Министерство образования Республики Беларусь
БЕЛОРУССКАЯ ГОСУДАРСТВЕННАЯ ПОЛИТЕХНИЧЕСКАЯ АКАДЕМИЯ

Кафедра “Двигатели внутреннего сгорания”

ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ПРОГРАММИРОВАНИЕ

ЛАБОРАТОРНЫЙ ПРАКТИКУМ
для студентов специальностей Т.05.10.00 - “Двигатели внутреннего сгорания” и Т.04.07.00 - “Техническая эксплуатация летательных аппаратов и двигателей”

В 2-х частях

Часть I

ПРОГРАММИРОВАНИЕ В СРЕДЕ TURBO PASCAL 7.0

Минск 1998

УДК 681.3.06

Учебное пособие соответствует программе дисциплины “Вычислительная техника и программирование” для студентов дневного и заочного обучения по специальностям Т.05.10.00 - “Двигатели внутреннего сгорания” и Т.04.07.00 - “Техническая эксплуатация летательных аппаратов и двигателей” и включает в себя краткое описание программ Norton Commander 3.0 и Turbo Pascal 7.0, задания к лабораторным работам и примеры их выполнения, а также в приложении сообщения об ошибках и “горячие” клавиши.

Составители:

В.А.БАРМИН
А.П.КИСЕЛЕВА
В.В.ТРИКОЗЕНКО

Рецензенты:

Л.А.МОЛИБОШКО
А.С.ПОВАРЕХО

В в е д е н и е

Лабораторный практикум составлен в соответствии с программой дисциплины “Вычислительная техника и программирование”, преподаваемой студентам дневного и заочного обучения по специальностям Т.05.10.00 - “Двигатели внутреннего сгорания” и Т.04.07.00 - “Техническая эксплуатация летательных аппаратов и двигателей”.

Материал, приведенный в начале лабораторного практикума, содержит минимум теоретических и максимум практических сведений по основным разделам языка Pascal. Сведения эти представлены в виде примеров описания скалярных и структурированных типов данных и операторов языка, включенных в структуру программы Pascal, а также справочных сведений по операциям, стандартным библиотечным модулям, встроенным процедурам и функциям.

В практикум включены 10 лабораторных работ. Первые две лабораторные работы ознакомительные содержат краткое описание сервисной оболочки операционной системы MS DOS Norton Commander (версии 3.0) и интегрированной среды программирования Turbo Pascal (версии 7.0), а также задания к работе по освоению указанных пакетов программ. Последующие лабораторные работы охватывают программирование алгоритмов линейной, разветвляющейся и циклической структур, а также программирование задач с использованием подпрограмм пользователя, построения графиков на экране дисплея и обработки файловых структур данных. В лабораторных работах даются задания для самостоятельной подготовки и к выполнению работ. Кроме того, в них приводятся примеры выполнения этих работ, включающие схему алгоритма и программу решения задачи. В приложение вынесены сообщения об ошибках при компиляции и выполнении программы, списки «горячих» клавиш для программ Norton Commander и Turbo Pascal, пример оформления отчета по лабораторной работе.

Основное назначение лабораторного практикума - дать практические и наглядные рекомендации по разработке алгоритма решения задачи и составлению по нему программы, используя приведенные примеры и необходимую информацию для выполнения лабораторной работы. По мнению авторов, такое содержание практикума ускорит освоение алгоритмического языка и приобретение навыков программирования в среде Turbo Pascal.

Практикум может быть использован студентами и других специальностей, изучающих программирование в среде Turbo Pascal.

