Министерство образования Республики Беларусь
БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра « Робототехнические системы»

А.А. Москаленко
З.И. Кононенко

РЕШЕНИЕ ПРИКЛАДНЫХ ЗАДАЧ В ИНТЕГРИРОВАННОЙ СРЕДЕ ТУРБО ПАСКАЛЬ

Методическое пособие
по дисциплинам «Информатика»,
«Математические модели информационных процессов и управления», «Основы алгоритмизации и программирование»
для студентов специальностей: 1-53 01 01 «Автоматизация технологических процессов и производств»,
1-53 01 02 «Автоматизированные системы обработки информации» и 1-53 01 06 «Промышленные роботы и робототехнические комплексы»

Минск
БНТУ
2011

УДК 004.438 (075.8)
ББК 32.97-018.1
М 82

Рецензенты:
Р.В. Новичихин, О.В. Бугай

Москаленко, А.А.

М 82 Решение прикладных задач в интегрированной среде Турбо Паскаль: методическое пособие по дисциплинам «Информатика», «Математические модели информационных процессов и управления», «Основы алгоритмизации и программирование» для студентов специальностей: 1-53 01 01 «Автоматизация технологических процессов и производств», 1-53 01 02 «Автоматизированные системы обработки информации» и 1-53 01 06 «Промышленные роботы и робототехнические комплексы» / А.А. Москаленко, З.И. Кононенко. – Минск: БНТУ, 2011. – 62 с.

ISBN 978-985-525-602-2.

В методическом пособии приведен комплекс для обучения студентов программированию в интегрированной среде Турбо Паскаль.

Излагается краткая теория, особое внимание обращается на ряд специфических моментов, вызывающих трудности у студентов при программировании.

Пособие также может быть использовано инженерами, самостоятельно изучающими современные языки программирования.

УДК 004.438 (075.8)
ББК 32.97-018.1

ISBN 978-985-525-602-2

ВВЕДЕНИЕ

Алгоритмический язык Паскаль получил широкое распространение во всем мире благодаря простоте и доступности большому количеству людей, работающих в различных отраслях народного хозяйства. Именно такую цель и ставил швейцарский профессор Никлас Вирт, создавший в конце 60-х годов этот язык как специальный для обучения начинающих хорошему стилю программирования.

Успеху языка способствовало и то, что по своей идеологии Паскаль наиболее близок к современной теории и технологии программирования, так как довольно полно отражает идеи структурного программирования. Кроме того, он приспособлен для применения общепризнанной в настоящее время технологии разработки программ методом пошаговой детализации (нисходящего программирования), позволяет создавать не только несложные программы, но и структурированные программы трудоемких и сложных вычислений, предоставляя возможности работы как с числовой, так и с символьной информацией.

Язык Паскаль постоянно совершенствовался. Для персональных компьютеров появились революционные, по своей сущности, компиляторы с языка Паскаль для IBM PC (главным образом фирм Microsoft и Borland), представляющие диалоговые системы, называемые интегрированными средами (Turbo Pascal). К настоящему времени сформировался ряд версий интегрированных сред Turbo Pascal: 5.0, 5.5, 6.0, 7.0
и 8.0, причем соблюдается принцип их совместимости снизу вверх. В версию 5.0 был включен интегрированный Turbo – отладчик; версия же 5.5 позволила создавать объектно-ориентированные программы. В свою очередь, версия 6.0 располагала тремя новыми возможностями: интегрированной средой для разработчика; дополнительными режимами транслятора и встроенным Ассемблером; объектно-ориентированной оболочкой для создания прикладных программ – Turbo Vision. Кроме того, версия 6.0 позволяет редактировать несколько файлов и дает новые возможности управления в интегрированной среде при помощи манипулятора «мышь».

Задача данного издания состоит в том, чтобы познакомить студентов с современной интегрированной средой Турбо Паскаль, в частности с версией 7.0. Что же касается версии 8.0, то она требует

большого объема памяти и представляет мощную систему для разработки технических программ.

1. ИНТЕГРИРОВАННАЯ СРЕДА ПРОГРАММИРОВАНИЯ. РАБОТА С ГЛАВНЫМ МЕНЮ. КРАТКИЕ СВЕДЕНИЯ

После запуска компилятора на экране отображается основное окно интегрированной среды программирования (рис. 1.1).

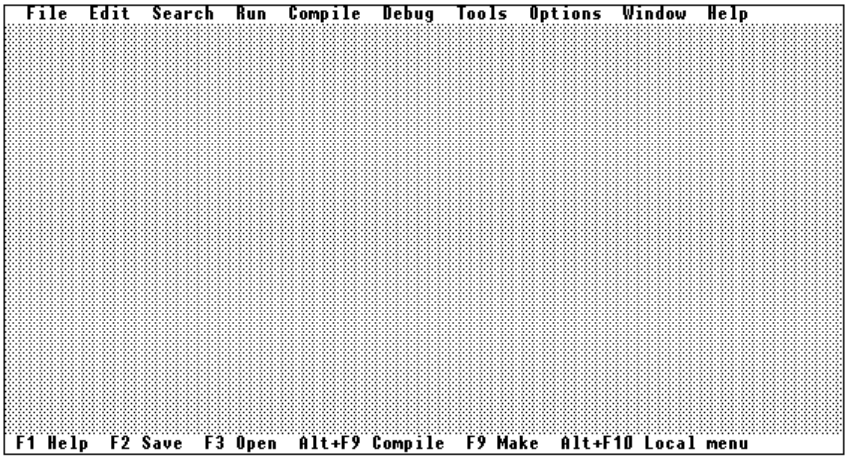

Рис. 1.1. Основной экран системы Турбо Паскаль

Интегрированная среда (Turbo-среда) позволяет одновременно набирать тексты программ с использованием встроенного редактора текстов, компилировать их, выполнять, производить отладку программ и т. д.

Основной экран интегрированной среды Турбо Паскаль состоит из трех различных по функциональному назначению частей: строки главного меню, рабочей зоны и строки состояния.

Строка главного меню

Обратиться к любой команде главного меню можно одним из трех способов:

• нажать клавишу F10 и с помощью клавиш перемещения курсора выбрать необходимую команду;

• можно использовать мышь (курсор мыши необходимо установить на любую команду меню и нажать левую клавишу мыши);

• используя «горячие клавиши» (в ключевом слове каждой команды выделяется одна литера, как правило, заглавная. Одновременно используя нажатие клавиш Alt с любой из таких литер, можно перейти к выполнению существующей команды.).

Строка меню содержит имена следующих команд:

• **File** – позволяет выполнять все основные операции с файлами (создавать новые, загружать имеющиеся, сохранять созданные и отредактированные файлы, выводить на печатающее устройство содержимое этих файлов);

• **Edit** – позволяет выполнять все основные операции редактирования текста (копировать, вставлять, удалять фрагменты текста, а также восстанавливать первоначальный вариант редактируемого текста);

• **Search** – позволяет осуществлять поиск фрагментов текста и при необходимости производить замену найденного фрагмента новым;

• **Run** – позволяет запускать программу, находящуюся в рабочей зоне, а также, при необходимости, пошагово выполнить данную программу или ее часть;

• **Compile** – позволяет осуществить компиляцию программы, которая находится в активном окне;

• **Debug** – содержит команды, облегчающие процесс поиска ошибок в программе (**Breakpoints** – точки останова, окно отладки – **Watch**, окно используемых подпрограмм, окно регистров, окно выходных результатов и некоторые другие);

• **Tools** – позволяет выполнить некоторые программы не выходя из интегрированной среды;

• **Window** – позволяет выполнять все основные операции с окнами (открывать, закрывать, перемещать, изменять размер);

• **Help** – позволяет получить имеющуюся в системе справочную информацию.

Выйти из меню можно, нажав клавишу [**Esc**].

Строка состояния, находящаяся в нижней части экрана, демонстрирует некоторые из доступных (часто используемых) операций интегрированной среды и комбинаций клавиш для их быстрого вызова.

Задание 1.1

1. Войти в режим редактирования и набрать текст следующей программы:

```
Program Example;
var
      A, B, C, Y : real;
Begin { Исходные данные }
      A:=1.5;
      B:=4.3;
      C:=7.1;
Y:=A*SQR(B)+C*B;
      Writeln (‘ Исходные данные ’);
      Writeln (‘A=’,A:4:1,’ B=’,B:4:1,’ C=’,C:4:1);
      Writeln (‘ Результат расчёта Y=’,Y:0:2);
Readln;
End.
```

2. Запустить программу на компиляцию и выполнить через команды главного меню.

3. Просмотреть результат выполнения программы (нажав <**Alt**>+<**F5**>), для возврата – любая клавиша.

При компиляции программы могут возникать ошибки. При этом курсор устанавливается на ту позицию, где находится ошибка. Если сразу после этого нажать **F1**, то на экране по-

явится дополнительная информация об ошибке. После исправления ошибки необходимо повторить компиляцию.

Задание 1.2

1. Изучить команду главного меню **RUN**.
2. Пошагово выполнить данную программу.
3. Посмотреть в **Watch**-окне (Ctrl+F7) значения исходных данных и результат расчета.

Задание 1.3

1. Изучить команду главного меню **Edit**.
2. Открыть новое окно и переписать часть набранной программы с 1-й строки до зарезервированного слова **Begin** в новое окно.

2. СТРУКТУРА ПРОГРАММЫ. АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ И ВЫРАЖЕНИЯ. СТАНДАРТНЫЕ ФУНКЦИИ. КОММЕНТАРИИ. ОПЕРАТОРЫ ПРИСВАИВАНИЯ. ЛИНЕЙНАЯ ПРОГРАММА

Структура программы

После заголовка следует программный блок, состоящий в общем случае из семи разделов: списка имен подключаемых библиотечных модулей (он определяется зарезервированным словом USES), описания меток, описания констант, определения типа данных, описания переменных, описания процедур и функций, операторов.

Структура программы в общем случае выглядит следующим образом:

{ <директивы компилятора>}

USES < имя 1, имя 2 ...>;
LABEL ...;
CONST ...;
TYPE ...;
VAR...;
PROCEDURE <имя>;
<тело процедуры>
FUNCTION <имя>; <тело функции>
BEGIN <операторы> END.

Любой раздел, кроме раздела операторов основной программы, может отсутствовать.

Разделы описания констант и переменных

В программе каждый элемент данных является либо константой, либо переменной. Константы и переменные определяются идентификаторами (именами), по которым к ним можно обращаться для получения текущих значений.

Константами называются элементы данных, значение которых известно заранее и в процессе выполнения программы не изменяется. В Паскале для определения констант служит зарезервированное слово **const**:

Const <идентификатор>=<значение константы (целые, вещественные и шестнадцатеричные числа; логические константы, строковые, символьные)>.

Кроме обычных широко используются так называемые ***типизированные константы***, которые могут менять свое значение как переменные. При их описании дополнительно указывается тип, т. е. фактически они представляют собой переменные с начальными значениями:

```
Const
   Max : word = 100;
   X : byte = 16;
```

Типизированные константы могут быть любого типа, кроме файлов.

Переменные в отличие от констант могут менять свои значения в процессе выполнения программ. Каждая переменная и константа принадлежат к определенному типу данных. Тип констант автоматически распознается компилятором без предварительного описания. Тип переменных должен быть описан перед тем, как с переменными будут выполняться какие-либо действия. Для описания переменных предназначено зарезервированное слово **Var**:

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

Стандартные функции

Рассмотрим основные стандартные функции языка Паскаль (табл. 2.1). В тригонометрических функциях синуса или косинуса аргумент должен быть задан в радианах. Если аргумент задан в градусах, то для перевода его в радианы используется формула

$$Y = X \cdot P_i / 180.$$

Таблица 2.1

Функции языка Паскаль

Функция	Назначение
ABS(x)	Вычисление абсолютного значения *x*
SQR(x)	Вычисление квадрата *x* (*x* · *x*)
SIN (x)	Вычисление синуса *x*
COS (x)	Вычисление косинуса *x*
ARCTAN(x)	Вычисление арктангенса *x*
EXP(x)	Вычисление экспоненты *x*
LN(x)	Вычисление натурального логарифма *x*
SQRT(x)	Вычисление квадратного корня из *x*

TRUNC(x)	Вычисление целой части *x*
ROUND(x)	Округление *x* в сторону ближайшего целого
ODD(x)	TRUE, если *x* – нечетное FALSE, если *x* – четное

Для вычисления остальных тригонометрических функций необходимо использовать следующие соотношения:

tg *x* = sin *x* / cos *x* **csc *x* = 1/ sin *x***

ctg *x* = cos *x* / sin *x* **sc *x* = 1/ cos *x*** и т. д.

