова}
  end;
  Delete(otv, Length(otv), 1); {удаляем лишнюю запятую в конце}
  Writeln(otv);
  Readln
end.
```

Варианты заданий

1. Составить программу обработки данной строки, позволяющую:

a) выписать все прописные буквы из данной строки в порядке их следования;

b) определить, содержатся ли в тексте цифры;

c) найти общее количество круглых, квадратных и фигурных скобок в тексте;

d) выписать все знаки препинания из данной строки, сохранив их последовательность.

2. Дана строка, состоящая из слов, разделенных пробелами (одним или несколькими). Составить программу обработки данной строки, выводящую через запятую следующие слова:

a) одинаково читающиеся справа налево и слева направо;

b) заданной длины, в которые входит данная буква;

c) начинающиеся с прописной буквы, в которых все остальные буквы строчные;

d) начинающиеся и заканчивающиеся одной и той же буквой и содержащие хотя бы одну введенную с клавиатуры букву.

Дополнительное задание

1. Дана строка, состоящая из слов, разделенных пробелами (одним или несколькими). Составить программу обработки данной строки, позволяющую:

a) определить количество слов в строке;

b) удалить из строки избыточные пробелы, чтобы между словами осталось только по одному пробелу;

c) найти самое короткое слово и его длину;

d) найти в тексте слова-перевертыши.

Контрольные вопросы

1. Как описываются в языке *Pascal* строковые величины?

2. В чем сходство и различие между массивами и строками?

3. Существуют ли ограничения, накладываемые на длину строки?

4. Какие строковые процедуры существуют в языке *Pascal*?

5. Для вывода значений каких строковых функций нужны переменные типа *String*, а для каких — *Integer*?

6. Какие есть возможности извлечения из строки одного символа?

7. Строка для обработки процедурой должна быть ее параметром-аргументом или параметром-результатом?

Практическая работа № 7

Цель: научиться использовать объекты для решения задач.

Пример. Известны длины сторон треугольника a, b и c (рис. 4). Вычислить площадь S, периметр P и величины углов α, β и γ треугольника.

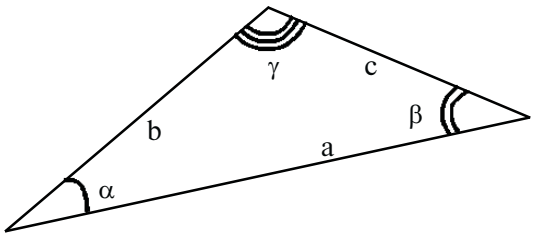


Рис. 4. Треугольник

Решение. Прежде чем приступить к написанию программы, вспомним математические формулы, необходимые для решения задачи.

Для вычисления площади треугольника применим теорему Герона:

$$S = \sqrt{p(p-a)(p-b)(p-c)},$$

где полупериметр:

$$p = (a + b + c) / 2.$$

Один из углов найдем по теореме косинусов:

$$\cos = \frac{b^2 + c^2 - a^2}{2bc};$$

второй — по теореме синусов:

$$\sin(\beta) = \frac{b}{a}\sin(\alpha);$$

третий — по формуле:

$$\gamma = \pi - (a + \beta).$$

Решение задачи разобьем на следующие этапы:

1. Определение значений *a*, *b* и *c* (ввод величин *a*, *b* и *c* в память компьютера).
2. Расчет значений *S*, *P*, α, β и γ по приведенным формулам.
3. Вывод значений *S, P,* α, β и γ.

Попытайтесь самостоятельно разработать внешний вид данной программы. Разместите на форме десять меток, три поля ввода и одну кнопку (рис. 5).

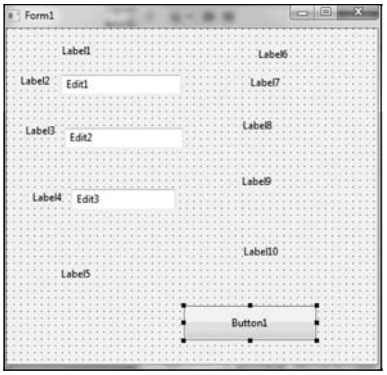


Рис. 5. Вид формы с компонентами

Измените их заголовки (свойство *Caption*) в соответствии с таблицей 7.

Таблица 7

Свойства компонента

Компонент	Свойство *Caption*
Form1	Параметры треугольника
Label1	Введите длины сторон
Label2	a=
Label3	b=
Label4	c=
Label5	Величины углов
Label6	alfa=
Label7	betta=
Label8	*gamma*=
Label9	Периметр P=
Label10	Площадь S=
Button1	ВЫЧИСЛИТЬ

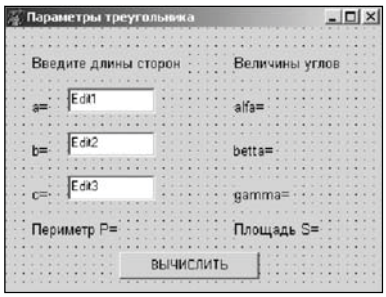


Рис. 6. Интерфейс программы

Двойной щелчок по кнопке «Вычислить» приведет к созданию процедуры TForm1.Button1Click.

Задача программиста — заполнить шаблон описаниями и операторами. Все команды, указанные в процедуре между словами *begin* и *end*, будут выполнены при щелчке по кнопке «Выполнить». В нашем случае процедура TForm1. Button1Click будет иметь вид:

```
procedure TForm1.Button1Click(Sender: TObject);
//Описание переменных:
// a, b, c — стороны треугольника;
// alfa, betta, gamma — углы треугольника;
// S - площадь треугольника;
// r - полупериметр треугольника
//Все переменные вещественного типа.
var a, b, c, alfa, betta, gamma, S,p: real;
begin
//Из полей ввода Edit1, Edit2, Edit3
//считываются введенные строки,
//с помощью функции StrToFloat(x)
//преобразовываются в вещественные числа
//и записываются в переменные a, b, c.
a:=StrToFloat(Edit1.Text);
b:=StrToFloat(Edit2.Text);
c:=StrToFloat(Edit3.Text);
//Вычисление значения полупериметра.
p:=(a+b+c)/2;
//Вычисление значения площади,
//для вычисления применяется функция:
// sqrt(x) — корень квадратный из x.
S:=sqrt(p*(p-a)*(p-b)*(p-c));
//Вычисление значения угла alfa в радианах.
//Для вычисления применяем функции:
// arccos(x) - арккосинус x;
// sqr(x) — возведение x в квадрат.
alfa:=arccos((sqr(b)+sqr(c)-sqr(a))/2/b/c);
//Вычисление значения угла betta в радианах.
//Для вычисления применяем функции:
// arcsin(x) - арксинус x;
```

```
betta:=arcsin(b/a*sin(alfa));
//Вычисление значения угла gamma в радианах.
//Математическая постоянная определена
//функцией без аргумента pi.
gamma:=pi-(alfa+betta);
//Перевод радиан в градусы.
alfa:=alfa*180/pi;
betta:=betta*180/pi;
gamma:=gamma*180/pi;
//Для вывода результатов вычислений используем
//операцию слияния строк ?+?
//и функцию FloatToStrF(x), которая
//преобразовывает вещественную переменную x
//в строку и выводит ее в указанном формате,
//в нашем случае под переменную отводится
//три позиции, включая точку
//и ноль позиций после точки.
//Величины углов в градусах выводятся на форму
//в соответствующие объекты типа надпись.
Label6.Caption:='alfa='+ FloatToStrF(alfa,ffFixed,3,0);
Label7.Caption:='betta='+ FloatToStrF(betta,ffFixed,3,0);
Label8.Caption:='gamma='+FloatToStrF(gamma,ffFixed,3,0);
//Используем функцию FloatToStrF(x)
//для форматированного вывода, в нашем случае
//под все число отводится пять позиций,
//включая точку, и две позиций после точки.
//Значения площади и периметра
//выводятся на форму.
Label9.Caption:='Периметр P='+FloatToStrF(2*p,ffFixed,5,2);
Label10.Caption:='Площадь S='+FloatToStrF(S,ffFixed,5,2);
end;
```

Обратите внимание, что было написано всего десять команд, предназначенных для решения поставленной задачи. Все остальное — комментарий, который писать необязательно.

Задачи для самостоятельного решения

Разработать программу в среде программирования *Lazarus*. Для каждой задачи создать интерфейс, соответствующий следующему условию:

1. Заданы два катета прямоугольного треугольника. Найти гипотенузу и углы треугольника.

2. Известна диагональ квадрата *d*. Вычислить площадь *S* и периметр *P* квадрата.

3. Треугольник задан величинами своих сторон *a*, *b*, *c*. Найти углы треугольника — α, β, γ.

4. Тело имеет форму параллелепипеда с высотой *h*. Прямоугольник в основании диагональ — *d*. Известно, что диагонали основания пересекаются под углом *a*. Найти объем тела *V* и площадь поверхности *S*.

5. Задан первый член геометрической прогрессии и ее знаменатель. Вычислить сумму *n* членов геометрической прогрессии и значение *n*-го члена.

6. Тело падает с высоты *h*. Какова его скорость в момент соприкосновения с землей? когда это произойдет?

Практическая работа № 8

Цель: научиться использовать компоненты *StringGrid* и поле *Memo* для работы с массивом.

Пример 1. Рассмотрим программу, которая вычисляет среднее арифметическое значение элементов массива.

Решение. Диалоговое окно программы приведено на рисунке 7.

Компонент *StringGrid* используется для ввода массива, компоненты *Label*1 и *Label*2 — для вывода пояснительного текста и результата расчета; *Button*1 — для запуска процесса расчета (рис. 7).

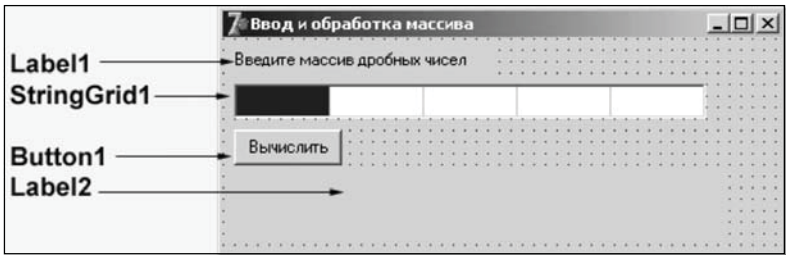


Рис. 7. Окно интерфейса программы

Добавляется компонент *StringGrid* в форму точно так, как и другие компоненты.

После добавления компонента к форме нужно выполнить его настройку в соответствии с таблицей 8.

Значения свойств *Height* и *Width* следует при помощи мыши установить так, чтобы размер компонента был равен размеру строки.

Таблица 8

Значение свойств

Свойство	Значение
ColCount	5
FixedCols	0
RowCount	1
DefaultRowHeight	24
Height	24
DefaultColWidth	64
Width	328
Options .goEditing	True
Options .AlwaysShowEditing	True
Options .goTabs	True

На кнопку используйте процедуру:

```
procedure TForml.ButtonlClick(Sender: Tobject);
 var
a : array[1..5] of integer; // массив
summ: integer; // сумма элементов
sr: real; // среднее арифметическое
i: integer; // индекс
begin
// ввод массива
// считаем, что если ячейка пустая, то соответствующий
// ей элемент массива равен нулю
for i:= 1 to 5 do
if Length(StringGridl.Cells[i-1, 0]) 0
then a[i] := StrToInt(StringGridl.Cells[i-1,0])
else a[i] := 0;
// обработка массива
summ := 0;
for i :=1 to 5 do
summ := summ + a[i]; sr := summ / 5;
Label2.Caption :='Сумма элементов: ' + IntToStr(summ)+ #13+ 'Среднее арифметическое: ' + FloatToStr(sr);
end;
```

Чтобы курсор автоматически переходил в следующую ячейку таблицы, например, в результате нажатия клавиши, добавим процедуру обработки события *OnKeyPress*. На нее можно также возложить задачу фильтрации вводимых в ячейку таблицы данных. В нашем случае надо разрешить ввод в ячейку только цифру. Текст процедуры обработки события *OnKeyPress* приведен ниже.

Следует обратить внимание на свойство *Col*, которое во время работы программы содержит номер колонки таблицы, в которой находится курсор. Это свойство можно использовать и для перемещения курсора в нужную ячейку таблицы. Однако нужно учитывать, что колонки таблицы (впрочем, как и строки) нумеруются с нуля.