1. СТРУКТУРА ПРОГРАММЫ PASCAL

Структура программы с примерами описания данных и действий над ними выглядит следующим образом:

```
Program Example; {Заголовок программы. “Example” - имя программы}
{Структура программы PASCAL. Автор - Иванов И.И.} (*Комментарий о назначении программы и кто ее автор*)
Uses CRT, SISTEM, GRAPH, MyLib; {Список подключаемых стандартных и пользовательских библиотечных модулей}
Label Metka1, Metka2, 111; {Описание меток для прямого перехода с помощью оператора goto к оператору программы с указанной меткой}
Const {Раздел описания констант}
    Maxind:integer=100; {Типизированная константа - это инициализированная переменная с заданным начальным значением}
    Vxod='Блок_1'; {Строковая константа}
    MinReal=1.7E-3; {Вещественная константа}
Type {Раздел описания используемых типов данных}
    Matr=array[1..5,1..5] of real; {Описание типа матрица 5×5}
    Days=(Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday); {Описание перечисляемого типа пользователя}
    Dni=1..31; {Описание интервального типа пользователя}
    Line=string[80]; }Описание типа “строка”}
    Prostoe=set of  ‘A’..’Z’; {Описание типа “множество”}
    Mash=record {Описание типа “запись”}
        Nomer:integer; {Номер}
        Marka:string[20]; {Марка автомобиля}
        Fio:string[40]; {Фамилия и инициалы владельца}
        Adres:string[60] {Адрес владельца}
    end;
Var {Раздел описания переменных}
    Matr1,Matr2:Matr;
    WorkDay, RedDay:Days;
    RabDni,BolnDni:Dni;
    St1,St2:Line;
    Pro:Prostoe;
    Ft:file of Mash;
    M:Mash;
    Fio1:string[40];
    z1,y1,a,b,c:real;
    x1,x2,n,k:integer;
```

```
      ch:char;
      Flag1:boolean;
Procedure Sort (Var X1,X2: integer; R1,R2: integer); {Процедура передает в основную программу параметры-переменные X1, X2; параметры-значения R1, R2 используются в процедуре и в основную программу не передаются}
      begin
        X1:=R1+R1;
        X2:=R1*R2;
      end;
Function Prov (X,Y,Z:real):real; {Функция использует локальные параметры X, Y, Z, в основную программу передается имя, которому присваивается результат вычислений}
  Var X,Y,Z: real;
      begin
        Prov:=X+Y*Z
      end;
Begin   {Начало головной программы}
      Writeln('Ввести исходные данные: ');
      Readln(A,B,C,N,K); {Ввод исходных данных}
      Writeln(A:4:2, B:5:2, C:9:3, N:3, K:4); {Вывод исходных данных}
      Y1:=A+B-C; {Оператор присваивания}
Sort(X1,X2,N,K); {Вызов процедуры в основную программу}
Goto Metka1; {Оператор безусловного перехода}
      Z1:=Prov(A,B,C); {Присваивание переменной Z1 значения функции Prov}
Metka1: if X1<X2 then {Условный оператор}
                  N:=X1
            else
                  K:=X2;
Case WorkDay of {Оператор выбора}
      Monday: Y1:=A*B*C;
      Tuesday: Sort(x1,x2,n,k)
else  Writeln('Ошибка!')
end;
for N:=1 to 5 do {Оператор повтора}
      for k:=1 to 5 do Matr1[N,K]:=0; {Инициализация элементов матрицы}
      N:=1; K:=0;
repeat {Оператор повтора с постусловием}
      K:=K+N;
      N:=N+1; {Изменение переменной, влияющей на условие}
```

```
until (N>100); {Цикл завершается при выполнении условия}
While (N>100) do {Оператор повтора с предусловием}
    begin
        N:=N+1;
        if N>200 then goto Metka2 {Условие выхода из цикла}
    end;
Metka2: Writeln('Переменная N=',N:3);
Assign(Ft,'D:\File1.dat'); {Команда связывает имя файловой переменной Ft
с именем внешнего файла на диске D:\File1.dat}
Rewrite(Ft); {Открыть файл Ft для записи данных}
    write('Введите номер автомобиля '); {Вывод на экран пользова-теля
запроса о дальнейших действиях}
    readln(M.Nomer); {Ввод номера автомобиля}
    write('Введите марку автомобиля ');
    readln(M.Marka); {Ввод марки автомобиля}
    write('Введите фамилию и инициалы владельца ');
    readln(M.Fio); {Ввод фамилии владельца}
    write('Введите адрес владельца ');
    readln(M.Adres); {Ввод адреса владельца}
    write(Ft,M); {запись переменной M типа Mash в файл Ft}
Close(Ft); {Закрыть фал Ft}
Reset(Ft); {Открыть файл Ft для чтения; указатель записи установлен на
первой записи файла, EOF(Ft)=False}
while (EOF(Ft)<>True) do {Цикл выполняется до тех пор, пока не обнару-
жен конец файла; EOF(Ft)=True}
begin
    read(Ft,M); {Считать в переменную M один элемент файла Ft}
    writeln(M.Nomer:3,M.Marka:20,M.Fio:40,M.Adres:60); {Вывод пере-
менной M}
end;
Close(Ft); {Закрыть фал Ft}
Reset(Ft); {Открыть файл Ft для корректировки данных; указатель записи
установлен на первой записи файла, EOF(Ft)=False}
    Seek(Ft,N-1); {Подвести указатель файла к корректируемой записи с
порядковым номером N}
    Read(Ft,M); {Считать корректируемую запись}
    Writeln(M.Nomer:3,M.Marka:20,M.Fio:40,M.Adres:60); {Вывод пере-
менной M}
    Writeln('Измените фамилию владельца автомобиля', );
    Read(Fio1); {Ввод новой фамилии владельца}
    M.Fio:=Fio1; {Присвоение переменной нового значения}
```

Seek(Ft,N-1); {Повторение процедуры подвода указателя файла к нужной записи}

write(Ft,M); {запись переменной M типа Mash в файл Ft}

Close(Ft) {Закрыть фал Ft}

End.