Для вычисления логарифма с основанием *a*:

$$\log_a x = \ln(x) / \ln(a).$$

Для операции возведения в степень применяют стандартные функции:

x^a соответствует **EXP (a * Ln (x))**.

Приоритет операций

Операции с более высоким приоритетом выполняются раньше, чем операции с более низким.

Если все операнды в выражениях имеют одинаковый приоритет, то они выполняются слева направо.

Выражения, заключенные в круглые скобки, выполняются в первую очередь.

В Турбо Паскале определены следующие операции:

- **унарные: изменение знака (-), not, @;**
- **мультипликативные: / , * , div, mod, and, shl, shr;**
- **аддитивные: +, - , or, xor;**
- **отношения: =, <>, <, >, <=, >=, in.**

Приоритет операций убывает в указанном порядке, т. е. наивысшим приоритетом обладают унарные операции, низшим – операции отношения.

З а д а н и е

1. Записать операторы присваивания, реализующие зависимости, приведенные в табл. 2.2.

2. Составить простейшую программу с использованием комментариев.

3. Получить листинг.

4. Выполнить программу, задавшись значениями переменных в диапазоне [0,25–6,5] с помощью операторов присваивания.

Таблица 2.2

Исходные данные

Вариант	Арифметические выражения	
1	$\frac{\sqrt[3]{\lvert x+\sin x^2\rvert}-2ab}{\log_2\lvert x\rvert}$	$\frac{\pi\cdot\lg\lvert x\rvert-a}{\cos^4 x}$
2	$\frac{\sqrt{c^2+a^2}+2c\cdot d\cdot \operatorname{tg}x}{\ln\lvert x\rvert+e^{\sqrt{\lvert x-1\rvert}}}$	$\frac{1{,}8\cdot\sqrt[3]{\lvert\sin x\rvert}+e^{\sqrt{\lvert x-1\rvert}}}{\sqrt{\lvert c\rvert}+\sqrt[3]{\lvert a\rvert}}$
3	$\frac{a^2-4a\cdot b+b^2}{3x\cdot\lg b}$	$\frac{(a+b)\cdot e^x}{\lvert x+a\rvert}+\frac{x^2+w}{w\cdot\lvert\cos x\rvert}$
4	$\frac{\sqrt[3]{\lvert a\cdot x+r\rvert}}{\sqrt{\lvert\operatorname{tg}x\rvert}}-e^a$	$\frac{\sin x^2+\operatorname{tg}a}{a+\lvert\sin a\rvert}$
5	$\frac{2\pi\cdot j+\operatorname{tg}x^2}{2j\cdot\lvert\sin x\rvert}$	$\frac{a^6+\cos^4 x}{\sqrt[3]{\lvert j\rvert}}$
6	$\frac{\left(a\cdot x+2b\cdot x+x^2\right)\cdot\lvert\cos x^2\rvert}{\pi\cdot e^x}$	$\frac{e^{\sqrt{\lvert w+1\rvert}}\cdot\operatorname{tg}x\cdot\sin w}{1{,}85w+\sqrt[3]{y^2}}$
7	$\frac{\sqrt{\lvert t\rvert}+a^3}{t\cdot\lvert\sin a/2\rvert}$	$\frac{\left(\sin a+\operatorname{tg}^2 x\right)\cdot e^{\lvert a\rvert}}{\log_2 a}$
8	$\frac{\sqrt{\lvert(a+b)\rvert}/(c+d)}{3\pi\sqrt[3]{\lvert a-b\rvert}}$	$\frac{\left(\lvert\ln x^2\rvert+a\cdot c\right)\cdot e^{\lvert a\cdot b\rvert}}{b^2+c^2}$

9	$\dfrac{\sqrt[3]{\lvert x+\sin x^2\rvert}-2a\cdot b}{\lvert\log_2 x\rvert}$	$\dfrac{\pi\cdot\lvert\lg x\rvert-a}{\cos^4 x}$

Окончание табл. 2.2

Вариант	Арифметические выражения	
10	$\dfrac{\sqrt{c^2+a^2}+2c\cdot d\cdot\operatorname{tg}x}{\lvert\ln x+e^c\rvert}$	$\dfrac{1{,}8\cdot\sqrt[3]{\lvert\sin x\rvert}+e^{\sqrt{\lvert x-1\rvert}}}{\sqrt{\lvert c\rvert}+\sqrt{\lvert a\rvert}}$

3. ПРОГРАММИРОВАНИЕ РАЗВЕТВЛЯЮЩИХСЯ ВЫЧИСЛИТЕЛЬНЫХ ПРОЦЕССОВ С ИСПОЛЬЗОВАНИЕМ ОПЕРАТОРОВ УСЛОВНОГО И БЕЗУСЛОВНОГО ПЕРЕХОДОВ И ЛОГИЧЕСКИХ ВЫРАЖЕНИЙ ОБЩЕГО ВИДА

Условные операторы

Условные операторы обеспечивают выполнение или невыполнение некоторого оператора, групп операторов или блока в зависимости от заданных условий. Паскаль допускает использование двух условных операторов: **if** и **case**.

Оператор условия **if** является одним из самых популярных средств, изменяющих естественный порядок выполнения операторов программы. Он может принимать одну из следующих форм:

1. **if <условие> then <оператор 1 >**
else <оператор 2>;
2. **if < условие> then <оператор >;**

Оператор выбора **case** является обобщением оператора **if** и позволяет сделать выбор из произвольного числа изменяющихся вариантов. Он состоит из выражения, называемого селектором, и альтернативных операторов, каждому из которых предшествует список констант выбора. Как и в операторе **if**, здесь может присутствовать слово **else**, имеющее тот же смысл.

Общий вид

case <выражение селектор> of
<список констант выбора 1> : <оператор 1; >
<список констант выбора 2> : <оператор 2; >
. . . .
<список констант выбора N> : <оператор N >
else <оператор>
end;

Логические операции

Операция	Название	Пример
Not (¬)	Логическое отрицание	Not A
And (∧)	Логическое И	A And B
Or (∨)	Логическое ИЛИ	A Or B
Xor (+)	Исключающее ИЛИ	A Xor B

And производит логическое умножение в соответствии со следующей таблицей истинности:

1 and 1 = 1 **0 and 1 = 0**
1 and 0 = 0 **0 and 0 = 0**

Or выполняет сложение операндов в двоичной форме в соответствии с таблицей истинности:

1 or 1 = 1 **0 or 1 = 1**
1 or 0 = 1 **0 or 0 = 0**

Xor производит сложение операндов в соответствии с таблицей истинности:

1 xor 1 = 0 **0 xor 1 = 1**
1 xor 0 = 1 **0 xor 0 = 0**

З а д а н и е 3.1

1. Составить схему алгоритма и программу, используя, оператор условного перехода **IF**.

2. Вывести листинг программы, распечатки исходных данных и результатов расчета на экран видеотерминала и на принтер. Функции для вычисления приведены в табл. 3.1.

Таблица 3.1

Исходные данные

Вариант	Функция	Исходные данные
1	2	3
1	$Y=\begin{cases}\sqrt{\lvert a-x\rvert}\cdot\sin^2 x, & \text{если } a<x;\\ \left(\dfrac{x}{\lvert a+x\rvert}\right)\cdot\sqrt[3]{\lvert\sin x\rvert}, & \text{если } a=x;\\ e^{\sqrt{\lvert x\rvert}}, & \text{если } a>x\end{cases}$	$a=2{,}37\cdot10^{-2}$ $x=0{,}927$
2	$Y=\begin{cases}ax+0{,}23x^2\log_2 a, & \text{если } a<x;\\ \left(\dfrac{xe^a}{\lvert a+x\rvert}\right)\cdot\sqrt{\lvert\cos x\rvert}, & \text{если } a=x;\\ x\cdot\operatorname{tg}a, & \text{если } a>x\end{cases}$	$a=0{,}462\cdot10^{-1}$ $x=0{,}86$

3	$Y=\begin{cases}\left(a^2+x^2\right)\cdot e^x, & \text{если } a<x;\\ \sqrt{\lvert a\rvert}\cdot\sin^4 x, & \text{если } a=x;\\ \ln\left\lvert(a-x)\cdot\cos x^2/a\right\rvert, & \text{если } a>x\end{cases}$	$a=0{,}357\cdot10^{-2}$ $x=0{,}983$

Продолжение табл. 3.1

1	2	3
4	$Y=\begin{cases}\ln\lvert a+x\rvert\cdot\cos\lvert x^3\rvert, & \text{если } a<x;\\ e^{1,2}-\sqrt{\lvert a+x\rvert}, & \text{если } a=x;\\ \dfrac{\sqrt[3]{\lvert a+x\rvert}}{(a-x)}, & \text{если } a>x\end{cases}$	$a=0{,}148\cdot10^2$ $x=0{,}573$
5	$Y=\begin{cases}a^2+\sqrt{\lvert a^2+x\cdot\sin x\rvert}, & \text{если } a<x;\\ 2x^2+a^3\cdot\mathrm{tg}x, & \text{если } a=x;\\ \dfrac{x^2}{\sqrt{\lvert a\rvert}}, & \text{если } a>x\end{cases}$	$a=0{,}637$ $x=0{,}234\cdot10^2$
6	$Y=\begin{cases}e^{0.2}+\sqrt{a+x}, & \text{если } a<x;\\ (a+x)\cdot\sqrt[3]{\lvert\sin x\rvert}, & \text{если } a=x;\\ \sqrt{\lvert a+x\rvert}, & \text{если } a>x\end{cases}$	$a=0{,}725\cdot10^{-1}$ $x=0.56$

7	$Y=\begin{cases} 2{,}75e^{\lvert x+a\rvert}+\cos^4 x, & \text{если } a<x; \\ \dfrac{(x+a)\cdot \operatorname{tg} x}{\lg\lvert x\rvert}, & \text{если } a=x; \\ e^{ax}+\dfrac{a\cdot\sin^2 x}{\sqrt[3]{\cos^2 x}}, & \text{если } a>x \end{cases}$	$a = 0{,}567\cdot10^{-1}$ $x = 0{,}37$

Окончание табл. 3.1

1	2	3
8	$Y=\begin{cases} e^{0,15}+\dfrac{(a+x)^2}{\ln x}\cdot e^x, & \text{если } a<x; \\ \left(x^2-a^2x\right)\cdot \operatorname{tg} x, & \text{если } a=x; \\ (a+x)\cdot e^{\sqrt{\lvert a+x\rvert}}, & \text{если } a>x \end{cases}$	$a = 0{,}832\cdot10^{-1}$ $x = 0{,}64$
9	$Y=\begin{cases} \lvert\sin^3 bx\rvert+\lvert\cos^3 a\rvert, & \text{если } a<x; \\ \operatorname{tg}(ax)+\ln\lvert b\rvert, & \text{если } a=x; \\ \ln\lvert ax+b\rvert-\ln\lvert(a-b)^3\rvert, & \text{если } a>x \end{cases}$	$a = 0{,}354\cdot10^{-1}$ $x = 0{,}56$ $b = 2{,}44$
10	$Y=\begin{cases} ae^x+\cos(3bx-0{,}2), & \text{если } a<x; \\ b\cdot\cos\left(a^3-\lvert x\rvert^3\right), & \text{если } a=x; \\ \operatorname{tg}4{,}5x+\dfrac{x-5}{\sin(a+b)^2}, & \text{если } a>x \end{cases}$	$a = 0{,}475\cdot10^{-1}$ $x = 3{,}76$ $b = 1{,}23$

З а д а н и е 3.2

Определить, по какому из четырех выражений вычислена функция. Исходными данными задаться в диапазоне +/– (3,75–12,35).