```
Procedure Tform1.StringGridlKeyPress(Sender: Tobject; var Key: Char);
begin
case Key of
#8,'0'..'9' : ; // цифры и клавиша Backspace
#13: // клавиша
if StringGridl.Col
then StringGridl.Col := StringGridl.Col+1;
else key := Chr(0); // остальные символы запрещены
end;
end;
```

Запустите программу, посмотрите как изменилась работа программы.

Пример. Рассмотрим задачу, в которой компонент *Memo* используется для вывода символьного массива.

Решение. Создайте интерфейс окна программы, как показано на рисунке 8.

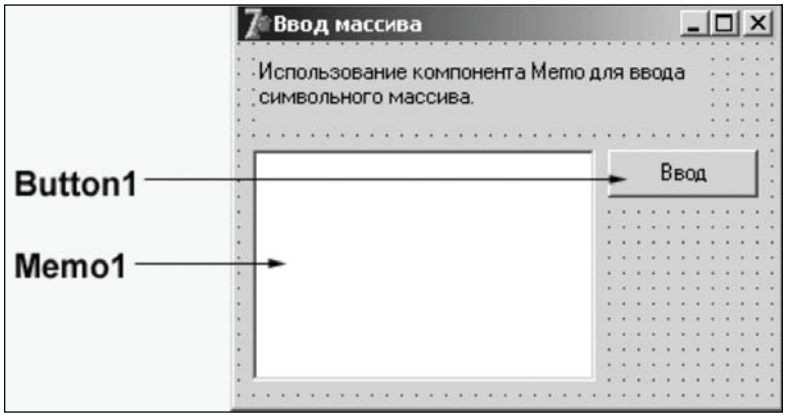


Рис. 8. Диалоговое окно приложения «Ввод массива»

На кнопку используйте процедуру:

```
procedure Tforml .ButtonlClick(Sender: Tobject);
const
SIZE=5; // размер массива
var
a:array[l..SIZE]of string[30]; //массив
n: integer; // количество строк, введенных в поле Memo
i:integer; // индекс элемента массива
st:string;
begin
n:=Memo1.Lines.Count;
if n = 0 then begin
ShowMessage('Исходные данные не введены!');
Exit; // выход из процедуры обработки события
end;
// в поле Memo есть текст
if n SIZE then begin
ShowMessage('Количество строк превышает размер массива.');
n:=SIZE; // будем вводить только первые SIZE строк
end;
for i:=1 to n do
a[i]:=Form1.Memol.Lines[i-1]; // строки Memo пронумерованы с нуля
// вывод массива в окно сообщения
if n 0 then begin
st:='Введенный массив:'+#13;
for i: =1 to n do
st:=st+IntToStr(i)+' '+ a[i]+f13; ShowMessage(st);
end;
end;
```

Задачи для самостоятельного решения

1. Записать положительные элементы массива *x* подряд в массив *y*. Вычислить сумму элементов массива *x* и произведение элементов массива *y*.

2. Из массива *y* удалить элементы, расположенные между максимальным и минимальным элементами.

3. Сформировать массив *b*, записав в него элементы массива *a* с нечетными индексами. Вычислить среднее арифметическое элементов массива *b* и удалить из него максимальный, минимальный и пятый элементы.

4. Дан массив целых чисел *x*. Переписать пять первых положительных элементов массива и последних два простых элемента в массив *y*. Найти максимальный отрицательный элемент массива *x*.

Практическое работа № 9

Цель: научиться использовать свойство объекта *Canvas* для рисования графики.

Решение. Создадим форму, установим ей размеры — Heigth — 500, Width — 500. Внизу разместим кнопку, зададим ей свойство *Caption* — «рисовать».

При запуске программы и щелчке по этой кнопке на форме прорисуются различные фигуры, рисунок. Ниже приведен листинг программы, демонстрирующий работу перечисленных методов. Результат работы программы приведен на рисунке 9.

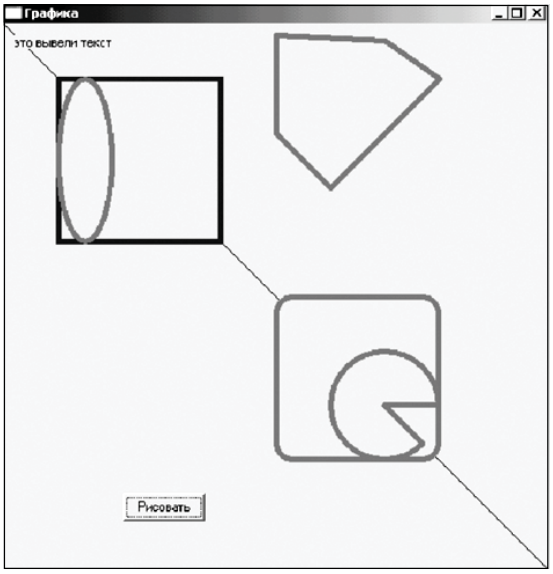


Рис. 9. Пример использования методов рисования фигур

```
unit Unit1;
{$mode objfpc}{$H+}
interface
uses
Classes, SysUtils, Lresources, Forms,
Controls, Graphics, Dialogs, StdCtrls;
type
```

```
{ Tform1 }
Tform1 = class(Tform)
Button1: Tbutton;
procedure Button1Click(Sender: Tobject);
private
{ private declarations }
public
{ public declarations }
end;
var
Form1: Tform1;
implementation
{ Tform1 }
procedure Tform1.Button1Click(Sender: Tobject);
var t: array [1..5] of Tpoint;
begin
Form1.Canvas.LineTo(500,500);//Рисование линии
//Изменение цвета и толщины линии.
Form1.Canvas.Pen.Color:= clMaroon;
Form1.Canvas.Pen.Width:= 5;
//Рисование прямоугольника.
Form1.Canvas.Rectangle(50,50,200,200);
Form1.Canvas.Pen.Color:= clolive;
//Рисование эллипса.
Form1.Canvas.Ellipse(50,50,100,200);
//Рисование прямоугольника с скругленными углами.
Form1.Canvas.RoundRect(250,250,400,400,30,30);
//Рисование сектора окружности.
Form1.Canvas.Pie(300,300,400,400,350,350,500,500);
//Массив координат вершин пятиугольника.
T[1].x:=250; t[1].y:=10;
t[2].x:=350; t[2].y:=15;
t[3].x:=400; t[3].y:=50;
t[4].x:=300; t[4].y:=150;
t[5].x:=250; t[5].y:=100;
Form1.Canvas.Polygon (t) ;
Form1.Canvas.TextOut(10,10,'это вывели текст');
end;
initialization
{$I unit1.lrs}
end.
```

Задание. Самостоятельно составьте программу для рисования нескольких окружностей, со сдвигом центра окружности на шаг *h*.

ПРИЛОЖЕНИЕ 2. ДИФФЕРЕНЦИРОВАННЫЕ ЗАДАНИЯ ПО ОСНОВНЫМ РАЗДЕЛАМ ЭЛЕКТИВНОГО КУРСА «АЛГОРИТМИНИЗАЦИЯ И ПРОГРАММИРОВАНИЕ»

Принципы работы с системой Turbo Pascal

Задания первого уровня

Упражнение 1. Создать программу, вычисляющую длину гипотенузы *c* и величины двух углов α и β прямоугольного треугольника, если заданы длины двух его катетов — *а* и *б*. Значения катетов вести с клавиатуры.

1. Наберите текст программы:

```
Program Pr_1;                          {Заголовок программы Pr_1}
var                                    {Раздел описаний}
  a, b, c, alf, bet : real;            {Переменные a, b, c, alf, bet -
                                        вещественные}
Begin                                  {Тело программы}
  Write('a=');                         {Вывод запроса на экран}
 Read (a);                             {Ввод значения a с клавиатуры}
  Write('b=');                         {Вывод запроса на экран}
  Read (b);                            {Ввод значения b с клавиатуры}
  c:=sqrt(a*a+b*b);                    {Вычисление гипотенузы c}
  alf:=arctan(a/b);                    {Вычисление угла alf }
  bet:=arctan(b/a);                    {Вычисление угла bet }
  Writeln('c=', c:6:2);  {Вывод ответа}
  Writeln('Радианы');  {Вывод ответа: углы в радианах}
  Writeln('alf=', alf:6:2, 'bet=', bet:6:2);
End.                                   {Конец программы}
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9*.

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5*.

4. Сохраните программу на своем диске: *<F2> A:\P1PR1*.

Упражнение 2. Закрепить навыки набора и редактирования текста.

1. Отредактируйте текст программы, добавив вывод значений углов в градусах:

```
Writeln('Градусы');                    {Вывод ответа: углы в градусах}
Writeln('alf=', alf*180/ pi :3:0, 'bet=', bet*180/ pi :3:0);
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9*.

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5*.

4. Сохраните программу на своем диске: *<F2> A:\P1PR2*.

Самостоятельная работа

Задания второго уровня

1. Заданы радиус основания и высота цилиндра. Вычислить площадь основания и объем.

Площадь основания вычисляется по формуле: объем цилиндра равен $V = S \cdot h$.

Сохранить как *P1PR3*.

2. Вычислите среднее арифметическое $(a + b) / 2$. Сохранить программу как *P1PR4*.

3. Составить программу вычисления площади и периметра прямоугольника по двум введенным сторонам. Предусмотреть вывод на экран сообщения о результате.

Сохранить как *P1PR5*.

4. За решение четырех задач студент получил оценки (от 1 до 5). Составить программу, которая определит среднее значение оценок, полученных студентом за задачи.

Сохранить программу как *P1PR6*.

Задания третьего уровня

1. Четыре человека пообедали в ресторане. Официант подал каждому счет. Они решают оставить официанту чаевые в размере 15 % от счета. Составить программу, которая выведет на экран сумму чаевых, которую получил официант.

Сохранить программу как *P1PR7*.

2. Составить программу, которая определяет, сколько времени в минутах затратит школьник на дорогу из школы до стадиона, если известны длина этого расстояния S км и средняя скорость движения школьника v км/ч?

Сохранить программу как *P1PR8*.

3. Три четверти пассажиров самолета имеют билеты II класса стоимостью X рублей каждый. Остальные пассажира имеют билеты I класса, которые стоят в два раза дороже билетов II класса. Написать программу, которая выведет сумму денег, получаемую авиакомпанией от продажи билетов на этот рейс, если салон самолета рассчитан на пять пассажиров.

Сохранить программу как *P1PR9*.

Структура программы. Описание переменных в Turbo Pascal

Задания первого уровня

Упражнение 1. Вычислить месячные выплаты m по займу в s рублей на n лет под процент p. Вычисления выполнить по формулам:

$$m = \frac{sr\ (1+r)^n}{12\left((1+r)^n - 1\right)};$$

$$r = p/100.$$

1. Наберите текст программы:

```
Program Zaim;
var
 m, s, p, r, n, a, d: real;
 rub, kop: integer;                  {Целая и дробная часть числа (рубли и копейки)}
Begin
  writeln ('Введите заем, процент и количество лет в одной строке');
  readln (s, p, n);
  r:=p/ 100;
  a:=exp (ln(1+r)*n);                {Вычисление степени числа через логарифм}
  m:=(s*r*a)/(12*(a-1));
  m:=trunc (100*m+0.5)/ 100;         {Округление до копейки}
  d:=m*n*12 - s;                     {Общая прибыль}
  writeln;
  rub:=round (s*100) div 100;        {Преобразование числа в денежный формат (выделения руб.и коп.)}
  kop:= round (s*100) mod 100;
  write ('Взято ', rub, 'руб.', kop, 'коп.');
  write ('под ', p:5:2, '% на ', n:5:2, 'лет');
  writeln;
  rub:=round (m*100) div 100;        {Преобразование числа в денежный формат (выделения руб.и коп.)}
  kop:= round (m*100) mod 100;
  writeln ('Месячная выплата = ', rub, 'руб.', kop, 'коп.');
  rub:=round (d*100) div 100;        {Преобразование числа в денежный формат (выделения руб.и коп.)}
  kop:= round (d*100) mod 100;
  writeln ('Общая прибыль= ', rub, 'руб.', kop, 'коп.')
End.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2>* *A:\P2PR1.*

Упражнение 2. Вычислить сумму цифр трехзначного числа.