2. ТИПЫ ДАННЫХ

Скалярные.

Целочисленные типы данных обозначают множества целых чисел в различных диапазонах (табл.2.1).

Таблица 2.1

Тип	Диапазон	Требуемая память (бит)
byte	0..255	8
word	0..65535	16
integer	-32768..32767	16
shorting	-128..127	8
longint	-2147483648.. 2147483648	32

Вещественные типы данных обозначают множества вещественных значений в различных диапазонах (табл.2.2).

Таблица 2.2

Тип	Диапазон	Мантисса	Требуемая память (байт)
real	2.9E-39..1.7E38	11-12	6
single	1.5E-45..3.4E38	7-8	4
double	5.0E-324..1.7E308	15-16	8
extended	3.4E-4932..1.1E4932	19-20	10
comp	-9.2E18..9.2E18	19-20	8

Символьные типы данных представляют собой символы из множества кодовой таблицы ASCII (American Standard Code For Information Interchange. Тип - char.).

Булевский тип данных представлен двумя значениями: True (истина) и False (ложь). Тип - boolean.

Перечисляемый тип данных задается непосредственно перечислением всех значений, которые может принимать переменная данного типа. Отдельные значения указываются через запятую, а весь список заключается в круглые скобки.

Интервальный тип данных позволяет задавать две константы, определяющие границы диапазона значений для данной переменной. Обе константы должны принадлежать одному из стандартных типов (тип real здесь недопустим). Значение первой константы должно быть обязательно меньше второй.

Структурированные.

Пример записи структурированных типов данных: строки (string), массивы (array), множества (set), записи (record), файлы (file) приведены в структуре программы PASCAL ранее.

3. ОПЕРАЦИИ

Арифметические операции выполняют арифметические действия в выражениях над значениями операндов целочисленных и вещественных типов (табл.3.1).

Таблица 3.1

Операция	Действие	Типы операндов	Тип результата
Бинарные			
+	Сложение	Целый, веществ.	Целый, веществ.
-	Вычитание	Целый, веществ.	Целый, веществ.
*	Умножение	Целый, веществ.	Целый, веществ.
/	Деление	Целый, веществ.	Вещес., вещес.
div	Целочисленное деление	Целый	Целый
mod	Остаток	Целый	Целый
and	Арифметическое И	Целый	Целый
shl	Сдвиг влево	Целый	Целый
shr	Сдвиг вправо	Целый	Целый
or	Арифметическое ИЛИ	Целый	Целый
hor	Исключающая дизъюнкция	Целый	Целый
Унарные			
+	Сохранение знака	Целый, веществ.	Целый, веществ.
-	Отрицание знака	Целый, веществ.	Целый, веществ.
not	Арифметическое отрицание	Целый	Целый

Операции отношения выполняют сравнение двух операндов и определяют значение выражения “истинно” или “ложно” (табл.3.2).

Таблица 3.2

Операция	Действие	Выражение	Результат
=	равно	A=B	True, если A равно B
<>	не равно	A<>B	True, если A не равно B
>	больше	A>B	True, если A больше B
<	меньше	A<B	True, если A меньше B
>=	больше и равно	A>=B	True, если A больше или равно B
<=	меньше или равно	A<=B	True, если A меньше или равно B
in	принадлежность	A in B	True, если A находится в списке B

Логические операции выполняются над выражениями, результатом операций являются значения True или False (табл.3.3).

Таблица 3.3

Операция	Действие	Выражение	A	B	Результат
not	Логическое отрицание	not A	True		False
			False		True
and	Логическое И	A and B	True	True	True
			True	False	False
			False	True	False
			False	False	False
or	Логическое ИЛИ	A or B	True	True	True
			True	False	True
			False	True	True
			False	False	False

Приоритет операций указан в таблице 3.4.