Таблица 3.2

Исходные данные

Вари-ант	Функции	Условие
1	2	3
1	$B=\begin{cases} j^3+\sin ax \\ -3\mathrm{tg}(a-2)+e^{-x} \\ \cos^4 ax/\,j \\ x^2-a \end{cases}$	Если $j < a \wedge x > j$ Если $j < a \vee x > a$ Если $\neg (x < a) \wedge \neg (x > j)$ В остальных случаях
2	$W=\begin{cases} \sqrt[3]{\lvert x\rvert}+\sqrt[4]{\lvert a\rvert} \\ ae^{i}\cdot \sin^2 x \\ \ln\lvert x-7\rvert+\dfrac{a}{i} \\ a\cdot\cos x \end{cases}$	Еслиu $\neg (x + a) < x^2$ Если $x < i \vee a > i$ Если $x < a \wedge x > i$ В остальных случаях

3	$Y = \begin{cases} kx + ab \\ (kx + ab) \cdot \cos^2 x \\ (kx + ab) \cdot e^i \\ \lg\lvert x\rvert + i \end{cases}$	Если $k < x \vee \neg (a < b)$ Если $k > x \wedge a > b$ Если $i > x \wedge \neg (a < b)$ В остальных случаях
4	$P = \begin{cases} \sqrt{\lvert \mathrm{tg}a + \mathrm{tg}b\rvert} \\ e^i \cdot \cos(a + b)^2 \\ \ln\lvert a\rvert \cdot \sqrt{\lvert a + b\rvert} \\ \log_2\lvert i\rvert \end{cases}$	Если $a < b \vee b > i$ Если $a > b \vee i < a$ Если $a < b \wedge i < a$ В остальных случаях

Продолжение табл. 3.2

1	2	3
5	$L = \begin{cases} 3\lvert x\rvert - e^a (a/\lvert x - b\rvert) \\ \sqrt{\lvert x\rvert}/a + \mathrm{arctg}x \\ \lg\lvert a\rvert + \sin x^2 \\ (a + b)^2 / \cos^2 x \end{cases}$	Если $x < a \wedge \neg (x > b)$ Если $x > a \vee x > b$ Если $x \neq a \wedge i \neq b$ В остальных случаях
6	$Y = \begin{cases} 0{,}5\cos ax + e^{\mathrm{i}}/i^2 \\ ((x^2 + a^2)/\lvert x - i\rvert) + \ln\lvert b\rvert \\ \sqrt{\lvert \sin^2 ix + \cos^4 b\rvert} + (a/b) \\ a^2 + 2ab + \log_2 x \end{cases}$	Если $x > a \vee \neg (x > b)$ Если $x > a \wedge x > b$ Если $x \neq a \wedge i \neq b$ В остальных случаях

7	$I = \begin{cases} 6{,}75x^2 + e^b \lg\lvert 7a\rvert \\ 2\mathrm{tg}a + \lvert b^3\rvert + x \\ \sqrt[3]{\lvert x - a + b\rvert} + (x/\lvert a - b\rvert) \\ \ln\lvert x\rvert + \sin^3 x \end{cases}$	Если $a < b \wedge x \neq a$ Если $a > b \vee \neg (x < a)$ Если $a \neq b \wedge a \neq x$ В остальных случаях
8	$C = \begin{cases} \sin(\ln\lvert ax\rvert) \\ (e^b/\lvert x - a\rvert) + \mathrm{arctg}b \\ \mathrm{tg}\lvert\sqrt{\lvert x - ab\rvert}\rvert \\ \sqrt{\lvert a + b + x\rvert} \end{cases}$	Если $a \neq 0 \wedge x \neq 0 \wedge a > b$ Если $a < b \vee \neg (a < x)$ Если $x < b \wedge x > a$ В остальных случаях
9	$J = \begin{cases} \sqrt{\lvert x\rvert} + \sqrt[3]{\lvert x\rvert} + e^a \ln b \\ \sin^2 a + \sin\lvert b\rvert + \mathrm{tg}\lvert x\rvert \\ (\lvert x + a\rvert/\lvert x - a\rvert) + \cos^4 x \\ (\sin^2 x)/\lvert ab - x\rvert \end{cases}$	Если $a > b \wedge b \neq 0$ Если $x = b \vee x = a$ Если $a < x \wedge \neg (b > a)$ В остальных случаях

Окончание табл. 3.2

1	2	3
10	$Y = \begin{cases} (x + b^2 + a^2) \cdot \ln\lvert x\rvert \\ (x - b) \cdot (\lvert x - a\rvert) \cdot \cos^3 x \\ ax/\sqrt{\lvert bx\rvert} + e^a \\ \lvert x - b\rvert/(ax^2 + \mathrm{tg}b) \end{cases}$	Если $a \neq 0 \wedge x = 0$ Если $x > a \vee x > b$ Если $x \neq 0 \vee \neg (a > b)$ В остальных случаях

4. ЦИКЛИЧЕСКИЕ ВЫЧИСЛИТЕЛЬНЫЕ ПРОЦЕССЫ. ЦИКЛЫ С ПАРАМЕТРОМ (С ПРЕДУСЛОВИЕМ И ПОСТУСЛОВИЕМ)

Оператор повтора Repeat

Оператор повтора Repeat состоит из заголовка (Repeat), тела и условия окончания (Until).

Формат:
Repeat
<оператор 1>;
…….
<оператор N>;
Until <условие>;

Условие – выражение булевского типа. При написании условия допустимы булевские операции и операции отношения. Операторы, заключенные между словами **Repeat** и **Until**, являются телом цикла. Вначале выполняется тело цикла, затем проверяется условие выхода из цикла. Если результат булевского выражения **False**, тело активизируется еще раз, результат **True** – происходит выход из цикла.

Оператор **Repeat** имеет три характерные особенности:

выполняется по крайней мере один раз;

тело цикла выполняется, пока условие равно **False**;

в теле цикла может находиться произвольное число операторов без операторных скобок **begin ... end**.

Оператор повтора While

Оператор While аналогичен оператору repeat, но проверка выполнения условия выполнения тела цикла производится в самом начале оператора.

Формат:

While <условие> do <тело цикла>;

Условие – булевское выражение, а ***тело цикла*** – простой или составной оператор. Перед каждым выполнением тела цикла вычисляется значение выражения условия. Если результат равен **True**, тело цикла выполняется и снова вычисляется выражение условия. Если результат равен **False**, происходит выход из цикла и переход к первому после **while** оператору. Если перед первым выполнением цикла значение выражения было **False**, тело цикла вообще не выполняется и происходит переход к следующему оператору.

З а д а н и е 4.1

1. Составить схему алгоритма для вычисления функций, приведенных в табл. 4.1.
2. Написать программу на языке Паскаль.
3. Произвести расчеты на микроЭВМ.
4. Распечатать листинг программы.
5. Исходные данные и промежуточные результаты вывести на экран видеотерминала (дисплея) и на печатающее устройство.

Таблица 4.1

Исходные данные

Вариант	Функции	Исходные данные
1	2	3

1	$Y=\begin{cases}\dfrac{(x^2-a^2x)\cdot \mathrm{tg}x}{\sqrt{\lvert a+x\rvert}}, & \text{если } a\le x;\\ (a+x)\cdot e^{\sqrt{\lvert x+a\rvert}}, & \text{если } a>x\end{cases}$	$x=0{,}35;$ $a=[0{,}5–2{,}5];$ шаг : $\Delta a=0{,}5$
2	$Y=\begin{cases}\dfrac{(x+a)\cdot \mathrm{tg}x}{\ln\lvert x\rvert}, & \text{если } a\le x;\\ e^{ax}+a\sin x^2, & \text{если } a>x\end{cases}$	$x=0{,}54;$ $a=[0{,}2–1];$ шаг : $\Delta a=0{,}2$
3	$Y=\begin{cases}\sqrt{a^2+x^2}, & \text{если } a>x;\\ a-x\sqrt[3]{\sin^4\lvert x\rvert}, & \text{если } a\le x\end{cases}$	$x=0.7;$ $a=[0{,}3–1{,}5];$ шаг : $\Delta a=0{,}3$
4	$Y=\begin{cases}\dfrac{x\cdot\sqrt{\lvert\sin x\rvert}}{\lvert a+x\rvert}, & \text{если } a\le x;\\ e^{\sqrt{\lvert x\rvert}}, & \text{если } a>x\end{cases}$	$x=0{,}8;$ $a=[0.4–2];$ шаг : $\Delta a=0{,}4$
5	$Y=\begin{cases}ax+0{,}23x^2\cdot\log_2 a, & \text{если } a\le x;\\ \dfrac{xe^a\cdot\cos x}{\lvert a+x\rvert}, & \text{если } a>x\end{cases}$	$x=1{,}2;$ $a=[0{,}1–0{,}5];$ шаг : $\Delta a=0{,}1$
6	$Y=\begin{cases}a^3+\sqrt{a^2+\lvert x\rvert}\cdot\sin^2 x, & \text{если } a>x;\\ 2x^2+a^3\cdot\mathrm{tg}x, & \text{если } a\le x\end{cases}$	$x=1{,}5;$ $a=[1{,}2–2];$ шаг : $\Delta a=0{,}2$

Окончание табл. 4.1

1	2	3
7	$Y=\begin{cases}\dfrac{\sqrt{\lvert a+x\rvert}}{\lvert a^2+x\rvert}, & \text{если } a\le x;\\ e^{1,2}-\sqrt{\lvert ax\rvert}, & \text{если } a>x\end{cases}$	$x=0{,}4;$ $a=[0{,}7–1{,}1];$ шаг : $\Delta a=0{,}1$

8	$Y=\begin{cases}\dfrac{(x^2-a^2x)\cdot \mathrm{tg}x}{\sqrt{\lvert a+x\rvert}}, & \text{если } a\le x;\\ (a+x)\cdot e^{\sqrt{\lvert x+a\rvert}}, & \text{если } a>x\end{cases}$	$x=0{,}35;$ $a=[0{,}5–2{,}5];$ шаг : $\Delta a=0{,}5$
9	$Y=\begin{cases}\dfrac{(x+a)\cdot \mathrm{tg}x}{\ln\lvert x\rvert}, & \text{если } a\le x;\\ e^{ax}+a\sin x^2, & \text{если } a>x\end{cases}$	$x=0{,}54;$ $a=[0{,}2–1];$ шаг : $\Delta a=0{,}2$
10	$Y=\begin{cases}\sqrt{a^2+x^2}, & \text{если } a>x;\\ a-x\sqrt[3]{\sin^4\lvert x\rvert}, & \text{если } a\le x\end{cases}$	$x=0{,}7;$ $a=[0{,}3–1{,}5];$ шаг : $\Delta a=0{,}3$

З а д а н и е 4.2

1. Составить схему алгоритма для вычисления функций, приведенных в табл. 4.1, взяв исходные данные из табл. 4.2.
2. Выполнить пункты 2–5 задания 4.1.

Таблица 4.2

Исходные данные

Вариант	Исходные данные
1	$x=[0{,}35–0{,}43]$, шаг : 0,02; $a=[0{,}5–2{,}5]$, шаг : 0,5

2	$x = [0{,}54–0{,}58]$, шаг : 0,01; $a = [0{,}2–1]$, шаг : 0,2
3	$x = [0{,}7–1{,}9]$, шаг : 0,3; $a = [0{,}3–1{,}5]$, шаг : 0,3
4	$x = [0{,}8–1{,}4]$, шаг : 0,2; $a = [0{,}4–2]$, шаг : 0,4
5	$x = [1{,}2–2]$, шаг : 0,2; $a = [0{,}4–2]$, шаг : 0,4
6	$x = [1{,}5–1{,}9]$, шаг : 0,1; $a = [1{,}2–2]$, шаг : 0,2
7	$x = [0{,}7–1{,}3]$, шаг : 0,2; $a = [0{,}3–1{,}2]$, шаг : 0,2
8	$x = [0{,}3–0{,}7]$, шаг : 0,1; $a = [0{,}5–1{,}1]$, шаг : 0,2
9	$x = [0{,}35–0{,}43]$, шаг : 0,02; $a = [0{,}5–2{,}5]$, шаг : 0,5
10	$x = [0{,}54–0{,}58]$, шаг : 0,01; $a = [0{,}2–1]$, шаг : 0,2

5. МАССИВЫ. ОБРАБОТКА МАССИВОВ. ОПЕРАТОР С УПРАВЛЯЮЩИМ ПАРАМЕТРОМ

Массив – это структурированный тип данных, состоящий из фиксированного числа элементов, имеющих один и тот же тип.