1. Наберите текст программы:

```
Program chislo;
var
 i, first, second, third, sum: integer;
Begin
  write ('Введите целое трехзначное число: ');
  readln (i);
  first := i div 100;                {Выделение первой цифры числа}
  second := i div 10 mod 10;         {Выделение второй цифры числа}
  third := i mod 10;                 {Выделение третьей цифры числа}
  sum := first + second + third;
  writeln ('Сумма цифр числа ', 100 * first+10 * second + third, '=', sum)
End.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2>* *A:\P2PR2.*

Упражнение 3. Перевести английские названия дней недели на русский язык.

1. Наберите текст программы:

```
Program WEEK;
type days=(mon, tue, wed, thu, fri, sat, sun)
var
  d : days;
Begin
  for d:=mon to sun do
  case d of
    mon: writeln('понедельник');
    tue: writeln('вторник');
    wed: writeln('среда');
    thu: writeln('четверг');
    fri: writeln('пятница');
    sat: writeln('суббота');
    sun: writeln ('воскресенье');
    end
End.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2>* *A:\P2PR3.*

Самостоятельная работа

Задания второго уровня

1. Вычислить произведение цифр трехзначного числа. Сохраните как *P2PR4.*

2. Заданы длины трех сторон треугольника *a*, *b*, *c*. Вычислить периметр и площадь треугольника по формуле Герона:

$$s = \sqrt{p(p-a)(p-b)(p-c)}.$$

Сохранить как *P2PR5.*

Задания третьего уровня

1. Вычислить значение выражения по формуле (все переменные принимают действительные значения):

a) $\dfrac{3+e^{y-1}}{1+x^2\left|y-tgx\right|}$;

b) $x-\dfrac{x^3}{3}+\dfrac{x^5}{5}$;

c) $\dfrac{\cos x}{\pi-2x}+16x\cos(xy)-2.$

Сохранить как *P2PR6*, *P2PR7*, *P2PR8*.

2. Вычислить арифметические выражения, при этом выбрать такую форму записи, чтобы количество «медленных» операций (умножение, деление) было сведено к минимуму.

3. Вычисления всех трех выражений произвести в одной программе:

$$\frac{3{,}18-4{,}98}{1{,}1-0{,}2}\cdot(1{,}7-0{,}24)+4{,}98;$$

$$\frac{3{,}18-4{,}98}{1{,}1-0{,}2}\cdot(0{,}72-0{,}24)+4{,}98;$$

$$\frac{3{,}18-4{,}98}{1{,}1-0{,}2}\cdot(0{,}89-0{,}24)+4{,}98.$$

Сохранить как *P2PR9*, *P2PR10*, *P2PR11*.

Составление программ с использованием основных операторов в Turbo Pascal

Задания первого уровня

Упражнение 1. Вычислить частное двух целых чисел. В связи с тем, что делить на ноль нельзя, организуем контроль ввода данных.

1. Наберите текст программы:

```
Program DEL;
var
  a, b: integer;          {Операнды — целые числа}
  result: real;           {Результат — вещественное число}
Begin
  write ('Введите значение делимого a: '); read(a);
  write ('Введите значение делимого b: '); read(b);
  if b=0                  {Условие выполнено}
    then writeln ('Неверные исходные данные: делитель - ноль')
                          {Условие не выполнено}
    else                  {Составной оператор нужен для объединения двух
                           команд в единое целое}
    begin                 {Начало составного оператора}
     result:=a/b;
     writeln ('Частное чисел', a, 'и ', b, ' = ', result:7:3);
    end                   {Конец составного оператора}
End.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9*.

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5*.

4. Сохраните программу на своем диске: *<F2> A:\P3PR1.*

Упражнение 2. Вычислить частное двух целых чисел. В связи с тем, что делить на ноль нельзя, организуем контроль ввода данных.

1. Наберите текст программы:

```
Program CHET;
var
  n: integer;
Begin
  write ('Введите целое число: ');
  readln(n);
  write ('Число ', n, ' - ');
  if n mod 2=0 then writeln ('четное') else writeln ('нечетное');
End.
```

Комментарий: для проверки на нечетность можно использовать функцию *odd*:

```
if odd(n) then writeln ('нечетное ') else writeln ('четное');
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9*.

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5*.

4. Сохраните программу на своем диске: *<F2> A:\P3PR2*.

Упражнение 3. Вывести на печать название дня недели, соответствующее заданному числу *D*, при условии, что в месяце 31 день и 1-е число — понедельник.

Для решения задачи воспользуемся операцией *mod*, позволяющей вычислить остаток от деления двух чисел, и условием, что 1-е число — понедельник.

Если в результате остаток от деления заданного числа *D* на 7 будет равен 1, то это понедельник; 2 — вторник; 3 — среда и так далее.

1. Наберите текст программы:

```
Program D_NED;
var
  D: byte;
Begin
  write ('Введите число D= ');
  readln (D);
  case D mod 7 of          {Вычисляется остаток от деления D на 7}
    (В зависимости от полученного значения на печать выводится название дня недели)
      1: writeln ('ПОНЕДЕЛЬНИК');
      2: writeln ('ВТОРНИК');
      3: writeln ('СРЕДА');
      4: writeln ('ЧЕТВЕРГ');
      5: writeln ('ПЯТНИЦА');
      6: writeln ('СУББОТА');
      0: writeln ('ВОСКРЕСЕНЬЕ')
      (Если результат управляющего выражения превышает 6, то выдается сообщение об ошибке)
  else writeln ('ОШИБКА ПРИ ВВОДЕ!!!')
      end;
End.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9*.

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5*.

4. Сохраните программу на своем диске: *<F2> A:\P3PR3*.

Самостоятельная работа

Задания второго уровня

1. Составить программу, которая сравнивает возраст брата и сестры и выводит соответствующее сообщение.

Записать программу под именем *P3PR4*.

2. Составить программу, проверяющую, принадлежит ли число, введенное с клавиатуры, интервалу (1; 5).

Сохранить программу под именем *P3PR5*.

3. По заданному номеру месяца m вывести на печать название времени года.

Сохранить программу под именем *P3PR6*.

4. Составить программу, которая выводит на экран различные приветствия в зависимости от введенного времени. Утро длится с 8 до 12 часов; день — с 12 до 17; вечер — с 17 до 23; ночь — все остальное время.

Задачу решить двумя способами: с помощью условного оператора и оператора выбора.

Сохранить программу под именем *P3PR7*.

5. Работник зарабатывает X рублей за 38 часов своей работы. Ему платят в 1,5 раза больше за каждый час сверх 38 часов. Какую сумму он получит, если отработает A часов?

Сохранить программу под именем *P3PR8*.

Задания третьего уровня

1. Составить программу, которая проверяет, может ли существовать треугольник с заданными сторонами. Известно, что сумма двух любых сторон должна быть больше третьей.

Сохранить программу под именем *P3PR9*.

2. Даны целые числа a, b, c. Если $a \le b \le c$, то все числа заменить наименьшим из них, в противном случае сменить знак каждого числа.

Сохранить программу под именем *P3PR10*.

3. Составить программу решения квадратного уравнения с использованием сложных условий.

Сохранить программу под именем *P3PR11*.

4. Составить программу, предназначенную для вычисления значения переменной y, где

$y = \sqrt{x} - 6$ при четных значениях x;

$y = x^2 - 6$ при значениях x, кратных 5;

$y = 0$ во всех остальных случаях.

Сохранить программу под именем *P3PR12*.

5. Написать программу, которая при вводе латинской прописной буквы выводит на экран такую же букву, но строчную.

Сохранить программу под именем *P3PR13.*

Указание: воспользуйтесь тем фактом, что все латинские прописные буквы расположены в кодовой таблице подряд, по алфавиту, начиная с символа ‘A’ с кодом 65.

Строчные буквы также расположены по алфавиту, начиная с символа ‘a’ с кодом 97.

Подсказка: для решения обратной задачи существует функция *UpCase*, которая преобразует строчные буквы латинского алфавита в прописные, но не изменяет другие, то есть

UpCase(‘p’)=’P’;
UpCase(‘P’)=’P’;
UpCase(‘+’)=’+’.

Применение циклов с параметром в Turbo Pascal

Задания первого уровня

Упражнение 1. Тест по таблице умножения: ученику задают ровно пять вопросов и в конце тестирования выставляют оценку по 5-балльной системе.

1. Наберите текст программы:

```
Program CIKL;
Var                              {Описание параметров цикла}
  i: integer;
  c: char;
  b: boolen;
begin                            {Вывод на печать целых чисел от 1 до 10}
  for i:=1 to 10 do writeln (i); {Вывод на печать целых чисел от 10 до -10}
  for i:=10 downto -10 do
  writeln (i);                   {Вывод на печать латинских символов от a до r}
                                 {Параметр цикла изменяется от a до r в алфавитном порядке}
  for c:='a' to 'r' do writeln (c);
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P4PR1.*

Упражнение 2. Тест по таблице умножения: ученику задают ровно пять вопросов и в конце тестирования выставляют оценку по 5-балльной системе.

1. Наберите текст программы:

```
Program TABL;
var
  s1, s2, otvet, k, prav: integer; {s1,s2—сомножители, otvet—ответ ученика, prav — прочий ответ }
begin
```

```
  randomize;                    {Инициализация датчика случайных чисел}
  clrscr;                       {Очистка экрана}
  for k:=1 to 5 do
  begin
   s1:= random(18)+2; s2:= random(18)+2  {s1,s2 — случайные числа в диа-
                                          пазоне от 2 до 19}
   write ('Сколько будет', s1, ' * ', s2, ' ? ');
   readln (otvet);
   if otvet=s1*s2 then
    begin
     write(' Правильно! ');
     prav:=prav+1;
    end
    else write(' Неверно... ');
    end;
    clrscr;                     {Очистка экрана}
    writeln(' Ваша оценка: ' , prav);
    readln
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P4PR2.*

Упражнение 3. Дано натуральное число *N.* Определить, является ли оно простым. Натуральное число *N* называется простым, если оно делится нацело без остатка только на 1 и *N.*

Число 13 — простое, так далее делится только на 1 и 13. *N* = 12 не является простым, так как делится на 1, 2, 3, 4, 6, 12.

Алгоритм решения этой задачи заключается в том, что число *N* делится на параметр цикла *i*, изменяющийся в диапазоне от 2 до *N*/2. Если среди значений параметра не найдется ни одного числа, делящего заданное число нацело, то *N* — простое число, иначе оно таковым не является.

1. Наберите текст программы:

```
Program PRCH;
Var
  N, i: integer;
  Pr: boolen;
begin
  writeln ('N= ');
  readln(N);
  Pr:=true;                     {Предположим, что число простое}
  for i:=2 to N div 2 do
    if N mod i=0 then           {Если найдется хотя бы один делитель, то}
    begin
```

```
        Pr:=false;              {число простым не является, и}
        break;                  {досрочный выход из цикла}
      end;
      if Pr then                {Проверка значения логического параметра и
                                вывод на печать соответствующего сообщения}
        writeln('Число ', N, ' - простое')
      else
        writeln('Число ', N, ' простым не является')
  end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9*.