Таблица 3.4

Операция	Приоритет	Вид операции
@, not	Первый (высший)	Унарная операция
*, /, div, mod, and, shl, shr	Второй	Операции типа умножения
+, -, or		
=, <>, <, >, <=, >=, in	Третий	Операции типа сложения
	Четвертый (низший)	Операции отношения

4. СТАНДАРТНЫЕ БИБЛИОТЕЧНЫЕ МОДУЛИ

CRT - содержит средства управления дисплеем и клавиатурой ПЭВМ, включая управление текстовыми и графическими режимами работы экрана, установку окон, цвета и фона для выводимых символов, обработку возвращаемых кодов клавиатуры, а также звуковые эффекты.

DOS - включает средства, позволяющие реализовать различные функции DOS. Поддерживает обслуживание прерываний, проверку состояния диска, специальных средств обработки файлов, управление процессами и операционной средой, таймером и звуком, старт .EXE-файлов.

OVERLAY - содержит средства организации оверлейных программ.

PRINTER - обеспечивает быстрый доступ к печатающему устройству. Для вывода на печать достаточно указать LST в процедуре ввода (например, write(LST, 'Вывод на печать');).

SYSTEM - является сердцем Turbo Pascal; содержащиеся в нем подпрограммы обеспечивают работу всех остальных модулей системы.

GRAPH - содержит пакет графических средств, обеспечи-вающих эффективную работу с адаптерами EGA, VGA и др.

5. ВСТРОЕННЫЕ ПРОЦЕДУРЫ И ФУНКЦИИ СТАНДАРТНЫХ МОДУЛЕЙ SYSTEM, CRT и GRAPH

Арифметические функции и процедуры реализуют наиболее часто встречающиеся в практике математические действия и операции. Ниже в их описании используются обозначения: X - целочисленные и вещественные типы, I - только целочисленные.

Abs(X) - вычисление абсолютной величины X. Тип результата совпадает с типом параметра.

Пр.: write(Abs(4-6):2); результат = 2.

ArcTan(X) - вычисление арктангенса аргумента X. Результат имеет вещественный тип.

Пр.: write(ArcTan(1)*180/Pi); результат = 4.5000000000E+1.

Cos(X) - вычисление косинуса X. Результат имеет вещественный тип.

Пр.: write(Cos(60*Pi/180)); результат = 5.0000000000E-01.

Exp(X) - вычисление экспоненты X, т.е. значение e в степени X. e является основанием натурального логарифма и равняется 2.718282. Результат имеет вещественный тип.

Пр.: write(Exp(1.5)*2); результат = 2.0085536923E+01.

Frac(X) - вычисление дробной части X. Результат имеет вещественный тип.

Пр.: write(Frac(0.25*11):4:2); результат = 0.75.

Int(X) - вычисление целой части X. Результат имеет вещественный тип.

Пр.: write(Int(422.117):4); результат = 422.

Ln(X) - вычисление натурального логарифма X. Результат имеет вещественный тип.

Пр.: write(Ln(0.12*10)); результат = 1.8232155679E-01.

Pi - Возвращает значение числа π (3,141592653897932385)

Пр.: write('Значение Пи = ',Pi);

Sin(X) - вычисление синуса X. Результат имеет вещественный тип.

Пр.: write(Sin(60*Pi/180)); результат = 8.6602540378E-01.

Sqr(X) - возведение в квадрат значения X. Тип результата совпадает с типом параметра.

Пр.: write(Sqr(-5):2); результат = 25.

Sqrt(X) - вычисление квадратного корня из X. Тип результата вещественный.

Пр.: write(Sqrt(25):1); результат = 5.

Random - генерирует значение случайного числа из диапазона 0..0.99. Тип результата вещественный.

Пр.: for I:=1 to 3 do write(Random:8:5);

результат = 0.06919 0.78539 0.17197.

Random(I) - генерирует значение случайного числа из диапазона 0..I. Тип результата целочисленный.

Пр.: for I:=1 to 3 do write(Random(10):2); результат = 7 2 9.

Примечание. В тригонометрических функциях аргумент должен быть задан толко в радианах. Если X аргумент задан в градусах, то для перевода его в радианы существует формула: У=X∗Pi/180.

В Turbo Pascal нет операции возведения в степень. При ее использовании можно применять стандартные функции:

$$A^B = Exp(B*Ln(A)).$$

При возведении в целую степень отрицательного числа используют операторы цикла.