Для описания массива предназначено словосочетание **Array of** (массив из). Возможны два способа описания массивов:

1. **Type**

<имя типа>=Array [tl, t2, ... , tn] of <t0>;
Var <имя массива>:<имя типа>;

2. Массив может быть описан и без представления типа в разделе описания типов данных.

Var <имя массива>: Array [t1, t2, ...,tn] of <t0>;

Здесь **t1, t2, ..., tn** – типы индексов массива. Количество индексов **n** определяет размерность массива.

Пример 5.1. Пусть в программе необходимо описать одномерный массив A{1, 5, 6, 7}.

Описание этого массива в соответствии с первым способом:

```
Type
TMas=array [1..4] of integer;
Var A: TMas;
```

Описание этого массива в соответствии со вторым способом:

```
Var A: array [1..4] of integer;
```

Пример 5.2. Пусть в программе необходимо описать двумерный массив (матрицу) *M*:

$$M = \begin{bmatrix} 1 & 2 & 5 \\ 6 & 7 & 2 \end{bmatrix}.$$

Описание этого массива:

```
Type TMas = array [1..2, 1..3] of integer;
Var M: TMas;
```

Для второго способа:

```
Var M: Array [1..2, 1..3] of integer;
```

После объявления массива каждый его элемент можно обработать, указав идентификатор (имя) массива и индекс элемента. Например, запись **Mas[2]**, **VektorZ[10]** позволяет обратиться ко второму элементу массива **Mas** и десятому элементу массива **VektorZ**. Индексированные элементы массива называются индексированными переменными и могут быть использованы так же, как и простые переменные.

Для обработки массивов, а также многократно выполняемых вычислений с целочисленными типами данных используется оператор цикла **FOR-do**.

Оператор повтора **for** состоит из заголовка и тела цикла. Он может быть представлен в двух форматах:

1. for <параметр цикла> := <S1> to <S2> do <оператор>;

2. for <параметр цикла> := <S1> downto <S2> do <оператор>;

S1 и S2 – выражения, определяющие соответственно начальное и конечное значения параметра цикла. **For...do – заголовок цикла; <оператор> – тело цикла.**

Значения элементам массива можно присвоить с помощью оператора присваивания, однако чаще всего они вводятся с экрана с помощью оператора **read** или **readln** с использованием оператора организации цикла **for**:

```
For I: =1 to 4 do readln (A [I]);
```

Значения двумерного массива вводятся с помощью вложенного оператора **for**:

```
For l:=l to 10 do
For J: =1 to 15 do
Readln (B [I, J]);
```

Можно ввести и значения отдельных элементов, а не всего массива:

Read (A [3]); Read (B[6,9]);

Вывод значения элементов массива выполняется аналогичным образом, но используются операторы **Write** или **Writeln**.

З а д а н и е 5.1

1. Составить схему алгоритма для вычисления функции, приведенной в табл. 5.1.

2. Составить программу для циклического вычислительного процесса на языке Паскаль.

3. Произвести расчеты на микроЭВМ.

4. Распечатать листинг программы.

5. Исходные данные, промежуточные и окончательные результаты расчета вывести на экран видеотерминала (дисплея) и печатающее устройство (принтер).

Таблица 5.1

Исходные данные

Вариант	Функции	Исходные данные
1	2	3
1	$Y_i = \begin{cases} (a_i - x_i)\cdot \sin x_i^2, & \text{если } a_i < x_i; \\ e^{\sqrt{\lvert x_i \rvert}} \cdot \operatorname{tg} a_i, & \text{если } a_i = x_i; \\ \sqrt{\lvert a_i \rvert} \cdot \cos x_i, & \text{если } a_i > x_i \end{cases}$	$a_1 = 1{,}2$; $x_1 = -12{,}3\cdot 10^2$; $a_2 = -15{,}1$; $x_2 = 83{,}4\cdot 10^{-1}$; $a_3 = 0{,}385$; $x_3 = 0{,}012$; $a_4 = 0.5$; $x_4 = 3{,}17\cdot 10^{-1}$

Продолжение табл. 5.1

1	2	3
2	$Y_i = \begin{cases} \left(a_i^2 - x_i^3\right)\cdot e^{\sqrt{x_i}}, & \text{если } a_i < x_i; \\ e^{\lvert x_i + a_i \rvert} \cdot \operatorname{tg}^2 x_i, & \text{если } a_i = x_i; \\ \sqrt{\lvert a_i \rvert} \cdot \log_2 \lvert x_i \rvert, & \text{если } a_i > x_i \end{cases}$	$a_1 = 3{,}57$; $x_1 = 149\cdot 10^{-1}$; $a_2 = 1{,}46$; $x_2 = 8{,}6\cdot 10^{-1}$; $a_3 = 0{,}59$; $x_3 = 59\cdot 10^{-1}$; $a_4 = -12{,}4$; $x_4 = 50\cdot 10^{-1}$

3	$Y_i = \begin{cases} e^{0,2} + \sqrt{\lvert a_i + x_i \rvert}, & \text{если } a_i < x_i; \\ \left(\lvert a_i + x_i \rvert\right)^2, & \text{если } a_i = x_i; \\ e^{\sqrt{\lvert x_i \rvert}} \cdot \sin^2 x_i, & \text{если } a_i > x_i \end{cases}$	$a_1 = 2{,}34$; $x_1 = 85 \cdot 10^{-1}$; $a_2 = 5{,}6$; $x_2 = 0{,}34 \cdot 10^{-1}$; $a_3 = -7{,}86$; $x_3 = -0{,}35 \cdot 10^{-1}$; $a_4 = 4{,}25$; $x_4 = 7{,}61 \cdot 10^{-1}$
4	$Y_i = \begin{cases} x_i \cdot e^{\sqrt{\lvert a_i + x_i \rvert}}, & \text{если } a_i < x_i; \\ a_i \cdot \sin x_i + 6{,}3 x_i^2, & \text{если } a_i = x_i; \\ a_i \cdot \operatorname{tg} x_i^2 / \ln x_i, & \text{если } a_i > x_i \end{cases}$	$a_1 = -2{,}65$; $x_1 = -0{,}14 \cdot 10^{-1}$; $a_2 = 4{,}63$; $x_2 = 8{,}6 \cdot 10^{-1}$; $a_3 = -5{,}25$; $x_3 = 0{,}04 \cdot 10^{-1}$; $a_4 = 16{,}1$; $x_4 = 0{,}9 \cdot 10^{-1}$
5	$Y_i = \begin{cases} 2x_i^4 + 16a_i - \operatorname{tg} x_i^3, & \text{если } a_i < x_i; \\ 0{,}5 x_i^2 + \sin\left(x_i^2 / a_i\right), & \text{если } a_i = x_i; \\ \sqrt{\lvert x_i \rvert} + \lvert a_i^3 + x_i^2 \rvert, & \text{если } a_i > x_i \end{cases}$	$a_1 = -2{,}75$; $x_1 = -40 \cdot 10^{-1}$; $a_2 = 3{,}86$; $x_2 = 6 \cdot 10^{-1}$; $a_3 = 7{,}85$; $x_3 = 113 \cdot 10^{-2}$; $a_4 = -4{,}5$; $x_4 = 2{,}5 \cdot 10^{-1}$
6	$Y_i = \begin{cases} (a_i + x_i)^2 \cdot \sin a_i, & \text{если } a_i < x_i; \\ \left(a_i + x_i^3\right) \cdot \operatorname{tg}\left(a_i / x_i\right), & \text{если } a_i = x_i; \\ \sqrt[3]{\lvert a_i - x_i \rvert}, & \text{если } a_i > x_i \end{cases}$	$a_1 = 23{,}58$; $x_1 = 153 \cdot 10^{-1}$; $a_2 = -0{,}3$; $x_2 = 0{,}82 \cdot 10^{-1}$; $a_3 = 1{,}14$; $x_3 = -0{,}37 \cdot 10^{-1}$; $a_4 = -0{,}75$; $x_4 = 0{,}57 \cdot 10^{-1}$
7	$Y_i = \begin{cases} \left(a_i^2 - x_i\right) \cdot \sin^2 x_i, & \text{если } a_i < x_i; \\ e^{\lvert x_i + a_i \rvert} \cdot \operatorname{tg}^2 a_i, & \text{если } a_i = x_i; \\ \sqrt[3]{\lvert a_i \rvert} \cdot \ln \lvert x_i \rvert, & \text{если } a_i > x_i \end{cases}$	$a_1 = 5{,}2$; $x_1 = -1{,}3 \cdot 10^{-1}$; $a_2 = 6{,}96$; $x_2 = 14{,}1 \cdot 10^{-1}$; $a_3 = 1{,}01$; $x_3 = 120 \cdot 10^{-2}$; $a_4 = -8{,}1$; $x_4 = 0{,}5 \cdot 10^{-1}$

Окончание табл. 5.1

1	2	3

8	$Y_i = \begin{cases} \left(a_i^2 - x_i\right) \cdot e^{\sqrt{x_i}} / \left(a_i^2 + 1\right), & \text{если } a_i < x_i; \\ e^{\lvert x_i + a_i \rvert} \cdot \sin^2 x_i, & \text{если } a_i = x_i; \\ \sqrt{\lvert a_i + 1 \rvert} \cdot \log_2 \lvert x_i \rvert, & \text{если } a_i > x_i \end{cases}$	$a_1 = 3{,}57$; $x_1 = 149$; $a_2 = 1{,}46$; $x_2 = 8{,}6$; $a_3 = 0{,}59$; $x_3 = 59$; $a_4 = -12{,}4$; $x_4 = 50$
9	$Y_i = \begin{cases} e^{0,2} + \sqrt{\lvert a_i + x_i \rvert}, & \text{если } a_i < x_i; \\ \left(\lvert a_i + x_i \rvert\right)^2 - \log_2 x_i^2, & \text{если } a_i = x_i; \\ e^{\sqrt{\lvert x_i \rvert}} \cdot \sin^2 x_i, & \text{если } a_i > x_i \end{cases}$	$a_1 = 2{,}34$; $x_1 = 85$; $a_2 = 5{,}6$; $x_2 = 0{,}34$; $a_3 = -7{,}86$; $x_3 = -0{,}354$; $a_4 = 4{,}25$; $x_4 = 7{,}61$
10	$Y_i = \begin{cases} \left(a_i - x_i^2\right) \cdot \sin^3 x_i, & \text{если } a_i < x_i; \\ e^{\sqrt{\lvert x_i \rvert}} \cdot \log_2 a_i, & \text{если } a_i = x_i; \\ \sqrt[3]{\lvert a_i \rvert} \cdot \cos x_i, & \text{если } a_i > x_i \end{cases}$	$a_1 = 1{,}2$; $x_1 = -12{,}3 \cdot 10^2$; $a_2 = -15{,}1$; $x_2 = 83{,}4 \cdot 10^{-1}$; $a_3 = 0{,}385$; $x_3 = 0{,}012$; $a_4 = 0{,}5$; $x_4 = 3{,}17 \cdot 10^{-1}$;

З а д а н и е 5.2

1. Составить схему алгоритма для вычисления функции, приведенной в табл. 5.2, домножив каждое значение функции на дополнительный член $\sum\left(\lvert a_i - x_i \rvert\right)$.

2. Выполнить пункты 2–5 задания 5.1.

6. ОБРАБОТКА ДВУМЕРНЫХ МАССИВОВ

З а д а н и е 6.1

1. Составить схему алгоритма для вычисления произведения вектора B на матрицу M третьего порядка. Численные значения элементов вектора и матрицы приведены в табл. 6.1.

2. Составить программу на языке Паскаль, реализующую построенный алгоритм.

3. Произвести расчеты на микроЭВМ.

4. Распечатать листинг программы.

5. Исходные данные, промежуточные и окончательные результаты расчета вывести на экран видеотерминала (дисплея) и печатающее устройство (принтер).

Указание: элементы результирующего вектора R вычисляются по формуле

$$r_i = \sum b_i m_{ij},$$

где $i = 1, 2, 3, \ldots, n$;

b_i – элементы вектора B;

m_{ij} – элементы матрицы M.