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5*.

4. Сохраните программу на своем диске: *<F2> A:\P4PR2*.

Самостоятельная работа

Задания второго уровня

1. Вычислить a ($n > 0$), где a — вещественное число, которое необходимо возвести в целую положительную степень n.

Записать программу под именем *P4PR3*.

2. Вычислить сумму натуральных нечетных чисел, не превышающих n.

Сохранить программу под именем *P4PR4*.

3. Вычислить произведение натуральных чисел кратных 3 и не превышающих n.

Сохранить программу под именем *P4PR5*.

4. Вывести на экран в три столбца список чисел от 1 до N, их квадратов и кубов. Число N ввести с клавиатуры.

Сохранить программу под именем *P4PR6*.

5. Вычислить сумму S и произведение P всех целых чисел от $N1$ до $N2$. Пример: для $N1 = 3$, $N2 = 7$ получим $S = 25$, $P = 2520$.

Сохранить программу под именем *P4PR7*.

Задания третьего уровня

1. Написать программу вычисления выражения $(3 - x)(6 - x)(9 - x) \cdot \ldots \cdot (21 - x)$, где x — действительное число.

Сохранить программу под именем *P4PR8*.

2. Последовательно ввести N целых чисел. Найти минимальное и максимальное число из введенных чисел.

Сохранить программу под именем *P4PR9*.

3. Ввести последовательность из M элементов. Каждый элемент последовательности — цифра (то есть находится в диапазоне от 0 до 9). Сформировать число N, считая первый элемент последовательности младшим разрядом. Например, дана последовательность 5, 4, 3, 2, 1. Тогда десятичное число формируется следующим образом: 5 + 4 * 10 + 3 * 100 + 2 * 1000 + 1 * 10000 = =12345.

Сохранить программу под именем *P4PR10*.

Применение циклов с предусловием и постусловием в Turbo Pascal

Задания первого уровня

Упражнение 1. Вывести на экран таблицу значений функции. Вывод выполнить в два столбца:

✓ первый — значения аргумента;

✓ второй — значения функции при изменении аргумента от значения *a* до *b* с шагом *dx*.

$$f(x) = \lg(30) + x\sqrt{5\sin\left(\frac{\pi x}{3}\right)}.$$

Sl1. Наберите текст программы:

```
Program FUNC;
uses crt;
var
  x, y, z, lg3, a, b, dx: real;
begin
     clrscr;                          {Очистка экрана}
     write ('Введите начальное значение аргумента: '); readln(a);
     write ('Введите конечное значение аргумента: '); readln(b);
     write ('Введите шаг табулирования: '); readln(dx);
     writeln ('———————————————— ':20);
     write ('x':9, '|':4, 'y':4);
     writeln ('————————————————':20);
     lg3:=ln(3.0)/ln(10.0);       {Вычисление lg(3)}
     x:=a;
     while x<(b+dx/2) do
      begin
       z:=sin((pi*x)/3;
       if (z<0) then writeln(x:10:3, ' | функция не определена':22)
       else
        begin
          y:=lg3+x*sqrt(5.0*z);
          writeln(x:10:3, ' | ', y:7:3);
     end;
        x:=x+dx;
     end;
   writeln ('————————————————':20)
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9*.

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5*.

4. Сохраните программу на своем диске: *<F2> A:\P5PR1*.

Упражнение 2. Тест по таблице умножения, которая уже приводилась в качестве примера. В результате использования цикла получается полноценная тестирующая программа, которая также будет выводить и результаты тестирования.

Обратите внимание, что команда инициализации датчика случайных чисел выполняется до начала цикла, так как она должна быть выполнена только один раз. Инструкции для вывода результатов теста выполняются после выхода из цикла.

1. Наберите текст программы:

```
Program TABL_1;
uses crt;
var
  s1, s2, otvet, kol, prav: integer; yn: char;      {s1,s2 — сомножители, otvet — ответ ученика, prav — прочий ответ}
begin
  randomize;                                        {Инициализация датчика случайных чисел}
  clrscr;                                           {Очистка экрана}
  repeat
    kol:=kol+1;
    s1:= random(18)+2; s2:= random(18)+2;           {s1,s2 — случайные числа в диапазоне от 2 до 19}
    write ('Сколько будет', s1, ' * ', s2, ' ? ');
    readln (otvet);
    if otvet=s1*s2 then
      begin
        write(' Правильно! '); prav:=prav+1;
      end
    else write(' Неверно... ');
    write(' Продолжим тест? (Y/N) '); readln(yn);
  until (yn='n') or (yn='N');
  clrscr;                                           {Очистка экрана}
  writeln(' Результаты теста: ');
  writeln(' Задано вопросов: ', kol, '. Правильных ответов: ', prav, '.');
  readln
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P5PR2.*

Самостоятельная работа

Задания второго уровня

1. Изменить программу из упражнения 2 таким образом, чтобы при необходимости можно отказаться от выполнения теста и корректно выйти из программы (используйте оператор *WHILE..DO*).

Записать программу под именем *P5PR3.*

2. Вычислить сумму натуральных четных чисел, не превышающих *N*. Программу составить двумя способами: используя цикл с предусловием и цикл с постусловием.

Сохранить программу под именами *P5PR4* и *P5PR5.*

3. В результате выполнения программы должны быть выведены значения функции $y = x^3 + 2x$ для значений x, лежащих в диапазоне [1; 3], с шагом 0,2.

Программу составьте двумя способами: используя цикл с предусловием и цикл с постусловием.

Сохранить программу под именами *P5PR6* и *P5PR6.*

Задания третьего уровня

1. Дано действительное число A. Найти первое значение числа N, при котором сумма $S = 1 + {}^1/_2 + {}^1/_3 + ... + {}^1/_N$ превышает A. Программу составить двумя способами: используя цикл с предусловием и цикл с постусловием.

Сохранить программу под именами *P5PR7* и *P5PR8.*

2. Составить программу, которая имитирует работу арифметического калькулятора. Организовать ввод первого и второго операндов и знака операции.

Сохранить программу под именем *P5PR4.*

3. Составить программу, которая производит суммирование произвольного количества целых чисел, вводимых с клавиатуры. Концом последовательности служит ввод отрицательного числа. Программу составить двумя способами: используя цикл с предусловием и цикл с постусловием.

Сохранить программы под именами *P5PR5* и *P5PR6.*

4. Составить программы для нахождения сумм приведенных ниже рядов и проверьте их работоспособность на компьютере:

1) $\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + K.$

Замечание: при вычислении суммы степенного ряда sin(x) можно использовать следующие выкладки: пусть S_k - значение k-го слагаемого, причем значение $S_0 = x$. Тогда выполняется следующее соотношение:

$$S_{k+1} = S_k \cdot \frac{(-x^2)}{2k(2k+1)}.$$

2) $\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + K,$

3) $\exp(x) = 1 + \frac{x}{1!} + \frac{x^2}{2!} + K.$

Работа с массивами в Turbo Pascal

Задания первого уровня

Упражнение 1. Найти в массиве элементы, равные числу, заданному пользователем. Подсчитать их количество и вывести номер первого найденного элемента. Массив задать при помощи ввода с клавиатуры.

1. Наберите текст программы:

```
Program MAS_1;
const count=10;
var n,                  {Число для поиска}
    a,                  {Номер первого элемента}
    b,                  {Количество элементов}
```

```
      i: integer;
      m: array [1..count] of integer;
begin
     writeln('Ввод исходного массива: ');
     for i:=1 to count do
     begin
     write('элемент №', i, ': ');
     readln(m[i]);
  end;
a:=0; b:=0;
write('Введите число для поиска >'); readln(n);
for i:=1 to count do        {Поиск элемента, равного n}
  if m[i]=n then
   begin
     if b=0 then a:=i;      {Запомним номер первого элемента, равного n}
     b:=b+1;                {Увеличить число найденных элементов на 1}
  end;
if b=0 then writeln('Нет таких элементов в массиве')
     else
     begin
     writeln('Количество элементов массива, имеющих значение ', n, ' =', b:3);
     writeln('Первый элемент имеет номер ', a:3);
     end
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P6PR1.*

Упражнение 2. Сформировать матрицу случайных чисел и транспонировать ее. При транспонировании элементы матрицы переставить таким образом, что строки исходной матрицы становятся столбцами транспонированной матрицы. При этом элементы, расположенные на главной диагонали исходной и транспонированной матриц, одни и те же.

Операция транспонирования сводится к обмену элементов матрицы, расположенных симметрично относительно главной диагонали.

1. Наберите текст программы:

```
Program MAS_2;
const row=3; col=row;
var a: array [1..row, 1..col] of integer;
     i, j, buf: integer;
begin
  randomize;                  {Инициализация датчика случайных чисел}
  writeln('Исходная матрица случайных чисел: ');
  for i:=1 to row do
   begin
```

```
      for j:=1 to col do
        begin
          a[i, j]:=random(100);      {Случайное значение элемента}
          write(a[i, j]:4);          {Вывод элемента массива на экран}
    end;
    writeln;
end;                                 {Транспонирование матрицы}
      for i:=1 to row do             {Просмотр всех строк матрицы}
                                     {Просмотр элементов в строке, расположенных
                                     выше главной диагонали}
      for j:=1 to col do
        begin
          buf:=a[i, j]; a[i, j]=a[j, i]; a[j, i]=buf;
      end;
 writeln('Результат транспонирования матрицы: ');
 for i:=1 to row do                  {Обмен элементов, симметричных относитель-
                                     но главной диагонали}
      begin
        for j:=1 to col do write(a[i, j]:4);
        writeln;
      end
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9*.

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5*.

4. Сохраните программу на своем диске: *<F2> A:\P6PR2*.

Самостоятельная работа

Задания второго уровня

1. В одномерном массиве выполнить поиск максимального элемента массива, а затем вывести на экран его значение и порядковый номер в массиве. Массив задается с помощью датчика случайных чисел.

Записать программу под именем *P6PR3*.

2. Вывести на экран сумму четных элементов массива из 10 чисел. Массив заполняется случайными числами. Записать программу под именем *P6PR4*.

3. Переписать элементы массива *a* в массив *b* в обратном порядке. Массив *a* заполняется случайными числами.

Сохранить программу под именем *P6PR5*.

4. Задана матрица A размера 5×7. Вывести ее на экран так, чтобы каждая строка матрицы выводилась на экран с новой строки.

Сохранить программу под именем *P6PR6*.

Задания третьего уровня

1. Переписать элементы массива *a* в массив *b*, удвоив все элементы меньшие первого элемента, остальные переписать без изменения. Массив *a* заполняется случайными числами.

Сохранить программу под именем *P6PR7*.

2. Транспонировать целочисленную матрицу 5×5. Массив задается при помощи ввода с клавиатуры.

Сохранить программу под именем *P6PR8*.

3. Составить программу, которая вводит с клавиатуры квадратный массив целых чисел и формирует два вектора. В первый записываются элементы исходного массива, расположенные на главной диагонали и выше, а во второй — элементы, лежащие ниже главной диагонали. Предусмотреть вывод на экран.

Сохранить программу под именем *P6PR9*.

4. В каждой строке матрицы *C* размера $M \times N$ найти минимальный элемент и поменять его с первым элементом в этой строке.

Сохранить программу под именем *P6PR10*.

5. Составить программу вычисления среднего арифметического элементов в каждом столбце матрицы *B* размера $N \times N$ и записать полученные значения в главную диагональ этой матрицы.

Сохранить программу под именем *P6PR11*.

Использование процедур и функций в Turbo Pascal

Задания первого уровня

Упражнение 1. Задан массив целых чисел. Необходимо удалить из него все совершенные числа.