Результат получается при использовании двух вложенных циклов.

Таблица 6.1

Исходные данные

Вариант	Элементы вектора	Элементы матрицы
1	2	3
1	$B_1 = \{1{,}21;\ 3{,}42;\ 4{,}51\}$	$M_1 = \begin{matrix} 0{,}24 & 0{,}27 & 0{,}31 \\ 0{,}43 & 0{,}84 & 0{,}92 \\ 1{,}21 & 1{,}63 & 1{,}98 \end{matrix}$
2	$B_2 = \{2{,}1;\ 3{,}40;\ 4{,}14\}$	$M_2 = \begin{matrix} 4{,}05 & 5{,}15 & 6{,}13 \\ 5{,}91 & 8{,}05 & 9{,}06 \\ 7{,}17 & 9{,}29 & 9{,}19 \end{matrix}$

Окончание табл. 6.1

1	2	3

3	B_3 = {0,56; 0,75; 0,89}	M_3 = 1,22 1,65 2,05 0,45 0,67 0,81 2,31 3,07 4,05
4	B_4 = {3,2; 4,31; 6,07}	M_4 = 0,95 1,32 2,40 40 3,72 4,12 2,17 3.60 5,20
5	B_5 = {0,75; 1,25; 2,5}	M_5 = 3,14 4,09 5,10 2,17 3,05 4,10 1,28 2,40 3,20
6	B_6 = {3,1; 4,25; 5,4}	M_6 = 0,34 0,65 0,85 1,34 2,17 3,05 0,75 0,85 4,07
7	B_7 = {0,8; 0,9; 1,3}	M_7 = 2,40 3,10 4,35 0,60 0,74 0,95 3,40 4,02 4,90
8	B_8 = {4,4; 5,45; 6,15}	M_8 = 0,32 0,47 0,62 0,85 0,70 0,25 3,52 4,17 2,10
9	B_9 = {1,21; 3,42; 4,51}	M_9 = 0,24 0,27 0,31 0,43 0,84 0,92 1,21 1,63 1,98
10	B_{10} = {2,1; 3,40; 4,14}	M_{10} = 4,05 5,15 6,13 5,91 8,05 9,06 7,17 9,29 9,19

З а д а н и е 6.2

1. Составить схему алгоритма для вычисления произведения матрицы M третьего порядка на матрицу P третьего порядка. Численные значения матрицы M взять согласно табл. 6.1, а численные значения матрицы P – из табл. 6.2.

2. Выполнить пункты 2–5 задания 5.1.

Указание: Элементы результирующей матрицы С вычисляются по формуле

$$c_{ij} = \sum m_{ik} p_{kj},$$

где ij = 1, 2, 3, …, n;

m_{ik} – элементы матрицы M;

p_{kj} – элементы матрицы P,

c_{ij} – элементы результирующей матрицы C.

Результат получается при использовании трех вложенных циклов.

Таблица 6.2

Исходные данные

Вариант	Матрица	Вариант	Матрица
1	$P_1 = \begin{matrix} 4{,}05 & 5{,}15 & 6{,}13 \\ 5{,}91 & 8{,}05 & 9{,}06 \\ 7{,}17 & 9{,}29 & 9{,}19 \end{matrix}$	6	$P_6 = \begin{matrix} 1{,}15 & 1{,}91 & 2{,}15 \\ 3{,}05 & 3{,}41 & 4{,}70 \\ 4{,}23 & 5{,}17 & 6{,}06 \end{matrix}$
2	$P_2 = \begin{matrix} 0{,}17 & 0{,}21 & 0{,}72 \\ 1{,}15 & 1{,}23 & 1{,}91 \\ 2{,}10 & 3{,}20 & 4{,}30 \end{matrix}$	7	$P_7 = \begin{matrix} 6{,}10 & 6{,}80 & 7{,}20 \\ 0{,}23 & 0{,}40 & 0{,}80 \\ 1{,}15 & 2{,}80 & 3{,}25 \end{matrix}$
3	$P_3 = \begin{matrix} 2{,}15 & 2{,}91 & 2{,}23 \\ 4{,}05 & 4{,}41 & 4{,}70 \\ 5{,}23 & 6{,}17 & 7{,}00 \end{matrix}$	8	$P_8 = \begin{matrix} 4{,}15 & 3{,}20 & 2{,}40 \\ 0{,}50 & 0{,}80 & 1{,}24 \\ 7{,}20 & 8{,}41 & 9{,}50 \end{matrix}$
4	$P_4 = \begin{matrix} 1{,}15 & 1{,}91 & 2{,}15 \\ 3{,}05 & 3{,}41 & 4{,}70 \\ 4{,}23 & 5{,}17 & 6{,}06 \end{matrix}$	9	$P_9 = \begin{matrix} 4{,}05 & 5{,}15 & 6{,}13 \\ 5{,}91 & 8{,}05 & 9{,}06 \\ 7{,}17 & 9{,}29 & 9{,}19 \end{matrix}$
5	$P_5 = \begin{matrix} 0{,}84 & 0{,}97 & 0{,}61 \\ 0{,}67 & 0{,}84 & 0{,}92 \\ 1{,}30 & 2{,}60 & 3{,}10 \end{matrix}$	10	$P_{10} = \begin{matrix} 0{,}17 & 0{,}21 & 0{,}72 \\ 1{,}15 & 1{,}23 & 1{,}91 \\ 2{,}10 & 3{,}20 & 4{,}30 \end{matrix}$

7. ОБРАБОТКА ОДНОМЕРНЫХ МАССИВОВ С ИСПОЛЬЗОВАНИЕМ ПОДПРОГРАММ

Часто в программе обнаруживаются однотипные участки, которые выполняют одни и те же вычисления, но с различными данными. Такие участки программ целесообразно оформлять в виде подпрограмм.

В Турбо Паскале выделяют два вида подпрограмм: процедуры и функции. Структура процедур и функций такая же, как и структура основной программы, т. е. состоит из раздела описаний и раздела операторов.

Описание процедур имеет вид

Procedure имя (список формальных параметров);
<раздел описаний>
Begin
<раздел операторов>
End;

и помещается в основной программе (program) в разделе описаний.

Здесь **имя** – имя процедуры. **Раздел описаний**, как и в основной программе (program), содержит разделы **uses**, **label**, **const**, **type**, **var** и раздел процедур и функций. **Формальные параметры** представляют собой список переменных с указанием их типа, которые отделяются друг от друга точкой с запятой. Эти переменные не описываются в разделе описаний процедур. Допускается использование процедур без списка параметров.

Параметры процедуры могут быть трех видов:

1. Параметры значения (входные параметры).
2. Параметры переменные (выходные параметры).
3. Параметры процедурного типа.

Описание входных параметров процедуры в списке формальных параметров имеет такой вид:

Список переменных 1: тип1; Список переменных 2: тип 2; ....

Соответственно описание выходных параметров:

Var список переменных 1: тип 1; var список перем. 2: тип 2; ...

Вызов процедуры в основной программе производится оператором вида

Имя процедуры (фактические параметры);.

Здесь параметры представляют собой список фактических параметров, перечисленных через запятую (без указания их типа). Между формальными и фактическими параметрами должны быть соответствия по количеству параметров, порядку их следования и типу данных. Имена соответствующих параметров могут быть одинаковыми или разными.

Входными фактическими параметрами могут быть константы, переменные, выражения.

Выходными фактическими параметрами могут быть переменные.

При использовании в качестве параметров подпрограмм данных структурированного типа (массивы, множества, записи) в основной программе необходимо предварительно описать имя типа этих данных в разделе **Type**, которые потом указываются в списке формальных параметров подпрограммы.

З а д а н и е

Составить программу, состоящую из трех подпрограмм:
- подпрограмма ввода массива;
- подпрограмма вывода массива;
- подпрограмма решений заданий, приведенных в таблице.

Исходные данные

Вариант	Задание	Исходные данные
1	2	3
1	В заданном массиве найти отрицательные элементы и определить их сумму	Даны три вещественных массива: A[A1, A2, …, A9], B[B1, B2, …, B5], C[C1, C2, …, C4]
2	В заданном массиве найти положительные элементы и определить из них наибольшее	Даны три вещественных массива: A[A1, A2, …, A7], B[B1, B2, …, B8], C[C1, C2, …, C5]
3	В заданном массиве вычислить сумму элементов с четными индексами	Даны три вещественных массива: A[A1, A2, …, A4], B[B1, B2, …, B8], C[C1, C2, …, C6]
4	В заданном массиве найти среднеарифметическое положительных элементов	Даны три вещественных массива: A[A1, A2, …, A8], B[B1, B2, …, B5], C[C1, C2, …, C4]
5	В заданном массиве найти положительные элементы и определить их произведение	Даны три вещественных массива: A[A1, A2, …, A9], B[B1, B2, …, B6], C[C1, C2, …, C5]
6	В заданном массиве найти наименьшее значение модуля разности между соседними элементами	Даны три вещественных массива: A[A1, A2, …, A7], B[B1, B2, …, B5], C[C1, C2, …, C6]

Окончание таблицы

1	2	3
7	В заданном массиве найти все отрицательные элементы, подсчитать их число и переписать подряд в новый массив	Даны три вещественных массива: A[A1, A2, …, A5], B[B1, B2, …, B8], C[C1, C2, …, C4]
8	В заданном массиве найти наибольшее значение модуля разности между соседними элементами	Даны три вещественных массива: A[A1, A2, …, A6], B[B1, B2, …, B5], C[C1, C2, …, C9]
9	В заданном массиве найти элементы с индексами, кратными трем, и найти их произведение	Даны три вещественных массива: A[A1, A2, …, A4], B[B1, B2, …, B6], C[C1, C2, ..., C9]
10	В заданном массиве найти элементы с индексами, кратными трем, и найти среди них наибольшее	Даны три вещественных массива: A[A1, A2, …, A9], B[B1, B2, …, B6], C[C1, C2, …, C4]

8. ПОДПРОГРАММЫ ТИПА FUNCTION

И PROCEDURE

Другой вид подпрограмм в языке Турбо Паскаль – функция – оформляется аналогично процедуре и отличается от нее заголовком, общий вид которого такой:

Function имя(список формальных параметров):тип;.

Здесь **тип** – тип возвращаемого функцией результата.

Другие отличительные особенности функций следующие:

1. Функция имеет только один результат выполнения, но может иметь несколько входных параметров.

2. Результат обозначается именем функции. Поэтому в разделе операторов функции обязательно должен присутствовать оператор присваивания, в левой части которого стоит имя этой функции.

3. В заголовке функции обязательно должен быть указан тип функции.

4. Вызов функции в основной программе осуществляется непосредственно внутри выражения по ее имени с указанием фактических параметров.

П р и м е р.

Оформить программу вычисления суммы S от 1 до n в виде функции.

```
Function Sm(N:integer):integer;
Var S,I:integer;
begin S:=0;
for I:=1 to N do S:=S+I;
Sm:=S
end;
```

Тогда в основной программе можно использовать следующий оператор:

```
X:=Sm(10);
```

который присваивает переменной *x* значение суммы элементов I от 1 до 10.

Оператор **Y:=Sm(10)/Sm(20);** присваивает переменной Y значение $\sum_{i=1}^{10} i \Big/ \sum_{i=1}^{20} i$.

З а д а н и е 8.1

1. Составить схему алгоритма вычисления функции.

2. $Y = (e^{X1+Y2} + e^{X2+Y1}) \cdot p$ с использованием подпрограммы типа Procedure для вычисления корней квадратных уравнений (x_1, x_2, y_1, y_2) вида $az^2 + bz + c = 0$, приведенных в табл. 8.1.

3. Составить программу на языке Паскаль, реализующую построенный алгоритм.

4. Произвести расчеты на микроЭВМ.

5. Распечатать листинг программы.

6. Исходные данные промежуточные и окончательные результаты расчета вывести на экран видеотерминала (дисплея) и на печатающее устройство (принтер).

Таблица 8.1

Исходные данные

Вариант	Квадратные уравнения	Исходные данные (множитель)
1	$x^2 - 6x + 5 = 0$; $y^2 + 5y + 4 = 0$	$P = 2{,}25$
2	$0{,}6x^2 + 3{,}2x - 8{,}4 = 0$; $3y^2 + 7y + 4 = 0$	$P = 4{,}35$
3	$x^2 - 7x + 12 = 0$; $2{,}5y^2 + 12{,}5y + 10 = 0$	$P = 7{,}75$
4	$2{,}5x^2 + 10x + 7{,}5 = 0$; $y^2 - 4y + 3 = 0$	$P = 8{,}625$
5	$x^2 + 4x + 3 = 0$; $y^2 + 5y + 6 = 0$	$P = 10{,}25$
6	$x^2 - 3x + 2 = 0$; $y^2 + 12y + 10 = 0$	$P = 11{,}55$
7	$x^2 - 6x + 5 = 0$; $y^2 - 13y + 42 = 0$	$P = 14{,}85$
8	$x^2 - 11x + 30 = 0$; $2y^2 - 3y - 2 = 0$	$P = 15{,}45$
9	$x^2 - 6x + 5 = 0$; $y^2 + 5y + 4 = 0$	$P = 2{,}25$
10	$0.6x^2 + 3.2x - 8.4 = 0$; $3y^2 + 7y + 4 = 0$	$P = 4{,}35$

З а д а н и е 8.2

Выполнить задание 8.1 (п. 1–5). В качестве множителя p взять сумму или произведение элементов массива, указанных в табл. 8.2. Вычисление сумм и произведений оформить в виде подпрограмм типа Function.

Таблица 8.2

Исходные данные

Вариант	Вычисление P	Исходные данные
1	$P = \prod_{i=0}^{n} C_i$	$c_0 = 0{,}95$; $c_1 = 1{,}21$; $c_2 = -3{,}05$; $c_3 = 2{,}75$
2	$P = \sum_{i=0}^{n} C_i$	$c_0 = 1{,}65$; $c_1 = 2{,}6$; $c_2 = 3{,}5$; $c_3 = 5{,}25$
3	$P = \prod_{i=1}^{n} C_i$	$c_1 = 1$; $c_2 = 2$; $c_3 = 3{,}21$; $c_4 = 4$
4	$P = \sum_{i=1}^{n} C_i$	$c_1 = 0{,}55$; $c_2 = 0{,}67$; $c_3 = 1$; $c_4 = 2{,}5$
5	$P = \prod_{i=0}^{n} C_i$	$c_0 = 2{,}75$; $c_1 = 3{,}25$; $c_2 = 4{,}1$; $c_3 = 0{,}84$
6	$P = \sum_{i=0}^{n} C_i$	$c_0 = 3{,}2$; $c_1 = 4{,}1$; $c_2 = 0{,}05$; $c_3 = 0{,}085$
7	$P = \prod_{i=1}^{n} C_i$	$c_1 = 0{,}25$; $c_2 = 0{,}47$; $c_3 = 3{,}15$; $c_4 = 5{,}25$
8	$P = \sum_{i=1}^{n} C_i$	$c_1 = 2$; $c_2 = 5$; $c_3 = 3{,}5$; $c_4 = 2{,}5$
9	$P = \prod_{i=0}^{n} C_i$	$c_0 = 0{,}95$; $c_1 = 1{,}21$; $c_2 = -3{,}05$; $c_3 = 2{,}75$
10	$P = \sum_{i=0}^{n} C_i$	$c_0 = 1{,}65$; $c_1 = 2{,}6$; $c_2 = 3{,}5$; $c_3 = 5{,}25$

9. АЛГОРИТМИЗАЦИЯ ВЫЧИСЛИТЕЛЬНЫХ ПРОЦЕССОВ. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

З а д а н и е

1. Составить схему алгоритма и программу для решения нелинейного уравнения, указанного в таблице. Уравнение должно быть решено тремя методами: простой итерацией, методом Ньютона и методом деления пополам с точностью $\varepsilon = 10^{-4}$.

2. Распечатать листинг программы, результаты расчета и результаты сопоставления указанных методов по эффективности (в частности, по быстродействию).

Исходные данные

Вариант	Уравнения	Область существования корня
1	$x = e^{-x}$	[–0,1; 0,7]
2	$x - \sin x = 0{,}25$	[0,67; 2,34]
3	$\ln x - x + 1{,}8 = 0$	[1,7; 3,3]
4	$3\sin\sqrt{x} + 0{,}35x = 3{,}8$	[1,8; 3,2]
5	$x + \sin 1/x = 2$	[1,2; 2,0]
6	$0{,}25x^3 + x = 1{,}2502$	[0,1; 2,1]
7	$1 - x + \sin x - \ln(1+x) = 0$	[–0,2; 1,7]
8	$0{,}6 \cdot 3^x - 2{,}3x - 3 = 0$	[1,9; 3,1]
9	$3x + e^x - e^x = 14$	[0,8; 3,2]
10	$e^x + \ln x - 10x = 0$	[2,7; 4,3]

Программа должна содержать следующее:

- несколько комментариев;
- вывод на экран и принтер фамилии студента;

- номера группы и варианта задания;
- операторы, реализующие вычисления;
- распечатку на экран и принтер исходных данных и результатов расчета.

Задача должна быть решена с использованием подпрограмм.

10. ИСПОЛЬЗОВАНИЕ МНОЖЕСТВ И ТИПОВ ДАННЫХ, ЗАДАННЫХ ПЕРЕЧИСЛЕНИЕМ

Множество – это структурированный тип данных, представляющий собой набор взаимосвязанных по какому-либо признаку или группе признаков объектов, которые можно рассматривать как единое целое. Каждый объект в множестве называется элементом множества. Все элементы множества должны принадлежать одному из порядковых типов. Этот тип называется базовым. Если множество не имеет элементов, то оно называется пустым и обозначается как [ ]. Значения элементов множества указываются в квадратных скобках. Как и массивы, множества могут описываться двумя способами:

1) **TYPE <имя типа>= SET OF <баз. тип>;**
   VAR <имя множества,…> : <имя типа> ;

2) **VAR : <имя множества,… > : SET OF <баз. тип>;**,

где **SET OF** – множество из…;

<баз. тип> – базовый тип элементов множества, в качестве которого может использоваться любой порядковый тип, кроме WORD, INTEGER, LONGINT.

При работе с множествами PASCAL допускает операции отношения, объединения, пересечения, разности множеств и операции **IN**.

Операция 'равно'(=). Два множества **A** и **B** считают равными, если они состоят из одних и тех же элементов. В сравниваемых множествах порядок следования значения не имеет.

Операция 'не равно'(<>). Два множества считаются не равными, если они отличаются по значению хотя бы одного элемента.

Операция 'больше или равно'(>=). Используется для определения принадлежности множеств. Результат операции **A**>=**B** равен true, если все элементы множества **B** содержатся в множестве **A**. В противном случае результат равен false.

Операция' меньше или равно'(<=). Аналогична предыдущей операции.

Операция IN. Используется для проверки принадлежности какого-либо значения указанному множеству. Обычно применяется в условных операторах.

При использовании операции **IN** множество в квадратных скобках не обязательно предварительно описывать в разделе описаний. Операция **IN** позволяет наглядно производить сложные проверки условий. Например, выражение **if (a=1) or (a=2) or (a=3) or (a=4) or (a=4) or (a=5)** then можно заменить более коротким

if a IN [1..5] then.<оператор>;.

Объединение множеств (+), (AUB). Объединением двух множеств является третье множество, содержащее элементы обоих множеств.

Пересечение множеств (*), (A∩B). Пересечением двух множеств является третье множество, которое содержит элементы, входящие одновременно в оба множества.

Разность множеств (-), (A\B). Разностью двух множеств является третье множество, которое содержит элементы первого множества, не входящие во второе множество.

З а д а н и е

Составить программу, в которой необходимо описать множество или тип данных, заданные перечислением, и обработать эти данные в зависимости от задания, приведенного в таблице.

Исходные данные

Вариант	Задание
1	2
1	Даны три множества X_1, X_2 и X_3 , содержащие целые числа из диапазона 1–100. Известно, что мощность каждого из этих множеств равна 10. Сформировать новое множество $Y = (X_1 \cup X_2) \cap (X_1 \setminus X_3)$, из которого вы-делить подмножество нечетных чисел. На экран и принтер вывести исходные и полученные множества
2	Даны три множества X_1, X_2 и X_3 , содержащие целые числа из диапазона 1–100. Известно, что мощность каждого из этих множеств равна 10. Сформировать новое множество $Y = (X_1 \cup X_2) \setminus (X_1 \cap X_3)$ и вывести на экран его мощность. Проверить, есть ли в множестве Y числа, делящиеся на 6 без остатка. Результаты вывести на экран и печать
3	Даны два множества M и N, состоящие из 10 целых чисел диапазона 1–100. Из данных множеств выделить соответственно подмножества M_1 чисел, делящихся на 3 без остатка, и N_1 чисел, делящихся на 2 без остатка. Множества M и N описать как типизированные константы. На печать и экран вывести исходные данные, полученные данные, M_1, N_1, мощность и значения элементов множества $MN = M_1 \cap N_1$
4	Даны три множества X_1, X_2 и X_3, содержащие целые числа из диапазона 100–200. Известно, что мощность каждого из этих множеств равна 10. Сформировать новое множество $Y = (X1 \cap X2) \cup (X_1 \cup X_3)$ и найти его мощность. На печать вывести множества X_1, X_2, X_3, Y и мощность множества Y
5	Даны три множества $X_1 = \{1, 2, 3, \ldots, 20\}$, $X_2 = \{10, 11, 12, \ldots, 30\}$ и $X3 = \{1, 3, 5, \ldots, 19, 21\}$. Сформировать множество $Y = (X_1 \cup X_2) \cap (X_1 \cup X_3) \setminus (X_2 \cup X_3)$, из ко-торого выделить подмножество Y_1 чисел, делящихся на 4 без остатка. На печать и экран вывести

	множество Y и мощность множества Y_1

Продолжение таблицы

1	2
6	В восточном календаре года носят названия следующих животных: крыса, бык, тигр, заяц, дракон, змея, лошадь, овца, обезьяна, петух, собака, свинья. Кроме того, через каждые два года меняется цвет в следующем порядке: синий, красный, желтый, белый и черный. Таким образом, 1992 – год черной обезьяны, 1993 – год черного петуха, 1994 – год синей собаки и т. д. Написать программу, которая переводит заданный год в его название по восточному календарю (используйте «тип перечисление»). Результаты выводятся на экран и печать
7	Даны три множества $X_1 = \{1, 2, 3, \ldots, 20\}$, $X_2 = \{10, 20, \ldots, 190, 200\}$ и $X_3 = \{10, 11, 12, \ldots, 40\}$. Сформировать множество $Y = (X_2 \cap X_3) \setminus ((X_1 \cap X_2) \cup (X_1 \cap X_3))$ и множество Y_1, состоящее из элементов Y, деленных на 2. Если полученное в результате деления число не целое, то округлить его до ближайшего целого. На печать вывести исходные множества X_1, X_2, X_3 и полученные множества Y и Y_1
8	Даны три множества $X_1 = \{T_2, T_4, T_6, T_8, T_{10}\}$, $X_2 = \{T_1, T_2, T_3, T_4, T_5\}$ и $X_3 = \{T_2, T_3, T_5, T_7, T_8\}$. Сформировать множество $Y = (X_2 \setminus X_3) \cup (X_1 \setminus X_3)$. На печать вывести исходные множества X_1, X_2, X_3, множество Y и его мощность. Исходные множества описать как типизированные константы
9	Разработать программу для определения, какому алфавиту (латинскому или русскому) принадлежит введенный с клавиатуры символ. На печать вывести выведенный символ с комментарием, например:

	Набран символ «А» на русском регистре

Окончание таблицы

1	2
10	Разработать учебную программу для проверки знаний студентами алфавита языка Turbo Pascal. Программа должна формировать запрос на вывод очередного символа, проверять принадлежит ли он алфавиту языка Turbo Pascal, нет ли попытки повторно ввести один и тот же символ, выводить соответствующие комментарии и оценку (например: введены все символы верно – отлично, не более двух ошибок – хорошо и т. д.)