Совершенное число представляет собой сумму всех своих делителей меньших его самого. Для решения поставленной задачи необходимо проверить каждый элемент массива. Если он представляет собой совершенное число, удалить его (после удаления остается элемент с тем же номером, однако это уже следующий элемент). В противном случае надо перейти к следующему элементу.

В программе будут использоваться две подпрограммы: функция *Sover*, которая проверяет, является ли число совершенным, и процедура *Udal* — удаление одного элемента из массива.

1. Наберите текст программы:

```
Program MAS_3;
type
{Тип данных massiv будет использоваться при описании процедуры удаления
i-го элемента}
   massiv=array[1..1000] of longint;

{Описание функции, проверяющей, является ли число совершенным}
Function Sover (P: longint):boolean;
var sum, m: longint;
begin
   if p>1 then
   begin              {Нахождение суммы общих делителей}
    sum:=1;           {В начале сумма равна 1, так как любое число делится на 1}
    for m:=2 to p div 2 do
```

```
{Делителем числа может быть любое число от 2 до половины самого себя}
    if p mod m=0 then sum:= sum+m;    {Если число m — делитель, то добавляем его к сумме}
{Если сумма делителей = самому числу p, то функция возвращает значение true,}
    if sum=p then Sover:=true
    else Sover:=false      {иначе - false}
end;
{Числа, меньшие или равные 1, не являются совершенными, поэтому ф-ция возвращает значение false}
   else Sover:=false
end;

{Описание процедуры удаления i-го элемента из массива x}
Procedure Udal (var x: massiv; i: word; var N: word);
var j: word;
begin
  for j:=i to N-1 do {Сдвиг массива, начиная с i—го, влево на один элемент}
  x[j]:=x[j+1];
  N:=N-1;            {После удаления элемента его размер становится на один меньше}
end;

var x: massiv;
    i, n: word;
    L: boolean;
begin
    write ('n='); readln (n);
    writeln ('Массив X');
    for i:=1 to n do read (x[i]);
    i:=1;            {Просмотр массива начинаем с первого элемента}
    while (i<=n) do {Проверяем, не достигнут ли конец массива}
    begin
{Обращение к функции Sover, которая проверяет, является ли элемент массива x[i] совершенным;}
                     {результат записывается в логическую переменную L}
    L:=Sover (x[i]);
{Если число совершенное,то удаляем его из массива. При этом текущим остается элемент с номером i}
    {но после удаления там хранится другой элемент}
    if L then Udal (x, i, n)
    else i:=i+1
{Если число не является совершенным, то переходим к следующему элементу массива}
    end;
    writeln ('Преобразованный массив X');
    for i:=1 to n do
    write (x[i], ' ');
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P7PR1.*

Задание второго уровня

1. Разработать процедуры и функции для решения следующих задач обработки массивов (*по вариантам*):

a) посчитать количество отрицательных элементов одномерного массива вещественных чисел.

Записать программу под именем *P7PR2*;

b) посчитать сумму квадратов положительных элементов одномерного массива вещественных чисел.

Сохранить программу под именем *P7PR3*;

c) посчитать сумму квадратов отрицательных элементов одномерного массива вещественных чисел.

Сохранить программу под именем *P7PR4*;

d) преобразовать одномерный массив вещественных чисел, присвоив каждому элементу квадрат его значения.

Сохранить программу под именем *P7PR5*;

e) преобразовать одномерный массив вещественных чисел, уменьшив каждый элемент на абсолютную величину среднего значения элементов массива.

Сохранить программу под именем *P7PR6*;

f) преобразовать одномерный массив вещественных чисел, занеся в каждый элемент сумму всех предыдущих элементов (в первый элемент при этом необходимо поместить значение 0).

Сохранить программу под именем *P7PR7*;

g) посчитать сумму квадратов диагональных элементов двумерного массива вещественных чисел.

Сохранить программу под именем *P7PR8*;

h) посчитать максимальную сумму элементов в строках двумерного массива вещественных чисел.

Сохранить программу под именем *P7PR9*;

i) посчитать минимальную сумму элементов в столбцах двумерного массива вещественных чисел.

Сохранить программу под именем *P7PR10*;

j) преобразовать двумерный массив вещественных чисел, занеся значение 0 во все элементы с двумя четными индексами.

Сохранить программу под именем *P7PR11.*

Задания третьего уровня

Функции

1. Даны четыре числовых массива *f1*, *f2*, *f3*, *f4.* Вывести имя тех массивов, которые имеют наибольшее число элементов (если таких элементов более одного, то вывести имя одного из них). Описать в программе функцию *l(f)*, значение которой равно количеству элементов массива.

Сохранить программу под именем *P7PR12.*

2. Даны действительные числа *a, b, c*.
Получить:

$$\frac{\max(a, a+b) + \max(a, b+c)}{1 + \max(a+bc, 1.5)}.$$

Сохранить программу под именем *P7PR13*.
3. Дано: натуральное n, действительные $a_1, \ldots, a_{3n}$.
Получить $x + y^2 + z^3$, где
$x = a_1 \cdot a_2 \cdot \ldots \cdot an$,
$y = a_{n+1} \cdot a_{n+2} \cdot \ldots \cdot a_{2n}$,
$y = a_{2n+1} \cdot a_{2n+2} \cdot \ldots \cdot a_{3n}$
Сохранить программу под именем *P7PR14*.

Реализация алгоритмов сортировки

Задания первого уровня

Упражнение 1. Программа с процедурами сортировки простого выбора и простой вставки.

1. Наберите текст программы:

```
Uses Crt;
Const n=10;
Type
    Mas = Array [1 .. n] Of Integer;
Var A: Mas;

Procedure SetRandomMas ( Var A: Mas );      {Задание случайного массива}
Var i: Integer;
Begin
    Randomize;
    For i := 1 To n Do A[i] := Random (100);
End;

Procedure OutPutMas ( Var A: Mas );         {Вывод массива на экран}
Var i: Integer;
Begin
    For i := 1 To n Do Write( A[i]:3 );
    WriteLn;
End;

Procedure SortVybor ( Var A: Mas );         {Сортировка выбором}
Var i, k, m, j,Temp, Min: Integer;
Begin
    For i := 1 To n Do
    Begin
     Min := A[i];
     k := i;
     For j := i+1 To n Do
```

```
            If A[j] < Min Then Begin Min := A[j]; k := j; End;
            Temp := Min;
            For m := k-1 DownTo i Do A[m+1] := A[m];
            A[i] := Temp;
        End;
    End;

    Procedure SortVstav ( Var A: Mas );      {Сортировка вставкой}
    Var i, k, j, Temp: Integer;
    Begin
        For i := 1 To n-1 Do
        Begin
            If A[i+1] < A[i] Then
             Begin
                j := i;
                While (A[j] > A[i+1]) And (j >0) Do j := j — 1;
                k := j + 1;
                Temp := A[i+1];
                For j := i DownTo k Do A[j+1] := A[j];
                A[k] := Temp;
              End;
        End;
    End;

    Begin
        ClrScr;
        SetRandomMas(A);
        OutPutMas(A);
        SortVstav(A);
        OutPutMas(A)
    End.
```

2. Запустите полученную программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P8PR1.*

Самостоятельная работа

Задания второго уровня

1. Написать программу, реализующую алгоритм пузырьковой сортировки, причем просмотр элементов должен вестись наоборот (справа налево).

Записать программу под именем *P8PR2.*

2. Написать процедуру сортировки простым выбором, в которой выбирается не минимальный, а максимальный элемент.

Сохранить программу под именем *P8PR3.*

Задания третьего уровня

1. Запрограммировать сортировку выбором в виде процедуры. Поиск наименьшего числа сделайте ее внутренней функцией.

Сохранить программу под именем *P8PR4.*

2. Написать свои программы сортировки массива для изученных алгоритмов сортировки.

Сохранить программы.

Составление программ с использованием множеств в Turbo Pascal

Задания первого уровня

Упражнение 1. Пусть дана строка символов с точкой в конце строки. Необходимо определить число различных букв, входящих в данную строку.

1. Наберите текст программы:

```
Program Mn1;
Var
M: Set Of Char;
Str: String;
c: Char;
i, n: Integer;
Begin
    M:=[];              { M — пустое множество}
    n:=0;               {Переменная, считающая количество различных букв
                        в строке}
    WriteLn ('Введите строку');
    ReadLn (Str); {ввод строки}
    For i:=1 To Length (Str) Do
    If not (Str[i] In M) Then
    Begin
      M:=M+[Str[i]];    {Формирование множества, содержащего все буквы,
                        входящие в строку}
      n:=n+1;
    End;
    WriteLn ('Количество различных элементов в строке равно', n)
End.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P9PR1.*

Упражнение 2. Даны две символьные строки, содержащие только строчные латинские буквы. Построить строку *S*3, в которую войдут только общие символы *S*1 и *S*2 в алфавитном порядке и без повторений.

1. Наберите текст программы:

```
Program P2;
Type Mset=Set of 'a'..'z';
```

```
Var S1, S2, S3: String;
    MS1, MS2, MS3 : Mset;
    C: Char;
Procedure SM (S: String; Var MS: Mset);
{Процедуры формируют множество MS, содержащее все символы строки S }
Var I Byte;
Begin MS:=[];
 For I:=1 To Length (S) Do
 MS:=MS+[S[I]]
End;
Begin {Ввод исходных строк}
 ReadLn (S1); ReadLn (S2);
{Формирование множеств MS1 и MS2 из символов строк S1 и S2}
 SM (S1, MS1); SM (S2, MS2);
{Пересечение множеств — выделение общих элементов в множество MS3}
 MS3:=MS1*MS2;
{Формирование результирующей строки S3}
 S3:='';
 For C:='a' To 'z' Do
 If C In MS3 Then S3:=S3+C;
 WriteLn ('Результат:'S3)
End.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9*.

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5*.

4. Сохраните программу на своем диске: *<F2> A:\P9PR2*.

Самостоятельная работа

Задания второго уровня

1. Составить программу, демонстрирующую операции над множествами: операции объединения, разности, пересечения. Множества чисел заполнить следующим образом:

a) *D* 1 — четными числами 2, 4, 6, 8;

b) множество *D* 2 — числами 0, 1, 2, 3, 5;

c) множество *D* 3 — нечетными числами 1, 3, 5, 7, 9.

Сохранить программу под именем *P9PR3*.

2. Ввести строку. Подсчитать количество гласных букв в ней.

Записать программу под именем *P9PR4*.

Задания третьего уровня

1. Задано некоторое множество *M* и множество *T* того же типа. Подсчитать, сколько элементов из множеств *T* и *M* совпадает.

Сохранить программу под именем *P9PR5*.

2. Требуется сформировать последовательность натуральных чисел от 1 до *n*, расположенных в случайном порядке без повторения значений.

Сохранить программу под именем *P9PR6*.

Работа с файлами в Turbo Pascal

Задания первого уровня

Упражнение 1. Программа, в результате выполнения которой выводятся все четные числа из данного файла *int* с целочисленными компонентами.

1. Наберите текст программы:

```
Program rem (input, output);
type v=file of integer;
var int: v; i: integer;
begin reset (int);
while not eof (int) do
  begin
    read (int, i);
    if i mod 2=0 then writeln (i)
  end
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P10PR1.*

Упражнение 2. Программа, которая формирует типизированный файл из целых чисел, вводимый с клавиатуры. Их количество заранее неизвестно. Признаком конца ввода является 0.

Программа находит сумму и произведение чисел из файла, разность между предпоследним и вторым по счету числами, наибольшее из чисел.