11. ОБРАБОТКА СТРОК

Строки, как и числовые данные, подразделяются на константы и переменные. Строковая константа – это последовательность символов, заключенных в апострофы. Строковые константы могут быть описаны в разделе констант:

Const St=’строка’;

Описание типа

Переменную строкового типа можно описать двумя способами.

Формат:

1. Type<имя типа>=string[n] {n-длина строки}
var <идентификатор,…>:<имя типа>;.
2. var <идентификатор,…>:string[n];

Определение строкового типа устанавливает максимальное количество символов, которое может содержать строка. Количество символов в строке должно быть не меньше 1 и не больше 255, т. е. $1 \le n \le 255$.

Строковую переменную можно также описывать следующим образом:

var<имя переменной>:string;.

Это описание определяет строковую переменную максимально возможной длины (255 символов).

Операции над строками

1. Операция присваивания:

Общий вид этой операции следующий:

имя строковой переменной := строковое выражение;

2. Вывод и ввод значений строковых переменных осуществляется без апострофов.

3. Операция сцепления (+) применяется для сцепления нескольких строк в одну результирующую строку.

Выражение: ‘ГР.’+’ 107’+’410’. Результат: ’ГР. 107410’

Стандартные подпрограммы для обработки строк

Delete(St,Poz,N) – удаление N символов строки St, начиная с позиции Poz. Если значение Poz > 255 возникает ошибка.

Pos(Str1,Str2) – функция обнаруживает первое появление в строке Str2 подстроки Str1. Результат имеет целочисленный тип и равен номеру той позиции, где находится первый символ подстроки Str1. Если в Str2 не найдено подстроки Str1, то результат равен 0.

UpCase(Ch) – преобразует строчную букву в прописную. Обрабатывает буквы только латинского алфавита.

Str(IBR,St) – преобразование числового значения величины IBR и помещение результата в строку St. После IBR может записываться формат, аналогичный формату вывода. В зависимости от описания IBR может быть целым либо вещественным.

Val(St,IBR,Cod) – преобразует значение St в величину целочисленного или вещественного типа и помещает результат в IBR. Значение St не должно содержать незначащих пробелов в середине и в конце строки. Cod – целочисленная переменная. Если во время операции преобразования ошибки не обнаружено, значение Cod = 0, если ошибка обнаружена, то Cod будет содержать номер позиции первого ошибочного символа, а значение IBR не определено.

Copy(St,Poz,N) – выделяет из St подстроку длиной N символов, начиная с позиции Poz.

Length(St) – вычисляет количество символов в строке St. Результат имеет целочисленный тип.

З а д а н и е

1. Составить схему алгоритма и решить следующую задачу: ввести строку символов, состоящую из отчества, имени и фамилии, номера группы. Слова разделить пробелами.

2. Используя процедуры и функции для обработки строковых данных, подсчитать общее количество символов в данной строке и количество символов в каждом слове.

3. Преобразовать данную строку так, чтобы она содержала слова в следующей последовательности: фамилия, имя, отчество, номер группы.

Программа должна содержать:

– несколько комментариев;

– ввод-вывод на экран и принтер исходных данных;

– вывод на экран и принтер результатов обработки.

12. ТЕКСТОВЫЕ ФАЙЛЫ. ЗАПИСЬ В ФАЙЛ. ДОБАВЛЕНИЕ В ФАЙЛ. СЧИТЫВАНИЕ ИЗ ФАЙЛА

Текстовые файлы предназначены для хранения текстовой информации, компоненты текстового файла могут иметь переменную длину. Доступ к каждой строке данных осуществляется лишь последовательно, начиная с первой. При создании текстового файла в конце каждой строки отводится специальной признак **Eoln**, а в конце файла – **Eof** (конец файла).

Для описания используют стандартный тип **Text**, то есть **FV: Text.** С момента описания все операции с текстовым файлом выполняется посредством переменной файлового типа **(FV).**

Append (FV) – открывает текстовый файл для расширения. При этом указатель файла устанавливается в его конец.

Eoln(FV)):Boolean – возвращает true, если во входном текстовом фай-ле достигнут маркер конца строки.

Eof(FV):Boolean – функция тестирует конец файла. Возвращает true, если указатель стоит в конце файла, false – в противном случае.

Readln(FV,переменные) – чтение файла.

Writeln(FV,переменные) – запись в файл.

```
П р и м е р  12.1. {Запись в текстовый файл}
var  F: Text;
X:Array[1..5] of Real;
A,b,y:Real;
Begin
…….
 Assign(F,’d:\A1.txt’);
 Rewrite(F);
```

```
  Writeln(F,'исходные данные');
  Writeln(F, 'a=', a:0:2, ' b=',b:0:2 );
  Writeln(F, 'результат y=',y:0:2);
  Writeln(F,'исходный массив X: ');
  For I:=1 to 5 do
      writeln(F, 'X[',I,']=',X[I]:0:2,' ');
      Close(F);
end.
```

П р и м е р 12.2. {Чтение текстового файла}

```
var F: Text;
    S: String;
begin
      Assign(F, 'd:\A1.dat');
      Reset(F);
      While not EOF(F) do begin
        Readln (F, S);
        Writeln (S)
end;
…
END.
```

З а д а н и е

1. Составить схему алгоритма и программу, состоящую из подпрограмм записи в файл и считывания из файла.

2. С помощью подпрограммы решить уравнение

$$Z = \sum_{i=0}^{n} a_i \ ,$$

где a_i – элемент массива, считанный из файла.

Исходные данные

Вариант	Исходный массив
1	2

1	0; 0; 0; 0,563; 0,98; 1,32; 1,78; 1,94; 2,06; 2,16; 2,24; 2,3; 2,34; 2,36; 2,4; 2,42; 2,44
2	0; 0; 0,56; 0,972; 1,26; 1,46; 1,62; 1,72; 1,8; 1,86; 1,9; 1,92; 1,94

Окончание таблицы

1	2
3	0; 0; 0; 0,46; 0,82; 1,12; 1,36; 1,56; 1,74; 1,88; 1,98; 2,06; 2,16; 2,22; 2,28; 2,32; 2,34; 2,38; 2,4; 2,42; 2,44; 2,46
4	0; 0; 0; 0; 0,28; 0,32; 0,46; 0,56; 0,64; 0,68; 0,76; 0,8; 0,84; 0,86; 0,88; 0,9; 0,92; 0,94; 0,96; 0,98; 1,0
5	0; 0; 0,2; 0,38; 0,52; 0,66; 0,76; 0,86; 0,94; 1,02; 1,04; 1,06; 1,14;1,16; 1,22; 1,26; 1,3; 1,32; 1,34; 1,36; 1,38; 1,40; 1,42; 1,44; 1,46; 1,48; 1,50; 1,52; 1,54
6	0; 0; 0; 0,54; 0,96; 1,36; 1,64; 1,88; 2,02; 2,26; 2,38; 2,50; 2,52; 2,66; 2,72; 2,76; 2,80; 2,82; 2,90; 2,92; 2,94; 2,96
7	0; 0; 0; 0; 0,36; 0,66; 0.90; 1,1; 1,26; 1,38; 1,50; 1,58; 1,66; 1,72; 1,76; 1,80;1,82; 1,86; 1,90; 1,92; 1,94; 1,96
8	0; 0; 0; 0,6; 1,12; 1,56; 1,94; 2,28; 2,52; 2,74; 2,94; 3,1; 3,24; 3,36; 3,44; 3,54; 3,60; 3,66; 3,72; 3,76; 3,80; 3,82; 3,84; 3,86; 3,88; 3,92
9	0; 0; 0; 0,563; 0,98; 1,32; 1,58; 1,78; 1,94; 2,06; 2,16; 2,24; 2,3; 2,34; 2,36; 2,4; 2,42; 2,44
10	0; 0; 0,56; 0,972; 1,26; 1,46; 1,62; 1,72; 1,8; 1,86; 1,9; 1,92; 1,94

13. СОЗДАНИЕ И ИСПОЛЬЗОВАНИЕ СОБСТВЕННЫХ МОДУЛЕЙ

Модуль – это программная единица, текст которой компилируется независимо. Модуль включает определения констант, типов данных, переменных, процедур и функций, доступных для использования в вызывающих программах и, возможно, некоторые исполняемые операторы инициирующей части, т. е. каждый модуль – это библиотека объявлений, которую можно вставить и использовать внутри программы, что позволяет разделить программу на части и компилировать их отдельно. Структура модуля похожа на структуру программы, но имеет и отличия. Модуль состоит из трех разделов: интерфейса, реализация и инициализации.

Структура модуля:

Unit <имя модуля>;
Interface { интерфейсная часть }
Uses <список модулей>;
{ Открытые объявления }
Label - подраздел объявлений доступных глобальных меток
Const ---------“--------“--------“---------“---------“------ констант
Type ---------“--------“--------“---------“---------“--------- типов
Var ---------“--------“--------“---------“---------“------ переменных
Procedure - заголовки доступных процедур
Function ---------“--------“-------- функций
Implementation { реализационная часть }
uses < список модулей >;
{Собственные объявления}
Label - подраздел объявлений скрытых глобальных меток
Const ---------“--------“--------“---------“---------“------ констант

Type ---------“--------“--------“---------“---------“------ типов
Var ---------“--------“--------“---------“---------“------ переменных
{реализация процедур и функций}
Procedure - тела доступных и скрытых процедур
Function---------“--------“--------“--------- функций
begin
{код инициализации}
end.

Заголовок модуля – слово **Unit**, за которым следует имя модуля – правильный идентификатор.

Слово **Interface** обозначает начало раздела интерфейса модуля, доступного для всех других модулей и программ, использующих этот модуль.

После слова **Uses** указываются другие модули, которые может использовать этот модуль. **Uses** может ставиться в двух местах:

– сразу же после слова **Interface**; в этом случае константы или типы данных, объявленные в интерфейсах этих модулей, могут быть использованы в любых объявлениях;

– сразу же после слова **Implementation***:* в этом случае любые объявления этого модуля могут использоваться только внутри раздела реализации.

Раздел интерфейса

Раздел интерфейса – это «открытая» часть модуля. Она начинается ключевым словом **Interface,** следующим сразу за заголовком, и ограничена ключевым словом **Implementation.** Интерфейс определяет, что является видимым (доступным) для некоторой программы (или других модулей), использующей этот модуль. Любая программа, использующая этот модуль, имеет доступ к этим видимым элементам. В интерфейсе модуля можно объявить константы, типы данных, перемен-

ные, процедуры и функции. Они могут быть расположены в любом порядке, т. е. разделы могут встречаться повторно.

Процедуры и функции, доступные для программ, использующих этот модуль, описываются в разделе интерфейса, а их тела – операторы, реализующие их, – в разделе реализации. Объявление **Forvard** не разрешается. Тела всех доступных процедур и функций находятся в разделе реализации после раздела интерфейса, в котором перечислены их заголовки.

Раздел реализации

Раздел реализации – закрытая, недоступная часть – начинается со слова **implementation.** Все, что объявлено в части интерфейса, видимо для раздела реализации: константы, переменные, типы, процедуры и функции. Кроме того, в разделе реализации могут быть свои собственные дополнительные объявления, недоступные программам, использующим этот модуль. Программы не могут обращаться и ссылаться на них. Однако эти недоступные элементы могут использоваться видимыми процедурами и функциями, заголовки которых находятся в разделе интерфейса, а их описания – в разделе реализации.

Заголовок **procedure (function)** в разделе реализации должен быть такой же, как и в разделе интерфейса, или же иметь короткую форму. В короткой форме за ключевым словом **(procedure(function))** следует идентификатор (имя).

Раздел инициализации

Раздел реализации модуля заключен между словами **Implementation** и **end.** Но если присутствует слово **begin** перед **end** и операторы между этими словами, то получившийся составной оператор, похожий на тело главной программы, становится разделом инициализации модуля.

В данном разделе инициализируются структуры данных (переменных), используемые модулем или доступные программам, использующим этот модуль. Можно использовать этот раздел для открытия файлов.

При выполнении программы, использующей некоторый модуль, раздел инициализации вызывается перед выполнением тела главной программы. Если в программе используется несколько модулей, раздел инициализации каждого модуля вызывается (в порядке, указанном в разделе описания модулей Uses) до выполнения тела главной программы.