1. Наберите текст программы:

```
Program rabf
type file_type=file of integer;
var f: file_type;
  sum, mult, r, k1, k2, max: integer;
begin
 writeln ('Введите элементы файла, окончание - 0');
 {Запись элементов в файл}
 assign (f, 'data.dat'); rewrite (f);
 repeat
  readln (r); if r<>0 then write (f, r);
 until r=0;
 {Вычисление результатов}
 seek (f, 0); sum:=0; mult:=1;
 read (f, r); max:=r; seek (f, filepos(f)-1);
 while not eof (f) do
  begin
   read (f, r); sum:=sum+r; mult:=mult*r;
   {Поиск максимального элемента}
   if max<r then max:=r;
 end;
```

```
seek (f, 1); read (f, k1);                    {Чтение второго компонента}
seek (f, filesize (f)-2); read (f, k2);       { Чтение предпоследнего компонента}
    close (f);
{Вывод результатов}
 writeln ('Сумма = ', sum, #10#13'Произведение = ', mult);
writeln ('Разность = ', k2-k1, #10#13'Максимум = ', max);
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P10PR2.*

Задание второго уровня

1. Создать три программы:

а) считывает 10 действительных чисел с клавиатуры и записывает их в файл.

Сохранить программу под именем *P10PR3*;

б) добавляет в этот файл еще 5 чисел.

Сохранить программу под именем *P10PR4*;

в) читает этот файл и выводит числа, содержащиеся в нем на экран.

Сохранить программу под именем *P10PR5.*

Задания третьего уровня

1. Написать программу, в результате выполнения которой выводятся все числа кратные трем из данного файла *int* с целочисленными компонентами**.**

Сохранить программу под именем *P10PR6.*

2. Дан файл *Input.txt*, элементы которого являются целыми числами. Получить в файле *Output.txt* все элементы файла *Input.txt*, которые делятся на 3 и не делятся на 7.

Сохранить программу под именем *P10PR7.*

Составление программ с использованием записей в Turbo Pascal

Задания первого уровня

Упражнение 1. О каждом студенте известна следующая информация:

- ✓ фамилия, инициалы;
- ✓ год рождения;
- ✓ группа;
- ✓ отметка по математике;
- ✓ отметка по истории;
- ✓ отметка по ВТ;
- ✓ отметка по статистике.

Сформировать таблицу, записав в нее известную информацию о каждом студенте и его средний балл. Подсчитать средний балл каждого предмета, вывести таблицу на экран в алфавитном порядке.

1. Наберите текст программы:

```
Program stud;
Type  tablica=record                    {Описание записи о каждом студенте}
        name: string[15];
        group: string[8];
        god: integer;
        vt, history, stat, math: byte;
        sr_bal: real;
    end;
var i, j, n :integer; a: tablica;
     mas:array [1..30] of tablica;      {Таблица - массив записей}
s_vt, s_history, s_stat, s_math: real; {Переменные для хранения средних значений по предметам}
begin
write('n='); readln(n);                 {Ввод количества записей}
for i=1 to n do                         {Ввод элементов массив записей}
  with mas[i] do
  begin
    writeln('i=',i:4);
    writeln('FIO');
    readln(name);
    write('Group');
    readln(group);
    write('Year');
    readln(god);
    write('Otsenki');
    readln(vt, history, stat, math);
    sr_bal:=(vt+history+stat+math)/4;
  end;
s_vt:=0; s_history:=0; s_stat:=0; s_math:=0;
for i=1 to n do                         {Вычисление среднего балла по каждому предмету}
begin
  s_vt:=s_vt+mas[i].vt;
  s_history:=s_history+mas[i].history;
  s_stat:=s_stat+mas[i].stat;
  s_math:=s_math+mas[i].math;
end;
for i=1 to n do                         {Упорядочение записей массива в алфавитном порядке фамилий}
 for j:=1 to n-1 do
  if mas[j].name> mas[j+1].name then
  begin
    a:=mas[j];
    mas[j]:=mas[j+1];
    mas[j+1]:=a;
  end;
clrscr;
```

```
write(' ';4, 'FIO ',' ';4); {Вывод результатов}
write(' ';2, ' GROUP ',' ';2);
write(' ';5, ' OTSENKI ',' ';5);
writeln('Sr/ Bal ');
for i=1 to n do
  with mas[i] do
  begin
    write(name:15);
    write(' ',group:8);
    write((' ',god:4);
    writeln(' ',vt:3, ' ',history:3, ' ',stat:3, ' ',math:3, ' ',sr_bal:5:2);
  end;
 writeln(' Sr. Bal: ', ' ', s_vt:3:1, ' ', s_history:3:1, ' ', s_stat:3:1, ' ', s_math:3:1);
end.
```

2. Запустите программу на выполнение и проверьте ее работу: *Ctrl-F9.*

3. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

4. Сохраните программу на своем диске: *<F2> A:\P11PR1.*

Самостоятельная работа

Задания второго уровня

1. Создать массив записей, содержащий сведения об альбомах различных групп:

- ✓ название группы;
- ✓ название альбома;
- ✓ год выпуска;
- ✓ стиль.

Вывести на экран все альбомы в стиле *pop* и *rock.*

Сохранить программу под именем *P11PR2.*

2. О сотрудниках некоторого предприятия известно следующее:

- ✓ фамилия, инициалы;
- ✓ год рождения;
- ✓ должность;
- ✓ пол.

Сформировать таблицу, записав в нее известную информацию о каждом сотруднике. Вывести на печать информацию о количестве пенсионеров — женщин старше 55 лет и мужчин старше 60 лет.

Сохранить программу под именем *P11PR3.*

Задания третьего уровня

1. Известна информация о сотрудниках некоторого предприятия:

- ✓ фамилия, инициалы;
- ✓ год рождения;
- ✓ должность;
- ✓ стаж;
- ✓ оклад.

Сформировать таблицу, записав в нее известную информацию о каждом сотруднике. Создать поле «Зарплата», добавляя 10 % к окладу, если стаж работы более 10 лет, и 15 %, если более 20. Отсортировать таблицу в алфавитном порядке.

Сохранить программу под именем *P11PR4*.

2. Создать массив записей, содержащий информацию о машинах:

✓ марка;
✓ год выпуска;
✓ цвет;
✓ номер.

Вывести на экран информацию о машинах черного цвета. Вывести на экран марки машин, номера которых начинаются на букву *M*. Упорядочить записи в массиве по возрастанию года выпуска.

Сохранить программу под именем *P11PR4*.

Использование динамических переменных в Turbo Pascal

Задания первого уровня

Упражнение 1. Найти максимальный и минимальный элементы массива x(n).

1. Наберите текст программы.

Вариант I

```
Program din_mas1;
type massiw=array [1..150] of real;
  var x:^massiw;
  i, n: integer; max, min: real;
begin
  new(x);                                  {Выделяем память под динамический
                                           массив из 150 вещественных чисел}
  writeln ('Введите размер массива'); readln(n);
  for i:=1 to N do
  begin
    write ('x(', i, ')='); readln(x^[i]);
end;
max:=x^[1]; min:=x^[1];
for i:=2 to N do
begin
  if x^[i] > max then max:=x^[i];
  if x^[i] < min then min:=x^[i];
end;
writeln ('максимум= ', max:1:4, 'минимум= ', min:1:4);
dispose(x);                                {Освобождаем память}
end.
```

Вариант II

```
Program din_mas2;
type massiw=array [1..150] of real;
```

```
    var x:^massiw;
    i, n: integer; max, min: real;
begin
    writeln ('Введите размер массива'); readln(n);
    getmem (x, n*sizeof(real));        {Выделяем память под n элементов массива}
    for i:=1 to N do
    begin
    write ('x(', i, ')=”); readln(x^[i]);
    end;
    max:=x^[1]; min:=x^[1];
    for i:=2 to N do
    begin
    if x^[i] > max then max:=x^[i];
    if x^[i] < min then min:=x^[i];
    end;
    writeln ('максимум= ', max:1:4, 'минимум= ', min:1:4);
    freemem (x, n*sizeof(real)); {Освобождаем память}
end.
```

2. Проанализируйте работу программ.

3. Запустите программы на выполнение и проверьте их работу: *Ctrl-F9.*

4. Для просмотра результатов выполненных программ необходимо нажать *Alt-F5.*

5. Сохраните программы на своем диске: *<F2> A:\P12PR1* и *A:\P12PR2* соответственно.

Самостоятельная работа

Задания второго уровня

1. Записать все элементы массива $X = (x_1, x_2, ..., x_n)$, соответствующие условию $x_i \in [1,2]$, подряд в массив $Y = (y_1, y_2, ..., y_n)$. Определить минимальный элемент массива *X.*

Записать программу под именем *P12PR3.*

2. Переписать элементы массива целых чисел $X = (x_1, x_2, ..., x_n)$ в обратном порядке в массив $Y = (y_1, y_2, ..., y_n)$. Вычислить количество четных, нечетных и нулевых элементов массива *Y.*

Сохранить программу под именем *P12PR4.*

Задания третьего уровня

1. Во входном файле расположена последовательность целых чисел. Определить, имеются ли среди этих чисел два совпадающих.

Сохранить программу под именем *P12PR5.*

2. Во входном файле расположен символ, за которым следуют двадцать различных целых чисел, если этот символ есть *i*, или тридцать различных действительных чисел, если этот символ есть *r.* Написать программу, в результате выполнения которой выводится часть данной последовательности чисел, начиная с первого по порядку и заканчивая минимальным из данных.

Сохранить программу под именем *P12PR6.*

Составление программ с использованием модулей в Turbo Pascal

Задание первого уровня

Упражнение 1. Реализовать в виде модуля набор подпрограмм для выполнения следующих операций над обыкновенными дробями вида P/Q (P — целое, Q — натуральное):

1) сложение;
2) вычитание;
3) умножение;
4) деление;
5) сокращение дроби;
6) возведение дроби в степень N (N — натуральное);
7) функции, реализующие операции отношения (равно, не равно, больше или равно, меньше или равно, больше, меньше).

Дробь представить следующим типом:

```
type Frac=record
P: integer;
Q: 1..32767
end;
```

Используя этот модуль, решить задачи:

А. Дан массив A, элементы которого — обыкновенные дроби. Найти сумму всех элементов и их среднее арифметическое; результаты представить в виде несократимых дробей.

Б. Дан массив A, элементы которого — обыкновенные дроби. Отсортировать его в порядке возрастания.

1. Наберите текст программы:

```
Unit Droby;
Interface
    type
    Natur=1..High (longint);
    Frac=record
    P: longint;
    Q: Natur
    end;
Procedure Sokr (var A: Frac);
Procedure Summa (A, B: Frac; var C: Frac);
Procedure Raznost (A, B: Frac; var C: Frac);
Procedure Proizvedenie (A, B: Frac; var C: Frac);
Procedure Chastnoe (A, B: Frac; var C: Frac);
Procedure Stepen (A: Frac; N: Natur; var C: Frac);
Function Menshe (A, B: Frac): boolean;
Function Bolshe (A, B: Frac): boolean;
Function Ravno (A, B: Frac): boolean;
Function MensheRavno (A, B: Frac): boolean;
Function BolsheRavno (A, B: Frac): boolean;
Function NeRavno (A, B: Frac): boolean;
Implementation{Раздел реализации модуля}
```

```
                                        {Наибольший общий делитель двух чисел — вспомогательная функция, ранее необъявленная}
Function NodEvklid (A, B: Natur): Natur;
begin
 while A <> B do
   if A > B then
    if A mod B <> 0 then A:=A mod B
                    else A:=B
    else
    if B mod A <> 0 then B:=B mod A
                    else B:=A;
  NodEvklid:=A
end;

Procedure Sokr;                         {Сокращение дроби}
var M, N: Natur;
begin
  if A.P <> 0 then
  begin
    if A.P < 0 then M:=abs(A.P)
    else M:=A.P;                        {Совмещение типов, так как
                                         A.P - longint}
    N:= NodEvklid(M, A.Q);
    A.P:=A.P div N;
    A.Q:=A.Q div N
  end
end;