П р и м е р

```
UNIT MYUNIT;
  INTERFACE
IMPLEMENTATION
    USES CRT;
    BEGIN
CLRSCR;
WRITELN('ПРИВЕТ');
    END.
PROGRAM PRIM;
    USES MYUNIT;
    BEGIN
WRITELN('НАЧАЛО ПРОГРАММЫ');
……….
    END.
```

При выполнении этой программы вначале на экран выводится слово **ПРИВЕТ**, а затем – **НАЧАЛО ПРОГРАММЫ**.

Создание, компиляция и использование собственных модулей

Если написан собственный модуль, то его следует сохранить под тем же именем, которое содержится в заголовке модуля **UNIT**. Затем модуль компилируется так же, как и программа, но вместо файла с расширением **EXE** создается файл с расширени-

ем **TPU**. При компиляции модуля, если появляются сообщения об ошибках, их надо устранить. Можно оставить этот файл как одиночный или поместить его в библиотеку стандартных модулей TURBO.TPL с помощью утилиты TRUMOVER.EXE.

Чтобы созданный модуль использовать в программе, его необходимо описать в разделе **USES**.

З а д а н и е

1. Создать собственный модуль, состоящий из подпрограмм чтения файла и записи в файл.

2. Вывести на печатающее устройство листинг этой программы и содержимое файла.

14. СОЗДАНИЕ ТИПИЗИРОВАННЫХ ФАЙЛОВ, СОСТОЯЩИХ ИЗ ЗАПИСЕЙ

Типизированный файл состоит из последовательности компонентов одного типа и одной длины, что дает возможность организовать прямой доступ к каждой из них, (т. е. доступ к компоненту по его порядковому номеру).

Файловая переменная должна быть объявлена предложением **file of <тип компонента>** и связана с именем файла процедурой **Assign**. Для открытия файла используется процедура **Reset.** Для создания нового файла используется процедура **Rewrite.**

Seek(FV, Numrec) – процедура устанавливает указатель на компонент с номером **Numrec**. Первый компонент файла имеет номер 0.

Filesize(FV):longint – функция возвращает количество компонентов файла.

Чтобы переместить указатель в конец типизированного файла, можно написать **Seek(FV,Filesize(PV))**.

Filepos(FV):longint – функция возвращает порядковый номер компонента файла, который будет обрабатываться следующей операцией ввода-вывода.

Запись – структурированный тип данных, состоящий из фиксированного числа компонентов разного типа. Определение типа записи начинается идентификатором **record** и заканчивается зарезервированным словом **end.** Между ними заключен список компонентов, называемых полями, с указанием идентификаторов полей и типа каждого поля.

Формат:

Type <имя типа> = Record
<идентификатор поля, …>:<тип компонент1>;{поле1}
…
< идентификатор поля, …>:<тип компонентn>;{полеn}
end;
var <имя записи, …> : <имя типа>;

З а д а н и е

1. Написать программу с подпрограммами формирования типизированного файла, который характеризуется записями.

2. Вывести на печатающее устройство листинг этой программы и содержимое файла. Файл должен содержать не менее пяти элементов. Каждый элемент записи файла должен содержать данные (по вариантам), представленные в таблице.

Исходные данные

Вариант	Условие
1	2
1	Список студентов, проживающих в общежитии (номер комнаты, фамилия, возраст, номер группы, курс)
2	Списки студентов, которые содержат: номер группы, номер в группе по списку, Ф.И.О., год рождения, оценки за последнюю сессию, средний балл
3	Списки записавшихся на покупку мебельного гарнитура: порядковый номер, фамилия, домашний адрес покупателя, дата постановки на учет

4	Список больных по палатам: фамилия, инициалы, год рождения, пол (мужской или женский), диагноз
5	Список женихов: порядковый номер кандидата, сведения о кандидате (возраст, вес, рост), требование к партнеру (минимальное и максимальное значение соответствующего параметра)

Окончание таблицы

1	2
6	Список невест: порядковый номер кандидата, сведения о кандидате (возраст, вес, рост), требование к партнеру (минимальное и максимальное значение соответствующего параметра)
7	Список вакантных рабочих мест на предприятиях города: наименование организации, местоположение организации (в километрах от центра города), наименование должности, требуемый стаж работы по специальности, заработная плата в месяц
8	Список о сданной в ремонт радиоаппаратуре за квартал: наименование группы изделий (телевизор, видеомагнитофон, и т. д.), марка изделия, дата приемки в ремонт, состояние готовности (выполнен/не выполнен)
9	Списки об имеющихся свободных местах в железнодорожных кассах: дата выезда, номер рейса, конечный пункт назначения, время отправления, число свободных купейных мест, число свободных плацкартных мест
10	Список по заработной плате на предприятии (должность, ФИО, год рождения, стаж, заработная плата)

15. ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ ДИСКРЕТНЫХ ФУНКЦИЙ. ВЫВОД ГРАФИКОВ С ИСПОЛЬЗОВАНИЕМ МОДУЛЯ GRAPH

Для формирования графического изображения имеется библиотека **GRAPH**. Ее надо описать в разделе **USES**, т. е. **USES GRAPH**.

С момента подключения модуля **GRAPH** программисту доступны все находящиеся в ней подпрограммы. Для установления одного из возможных графических режимов используется процедура **InitGraph**.

Формат процедуры:

InitGraph (DriverVar, ModeVar, Path);,

где **DriverVar** – переменная типа **Integer**, определяющая тип графического драйвера;

ModeVar – переменная того же типа, задающая режим работы графического адаптера;

Path – выражение типа **String,** содержащее путь к драйверу **EGAVBA**. Процедура загружает драйвер в оперативную память и переводит адаптер в графический режим работы. Тип файла должен соответствовать типу графического адаптера. Имеется стандартная константа **DETECT**, которая автоматически инициирует нужный драйвер и устанавливает наиболее подходящий для дисплея режим.

Подытожив сказанное, напишем начальную группу инструкций, которая позволит избежать любых неприятностей на начальном этапе:

```
Uses Crt, Craph;
Var
    DriverVar, ModeVar; integer;
Begin
    DriverVar:= Detect;
    InitGraph (DriverVar, ModeVar; 'c:\Pascal\BGI');
```

З а д а н и е

1. Составить схему алгоритма и программу для вычисления методом Симпсона интеграла по кривой, заданной в виде дискретной функции

$$t_n = f(t_i),$$

где i = 0, 1, 2, …, n на временном интервале $t_n = n \cdot \Delta t$ – приращение аргумента (см. таблицу).

2. Вывести на печать листинг программы и результаты расчета.

Указания

– синтаксическая структура оператора ввода-вывода информации набирается самостоятельно;

– задача должна быть решена с использованием подпрограмм;

– для всех вариантов $\Delta t = 1\,\text{с}$;

– перед составлением программы для облегчения построить график $y_i = f(t_i)$;

– при выводе графика на печать необходимо использовать масштабирование функции.

Исходные данные

Вариант	Исходный массив $y_0 - y_n$
1	2
1	0; 0; 0; 0,6; 1,12; 1,56; 1,94; 2,28; 2,52; 2,74; 2,94; 3,1; 3,24;3,36; 3,44; 3,54; 3,60; 3,66; 3,72; 3.76; 3,80; 3,82; 3,84; 3,86; 3,88; 3,92
2	0; 0; 0; 0; 0,36; 0,66; 0,90; 1,1; 1,26; 1,38; 1,50; 1,58; 1,66; 1,72; 1,76; 1,80; 1,82; 1,86; 1,90; 1,92; 1,94; 1,96
3	0; 0; 0; 0,54; 0,96; 1,36; 1,64; 1,88; 2,02; 2,26; 2,38; 2,50; 2,52; 2,66; 2,72; 2,76; 2,80; 2,82; 2,90; 2,92; 2,94; 2,96

Окончание таблицы

1	2
4	0; 0; 0,3; 0,38; 0,52; 0,66; 0,76; 0,86; 0,94; 1,02; 1,04; 1,06; 1,14; 1,16; 1,22; 1,26; 1,30; 1,32; 1,34; 1,36; 1,38; 1,40; 1,42; 1,44; 1,46; 1,48; 1,50; 1,52; 1,54.
5	0; 0; 0; 0; 0,28; 0,32; 0,46; 0,56; 0,64; 0,68; 0,76; 0,80; 0,84; 0,86; 0,88; 0,90; 0,92; 0,94; 0,96; 0,98; 1,0.
6	0; 0; 0; 0,46; 0,82; 1,12; 1,36; 1,56; 1,74; 1,88; 1,98; 2,06; 2,16; 2,22; 2,28; 2,32; 2,34; 2,38; 2,40; 2,42; 2,44; 2,46
7	0; 0; 0,56; 0,972; 1,26; 1,46; 1,62; 1,72; 1,8; 1,86; 1,9; 1,92; 1,94
8	0; 0; 0; 0,56; 0,98; 1,32; 1,58; 1,78; 1,94; 2,06; 2,16; 2,24; 2,30; 2,34; 2,36; 2,40; 2,42; 2,44.
9	0; 0; 0; 1,25; 1,75; 2,25; 2,75; 3,25; 3,65; 4,05; 4,35; 4,45; 4,55; 4,65; 4,75; 4,85; 4,9.
10	0; 0; 0; 0; 0,75; 0,78; 0,81; 0,84; 0,87; 0,89; 0,91; 0,92; 0,94; 0,96; 0,98.

ЛИТЕРАТУРА

1. Информатика: базовый курс: учебное пособие для вузов / С.В. Симонович [и др.]; под ред. С.В. Симонович. – 2-е изд. – СПб.: Питер, 2009. – 639 с.

2. Фигурнов, В.Э. IBM PC для пользователя: краткий курс: сокращенная версия 7-го издания / В.Э. Фигурнов. – М.: ИНФРА-М, 2005. – 479 с.

3. Гордеев, А.В. Операционные системы: учебник для вузов по направлению «Информатика и вычислительная техника» / А.В. Гордеев. – 2-е изд. – СПб.: Питер, 2006. – 415 с.

4. Немнюгин, С.А. Turbo Pascal: программирование на языке высокого уровня: учебник для вузов по направлению «Информатика и вычислительная техника» / С.А. Немнюгин. – 2-е изд. – СПб.: Питер, 2008. – 543 с.

5. Марченко, А.И. Программирование в среде Turbo Pascal 7.0 / А.И. Марченко, Л.А. Марченко. – Киев: ВЕК+, 2000. – 464 с.

6. Шпак, Ю.А. Turbo Pascal 7.0 на примерах / Ю.А. Шпак. – Киев: Юниор, 2003. – 496 с.

7. Фаронов, В.В. Турбо Паскаль 7.0. Начальный курс: учебное пособие / В.В. Фаронов. – 7-е изд., перераб. – М.: Нолидж, 2000. – 576 с.

8. Меженный, О.А. Turbo Pascal: самоучитель / О.А. Меженный. – М.: СПб.; Киев: Диалектика. 2004. – 335 с.

ОГЛАВЛЕНИЕ

Учебное издание

МОСКАЛЕНКО Алексей Анисимович
КОНОНЕНКО Зоя Ивановна

РЕШЕНИЕ ПРИКЛАДНЫХ ЗАДАЧ В ИНТЕГРИРОВАННОЙ СРЕДЕ ТУРБО ПАСКАЛЬ

Методическое пособие
по дисциплинам «Информатика»,
«Математические модели информационных процессов
и управления», «Основы алгоритмизации и программирование»
для студентов специальностей: 1-53 01 01 «Автоматизация
технологических процессов и производств»,
1-53 01 02 «Автоматизированные системы обработки
информации» и 1-53 01 06 «Промышленные роботы
и робототехнические комплексы»

Редактор Т.Н. Микулик
Компьютерная верстка Н.А. Школьниковой

Подписано в печать 19.09.2011.
Формат $60\times84^{1}/_{16}$. Бумага офсетная.
Отпечатано на ризографе. Гарнитура Таймс.
Усл. печ. л. 3,60. Уч.-изд. л. 2,82. Тираж 100. Заказ 89.

Издатель и полиграфическое исполнение:
Белорусский национальный технический университет.
ЛИ № 02330/0494349 от 16.03.2009.
Проспект Независимости, 65. 220013, Минск.