Procedure Summa;                        {Сумма дробей}
begin
    C.Q:=(A.Q*B.Q) div NodEvklid(A.Q, B.Q);   {Знаменатель дроби}
    C.P:=A.P*C.Q div A.Q + B.P*C.Q div B.Q;   {Числитель дроби}
    Sokr (C)
end;

Procedure Raznost;                      {Разность дробей}
begin
    C.Q:=(A.Q*B.Q) div NodEvklid(A.Q, B.Q);   {Знаменатель дроби}
    C.P:=A.P*C.Q div A.Q - B.P*C.Q div B.Q;   {Числитель дроби}
    Sokr (C)
end;

Procedure Proizvedenie;                 {Умножение дробей}
begin
    C.Q:= A.Q*B.Q;                      {Знаменатель дроби}
    C.P:=A.P*B.P;                       {Числитель дроби}
    Sokr (C)
end;
```

```
Procedure Chastnoe;              {Деление дробей}
begin
    C.Q:= A.Q*B.P;               {Знаменатель дроби}
    C.P:=A.P*B.Q;                {Числитель дроби}
    Sokr (C)
end;
Procedure Stepen;                {Возведение дроби в степень}
var I: Natur;
begin
    C.Q:= 1;
    C.P:=1;
    Sokr (A);
    For I:=1 to N do
        Proizvedenie (A, C, C)
end;
Function Menshe;                 {отношение '<' между дробями}
begin
    Menshe:=A.P*B.Q < A.Q*B.P
end;
Function Bolshe;                 {отношение '>' между дробями}
begin
    Bolshe:=A.P*B.Q > A.Q*B.P
end;
Function Ravno;                  {отношение '=' между дробями}
begin
    Ravno:=A.P*B.Q = A.Q*B.P
end;
Function BolsheRavno;            {отношение '>=' между дробями}
begin
    BolsheRavno:= Bolshe(A, B) or Ravno(A, B)
end;
Function MensheRavno;            {отношение '<=' между дробями}
begin
    MensheRavno:= Menshe(A, B) or Ravno(A, B)
end;
Function NeRavno;                {отношение '<>' между дробями}
begin
    NeRavno:= Not Ravno(A, B)
end;
                                 {Раздел инициализации модуля}
begin
end.
```

Теперь подключаем модуль к программе, в которой выполняем суммирование массива дробей:

```
Program Sum;
```

```
Uses Droby;
var A: array [1..100] of Frac;
    I, N: integer;
    S: Frac;
begin
    write ('Введите количество элементов массива:');
    readln (N);
    S.P:=0; S.Q:=1;      {Первоначально сумма равна нулю}
    for I:=1 to N do     {Вводим и суммируем дроби}
    begin
    write ('Введите числитель', I, '-й дроби:');
    readln (A[I].P);
    write ("Введите знаменатель', I, '-й дроби:');
    readln (A[I].Q);
    Summa (A[I], S, S);
    end;
    write ('Ответ: ', S.P, '/', S.Q)
end.
```

2. Проанализируйте работу программ

Запустите программы на выполнение и проверьте их работу: *Ctrl-F9.*

4. Для просмотра результатов выполненной программы необходимо нажать *Alt-F5.*

5. Сохраните программу на своем диске *<F2> A:\P13PR1.*

Самостоятельная работа

Задание второго уровня

1. Вторую задачу из рассмотренного выше примера решить самостоятельно.

Записать программу под именем *P13PR2.*

Задания третьего уровня

1. Найти самый большой файл с расширением *.pas* в заданном каталоге.

Записать программу под именем *P13PR3.*

2. Проверить, правда ли что 28 июня, 5 июля, 13 сентября и 13 декабря попадают на один день недели. Если да, то определить, какой это будет день недели в году.

Сохранить программу под именем *P13PR4.*

Для проверки уровня освоения обучающимися основного программного материала предлагаются следующие контрольные, самостоятельные и тестовые работы.

Контрольная работа по теме «Типы данных»

Вариант I

1. Записать константы с фиксированной точкой и с плавающей точкой:
a) $0{,}5 \cdot 10^{-12}$
b) $-330 \cdot 10^{8}$
c) $-0{,}0000016$.

2. Записать константы через математическую запись:
a) -2.57E4
b) 3.234E6
c) 6.345E-7.

3. Определить правильность написания идентификаторов переменных:
a) DFGG
b) FGjk_1
c) CLГ2
d) RTY uj
e) 46HG
f) k.

4. Записать выражения на языке *Pascal* и расставить приоритеты (порядок выполнения арифметический действий):

a) $\dfrac{3a^3b-4b^2}{1{,}5ab^2-7ab}$;

b) $\dfrac{1-\cos^2(x+1)^3}{1-\sqrt{y}}-3\dfrac{5}{6}x-|tgy|$.

5. Записать рядом с каждой строкой, какое значение переменной будет находиться в соответствующей ей ячейки памяти. Составить для заданного фрагмента программу:
D := 4 →
A := D + 2*D →
D := A - D →
A : = A / D →

6. Определить значение логического выражения:
a) (4 < 6) and not(2 > 6) or not(4 > -7);
b) (c < 3) and ((b > 4) or a, при a = false, c = 5, b = 6.

Вариант II

1. Записать константы с фиксированной точкой и с плавающей точкой:

a) $1{,}75 \cdot 10^{-18}$

b) $-3{,}6 \cdot 10^{-12}$

c) 130000000.

2. Записать константы через математическую запись:

a) 3.173E5

b) -7.1134E7

c) 2.12E-4.

3. Определить правильность написания идентификаторов переменных:

a) 9Ly

b) kI_jkж1

c) ДR5

d) rIYj

e) hGG

f) k_pl

4. Записать выражения на языке *Pascal* и расставить приоритеты (порядок выполнения арифметический действий):

a) $\dfrac{3x^2 - 7{,}5y^2}{xy^2 - 13x}$

b) $\dfrac{1 - tg(x-y)}{ccos^2 x^3 - |\sin y|} - 1\dfrac{1}{7}x - \sqrt{y}.$

5. Записать рядом с каждой строкой, какое значение переменной будет находиться в соответствующей ей ячейки памяти.

Составить для заданного фрагмента программу:

F := -12 →

M := F / 4 →

F := F - A →

M : = F * M →

6. Определить значение логического выражения:

а) not((3 < 6) or (4 > 6)) and (3 < 2)

б) c and not(b < 6) or (a > 4), при c = true, a = 5, b = - 4.

Самостоятельная работа по теме «Линейные и разветвляющиеся программы»

Вариант I

1. Даны *a* и *b*. Вычислить *z*:

$$Z = \frac{8a^2 tga - 1{,}6ab^3}{3a - 2ab}.$$

2. Даны боковые стороны равнобедренного треугольника. Углы при основании равны 30°. Найти биссектрису, проведенную к основанию треугольника.

3. Подсчитать количество отрицательных чисел среди чисел *a, b, c*.

4. Для данного *x* вычислить значение функции:

$$F(x) = \begin{cases} \sin x^2, & \text{если } x \geq 0 \\ \cos x^2, & \text{если } -1 < x < 0 \end{cases}$$

Вариант II

1. Даны *a* и *b*. Вычислить *z*:

$$Z = \frac{b^2 - 0{,}1\sin^2 a}{a^2 + 0{,}5b}.$$

2. Определить время падения камня на поверхность земли с высоты *h*.

3. Подсчитать количество положительных чисел среди чисел *a, b, c*.

4. Для данного *x* вычислить значение функции:

$$F(x) = \begin{cases} 7 - 5x^2, & \text{если } 0 \leq x < 10 \\ \dfrac{\sqrt{x}}{x+3}, & \text{если } x \geq 10 \end{cases}$$

Самостоятельная работа по теме «Цикл с параметром»

Вариант I

1. Даны четырехзначные числа. Найти количество чисел, у которых сумма второй и третьей цифр равна *Z*.

2. Ввести с клавиатуры *n* целых чисел. Определить количество чисел, меньших числа ***a***, введенного с клавиатуры и кратных 3. Найти их сумму. Определить четность суммы.

3. Вычислить сумму элементов последовательности:

$$\frac{1+2}{1^3} + \frac{1+3}{2^3} + \frac{1+4}{3^3}.$$

Вариант II

1. Даны трехзначные числа. Найти сумму чисел, в которых вторая цифра равна *F*.

2. Ввести с клавиатуры *n* целых чисел. Определить количество чисел, оканчивающихся цифрой 3. Найти их сумму. Является ли она четной?

3. Вычислить сумму элементов последовательности:

$$\frac{1+1}{1+2} + \frac{2+1}{2+2} + \frac{3+1}{3+2}.$$

Самостоятельная работа на последовательность чисел и составление таблиц истинности логических выражений

Вариант I

1. Вычислить сумму *n* элементов последовательности:

$$1+\frac{1}{2^4}+\frac{1}{3^4}+\frac{1}{4^4}+\dots .$$

2. Вывести *n* элементов последовательности, заданной реккурентной формулой (a0 ввести с клавиатуры):

$$a_n = n + \frac{1}{\sin(a_{n-1})}.$$

3. Составить и вывести в табличном виде таблицу истинности логического выражения:

$$F(x,\ y,\ z) = \overline{x} \vee z \vee x \wedge y \wedge \overline{z}$$

Вариант II

1. Вычислить сумму *n* элементов последовательности:

$$(1+\frac{1}{1^2})+(1+\frac{1}{2^2})+(1+\frac{1}{3^2})+\dots .$$

2. Вывести *n* элементов последовательности заданной реккурентной формулой (a_0 и *x* ввести с клавиатуры)

$$a_n = a_{n-1} - nx.$$

3. Составить и вывести в табличном виде таблицу истинности логического выражения:

$$F(x,\ y,\ z) = z \vee x \wedge \overline{y} \vee \overline{x} \wedge y$$

Самостоятельная работа на цикл с предусловием

Вариант I

1. Дано длинное число. Сравнить старшую цифру числа с младшей цифрой числа.

2. Дано длинное число. Найти максимальную цифру числа и количества цифр, равных максимальной.

Вариант II

1. Дано длинное число. Является ли минимальная цифра числа нечетной.

2. Дано длинное число. Найти среднее арифметическое цифр числа и сумму цифр, больших среднего значения.

Самостоятельная работа на цикл с постусловием

1. Вычислить бесконечную сумму с заданной точностью ε (ε > 0). Считать, что требуемая точность достигнута, если вычислена сумма нескольких пер-

вых слагаемых и очередное слагаемое оказалось по модулю меньше, чем ε, — это и все последующие слагаемые можно не учитывать.

Вычислить:

$$\sum_{i=0}^{\infty} 1 \frac{1}{4^i + 5^{i+2}}.$$

2. Вычислить с заданной точностью константу p, используя бесконечный ряд Шарпа (1699 г.):

$$2\sqrt{3} \cdot (1 - \frac{1}{3^1 \cdot 3} + \frac{1}{3^2 \cdot 5} - \frac{1}{3^3 \cdot 7} + \frac{1}{3^4 \cdot 9} - ....).$$

Контрольная работа по теме «Основные алгоритмические структуры»

Вариант I

1. Дано натуральное число:

a) найти произведение цифр числа;

b) верно ли, что в данном числе нет цифры *A* (*A* вводить с клавиатуры).

2. Найти все трехзначные числа, которые при увеличении на 1 делятся на 2;

a) при увеличении на 2 делятся на 3;

b) при увеличении на 3 делятся на 4;

c) при увеличении на 4 делятся на 5.

3. Из данного натурального числа удалить все цифры *A* (*A* вводить с клавиатуры).

Вариант II

1. Дано натуральное число:

a) найти количество цифр числа;

b) верно ли, что в данном число заканчивается на нечетную цифру.

2. Найти количество трехзначных чисел, сумма цифр которых равна *A*, а само число заканчивается цифрой *B* (*A* и *B* вводить с клавиатуры).

3. Найти все трехзначные симметричные натуральные числа из промежутка от *A* до *B* (*A* и *B* вводить с клавиатуры).

Контрольная работа по теме «Вспомогательные алгоритмы»

Вариант I

1. Даны два числа. Определить, в каком из них больше четных цифр.

2. Найти сумму цифр числа.

3. Среди чисел из интервала от *A* до *B* найти все простые.

Вариант II

1. Даны два числа. Определить, в каком из них чаще встречается заданная цифра.

2. Найти сумму цифр числа.

3. Среди чисел из интервала от *A* до *B* найти все простые.

Контрольная работа по теме «Одномерный массив»

Вариант I

1. Правильно ли описан массив *A*? Если нет, то что надо изменить?

```
Type odmyarray=Array[1..n+20] Of Integer;
Var A : odmyarray;
```

2. Что получится в результате выполнения программы?

```
Program Variant_3;
 Const n=17;
  Type myarray=Array[1..n] Of Integer;
 Var B : myarray;
      I : Byte; p : Integer;
Begin
   P := 0;
   For i:=1 to n Do Begin
        B[i] := -35 + Random(121);
        If C[i] Mod 10 = 0 Then p:=p + 1;
    End;
  Writeln(p); Readln;
   End.
```

3. Дан массив целых чисел, состоящий из 15 элементов. Заполнить его с клавиатуры. Найти:

a) сумму положительных элементов, значение которых меньше 10;

b) вывести индексы тех элементов, значения которых кратны 3 и 5.

4. определить, равна ли сумма пары соседних элементов заданному числу.

Вариант II

1. Правильно ли описан массив *D*? Если нет, то что надо изменить?

```
Type odm=Array[-n..n] Of Integer;
Var D : odm;
```

2. Что получится в результате выполнения программы?

```
Program Variant_4;
 Const n=10;
  Type myarray=Array[1..n] Of Integer;

 Var A : myarray;
      I : Byte; p : Integer;
Begin
   P := 0;
   For i:=1 to n Do Begin
        A[i] := -50 + Random(71);
        If A[i] <= 10 Then p:=p + A[i];
    End;
  Writeln(p); Readln;
   End.
```

3. Дан массив целых чисел, состоящий из 10 элементов. Заполнить его с клавиатуры. Найти:

a) удвоенную сумму положительных элементов;

b) найти количество тех элементов, значения которых положительны и не превосходят числа A.

4. Найти номер последней пары соседних элементов с разными знаками.

Контрольная работа на сортировку, удаление и вставку элементов одномерного массива

Вариант I

Дан массив целых чисел ($n = 10$), заполненный случайным образом числами из промежутка: [-40; 30]. Необходимо:

a) удалить из него все элементы, которые состоят из одинаковых цифр (включая однозначные числа);

b) вставить число κ перед всеми элементами, в которых есть цифра 1 (κ вводить с клавиатуры);

c) переставить первые три и последние три элемента местами, сохраняя их следование.

Вариант II

Дан массив целых чисел ($n = 12$), заполненный случайным образом числами из промежутка [-10; 60]. Необходимо:

a) удалить из него все элементы, в которых последняя цифра четное, а само число делится на нее;

b) вставить число κ перед и после всех элементов, заканчивающихся на данную цифру (κ вводить с клавиатуры);

c) переставить элементы следующим образом: a[1], a[12], a[2], a[11], ..., a[5], a[8], a[6], a[7].

Контрольная работа на двумерные массивы

Вариант I

1. Найти максимальный и минимальный элементы массива по столбцам.

Вычислить среднее значение каждого столбца и среднее значение между max и min элементами массива.

Сравнить среднее значение min, max и среднее значение столбца.

2. Найти среднее значение по строкам и минимальный элемент всего массива. В нечетных строках нулевые элементы поменять на среднее значение соответствующей строки, а в четных — на минимальный элемент.

Вариант II

1. Найти максимальный и минимальный элементы массива по строчкам.

Вычислить среднее значение между max и min элементами.

Нулевые элементы поменять на среднее значениеmin,max соответствующей строки.

2. Найти среднее значение и минимальный элемент по столбцам. В каждом столбце элементы, стоящие после первого минимального элемента поменять на дробную часть среднего значения соответствующего столбца.

Контрольная работа на обработку строк

Вариант I

1. Вырежьте два первых символа и поместите их в конец данной символьной строки.

2. Найдите позицию первой и последней буквы *K*, входящих в символьную строку.

3. Напишите программу, располагающую латинские слова по алфавиту.

Вариант II

1. Написать программу, подсчитывающую количество слов в заданном предложении.

2. В данной строке символов все слоги *NAD* заменить на слоги *POD*.

3. Написать программу, выбирающую из заданной строки все слова, длина которых равна длине первого слова. Слова разделены одним пробелом.

Контрольная работа на работу с файлами

Вариант I

1. Дан файл *F*, компоненты которого являются действительными числами. Найти сумму компонент файла.

2. Дан файл *F*, компоненты которого являются действительными числами. Найти наибольшее значение компонент с нечетными номерами.

3. Даны символьные файлы *F* и *G*. Записать в файл *H* все начальные совпадающие компоненты файлов *F* и *G*.

Вариант II

1. Дан файл *F*, компоненты которого являются действительными числами. Найти произведение компонент файла.

2. Дан файл *F*, компоненты которого являются действительными числами. Найти наименьшее из значений компонент с четными номерами.

3. Даны символьный файл *F*. Записать в файл *H* с сохранением порядка те символы файла *F*, которым в этом файле предшествует буква «а».

Литература для учителей

Основная

1. *Абрамов, С. А.* Задачи по программированию для начинающих программистов, студентов вузов / С. А. Абрамов, Г. Г. Гнездилова [и др.]. — Москва : Наука : Главная редакция физико-математической литературы, 2010. — 224 с.

2. *Бабушкин, И. А.* Практикум по Турбо Паскалю : учебное пособие по курсам «Информатика и вычислительная техника», «Основы программирования» / И. А. Бабушкин [и др.]. — Москва : АБФ, 1998. — 384 с.

3. *Грацианова, Т. Ю.* Программирование в примерах и задачах / Т. Ю. Грацианова. — Москва : БИНОМ. Лаборатория знаний, 2013. — 368 с. — (ВМК МГУ — школе).

4. *Зеленяк, О. П.* Практикум программирования на Turbo Pascal. Задачи, алгоритмы и решения / О. П. Зеленяк. — Санкт-Петербург : ДиаСофтЮП, 2002. — 311 с.

5. *Тишин, В. И.* Программирование на Паскале : практикум / В. И. Тишин. — Москва : БИНОМ. Лаборатория знаний, 2013. — 396 с.

6. Турбо Паскаль 7.0. — Киев : Торгово-издательское бюро BHV, 1996. — 448 с. — (Для пользователя).

Дополнительная

7. *Вирт, Н.* Алгоритмы и структуры данных: с примерами на Паскале / Н. Вирт. — Санкт-Петербург : Невский Диалект, 2008. — 351 с.

8. *Грищенко, А. А.* Практикум по программированию на языке Паскаль : учебно-методическое пособие / А. А. Грищенко. — Перевоз : Перевозский строительный колледж, 2013. — 83 с.

9. *Гуриков, С. Р.* Программирование в среде Lazarus для школьников и студентов : учебное пособие / С. Р. Гуриков. — Москва : Инфра-М, 2018. — 336 с. — (Профессиональное образование).

10. *Милов, А. В.* Основы программирования в задачах и примерах / А. В. Милов. — Москва : Фолио, 2002. — 401 с.

11. *Павловская, Т. А.* Паскаль. Программирование на языке высокого уровня : практикум / Т. А. Павловская. — Санкт-Петербург : Литер, 2009. — 432 с.

12. *Рапаков, Г. Г.* Программирование на языке Pascal : учебное пособие / Г. Г. Рапаков, С. Ю. Ржеуцкая. — Санкт-Петербург : БХВ-Петербург, 2005. — 474 с.

13. *Ставровский, А. Б.* Турбо Паскаль 7.0 : учебник для вузов / А. Б. Ставровский. — Киев : BHV, 2000. — 400 с. — (Библиотека студента).

14. *Фаронов, В. В.* Система программирования Delphi. Наиболее полное руководство / В. В. Фаронов. — Санкт-Петербург : БХВ-Петербург, 2006. — 912 с.

15. *Фаронов, В. В.* Турбо Паскаль 7.0 : практика программирования : учебное пособие / В. В. Фаронов. — Санкт-Петербург : Питер, 2010. — 414 с.

16. *Фигурнов, В. Э.* IBM PC для пользователя / В. Э. Фигурнов. — Москва : Финансы и статистика. Информатика и компьютеры, 2000. — 638 с.

17. *Франкен, Г.* MS-DOS 6.0 ... для пользователя / Г. Франкен, С. Молявко. — Киев : BHV, 1993. — 474 с.

18. *Чернов, А. А.* Информатика. 9—11 классы. Контрольные и самостоятельные работы по программированию / А. А. Чернов, А. Ф. Чернов. — Волгоград : Учитель, 2009. — 302 с. — (Контрольные и самостоятельные работы).

19. *Юркин, А. Г.* Задачник по программированию / А. Г. Юркин. — Санкт-Петербург : Питер, 2012. — 193 с.

Литература для учащихся

Основная

20. *Грацианова, Т. Ю.* Программирование в примерах и задачах / Т. Ю. Грацианова. — Москва : БИНОМ. Лаборатория знаний, 2013. — 368 с. — (ВШК МГУ — школа).

21. *Грищенко, А. А.* Практикум по программированию на языке Паскаль : учебно-методическое пособие / А. А. Грищенко. — Перевоз : Перевозский строительный колледж, 2013. — 83 с.

22. *Тимофеевская, М.* Изучаем программирование / М. Тимофеевская — Санкт-Петербург : Питер, 2010. — 378 с.

23. *Тишин, В. И.* Программирование на Паскале : практикум / В. И. Тишин. — Москва : БИНОМ. Лаборатория знаний, 2013. — 396 с.

Интернет-источники

24. 40 уроков по Pascal. — Текст : электронный. — URL: http://www.gmcit.murmansk.ru/text/information_science/profile/methodic/pascal/pascal.html (дата обращения: 13.07.2020).

25. Паскаль. Основы программирования. — Текст : электронный. — URL: https: // pas 1/ru/programmig (дата обращения: 13.07.2020).

26. Полный обучающий курс Турбо Паскаль. — Текст : электронный. — URL: http:// books. kulichki.ru/data/pascal/pas1 (дата обращения: 13.07.2020).

27. *Ушаков, Д. М.* Паскаль для школьников / Д. М. Ушаков, Т. А. Юркова. — Текст : электронный. — URL: https: // may.alleng.org/d/comp/comp84.htm (дата обращения: 13.07.2020).

СОДЕРЖАНИЕ

Учебное издание

УЧЕБНАЯ ПРОГРАММА ЭЛЕКТИВНОГО КУРСА «АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ» ДЛЯ 10—11 КЛАССОВ

Редактор *Н. А. Елизарова*
Компьютерная верстка *Л. И. Половинкиной*

Оригинал-макет подписан в печать 22.06.2020 г.
Формат 60 × 84 $^{1}/_{8}$. Бумага офсетная. Гарнитура «TimesET».
Печать офсетная. Усл.-печ. л. 10,7. Тираж 100 экз. Заказ 2644.

Нижегородский институт развития образования,
603122, Н. Новгород, ул. Ванеева, 203.
www.niro.nnov.ru

Отпечатано в издательском центре учебной
и учебно-методической литературы ГБОУ ДПО НИРО

Учебная программа элективного курса «Алгоритмизация и программирование»

для 10–11 классов

НИРО
Нижегородский институт развития образования