Пр.: Возвести число -4 в степень 5.

x:=1; for n=1 to 5 do x:=x∗(-4);

Для вычисления остальных тригонометрических функций необходимо использовать известные соотношения:

Tg(x) = Sin(x)/Cos(x);

Ctg(x) = Cos(x)/Sin(x);

Csc(x) = 1/Sin(x);

$$\mathbf{Sc}(x) = 1/\mathrm{Cos}(x);$$

$$\mathbf{ArcSin}(x) = \mathrm{ArcTg}(x/(1-x^2))^{1/2};$$

$$\mathbf{ArcCos}(x) = \mathrm{Pi}/2 - \mathrm{ArcSin}(x)$$

$$\mathbf{ArcCtg}(x) = \mathrm{Pi}/2 - \mathrm{ArcTg}(x)$$

Вычисление логарифма с основанием a выполняется следующим образом:

$$\mathbf{Log}_a(x) = \mathrm{Ln}(x)/\mathrm{Ln}(a).$$

Скалярные процедуры и функции обрабатывают данные любого скалярного типа, кроме вещественного.

Процедуры.

Dec(X,n) - уменьшает значение целочисленной переменной X на n. При отсутствии необязательного параметра n значение X уменьшается на 1.

Пр.: X:=10; Dec(X,2); результат = 8.

Inc(X,n) - увеличивает значение целочисленной переменной X на n. При отсутствии необязательного параметра n значение X увеличивает на 1.

Пр.: X:=10; Inc(X,3); результат = 13.

Функции.

Pred(S) - возвращает элемент, предшествующий S в списке значений типа. Тип результата совпадает с типом параметра. Если предшествующего значения S элемента не существует, возникает программное прерывание.

Пр.: write(Pred(90)); результат = 89.

Succ(S) - возвращает значение, следующее за S в списке значений типа. Тип результата совпадает с типом параметра. Если следующее за S значение отсутствует, возникает программное прерывание.

Пр.: write(Succ(90)); результат = 91.

Odd(I) - возвращает значение булевского типа, равное True, если I нечетное, и False, если I четное.

Пр.: write(Odd(3)); результат = True.

Функции преобразования типов используются для преобразования значений одного скалярного типа в значение другого скалярного типа.

Chr(I) - возвращает символ стандартного кода обмена информацией с заданным в I порядковым номером в коде ASCII. Результат имеет литерный тип. Если значение параметра больше 255, возникает программное прерывание.

Пр.: write(Chr(105)); результат = ‘i’.

Ord(S) - возвращает порядковый номер значения S в множестве, определенном типом S. Результат целочисленного типа. Диапазон значений функции изменяется от 0 до 32767.

Пр.: write(Ord(‘A’)); результат = 65.

Round(X) - возвращает значение X, округленное до ближайшего целого числа. Результат имеет целочисленный тип.

Пр.: write(Round(5.6):2); результат = 6.

Trunc(X) - возвращает ближайшее целое число, меньшее или равное X, если X>=0, и большее или равное X, если X<0. Результат имеет целочисленный тип.

Пр.: write(Trunc(5.7):2); результат = 5.

Основные графические функции и процедуры модуля Graph используются для поддержания работы графического режима и отображения на экране дисплея графической информации.

InitGraph(Gd,Gm;Path) - процедура инициализирует работу графического режима, т.е. осуществляет загрузку драйвера, поддерживающего работу графического адаптера (модуль драйвера хранится в специальном файле с расширением BGI). Gd и Gm являются параметрами-переменными типа integer. Gd задает номер графического драйвера, Gm задает номер графического режима, допустимого для заданного драйвера. Path переменная типа string задает путь к каталогу, в котором находится графический драйвер (BGI-файл).

GetGraphMode:integer - возвращает код текущего графического режима.

SetGraphMode(Mode:integer) - устанавливает новый графический режим.

GetX:integer - возвращает текущую X-координату.

GetY:integer - возвращает текущую Y-координату.

GetMaxX:integer - возвращает максимальную координату X.

GetMaxY:integer - возвращает максимальную координату Y.

PutPixel(X,Y:integer; Pixel:word) - процедура выдает на экран точку с координатами X, Y и цветом Pixel.

SetViewPort(X1,Y1,X2,Y2:integer; Clip:boolean) - процедура, устанавливает окно с координатами (X1,Y1) - левый верхний угол и (X2,Y2) - правый нижний угол. Если значение Clip истинно, то изобра-жение, не вмещающееся в окно, отсекается, в противном случае не отсекается. После установки окна любой вывод на экран будет производится в относительных координатах текущего окна.

ClearDevice - процедура очищает экран и устанавливает те значения всех графических параметров, которые предусмотрены по умолчанию.

ClearViewPort - процедура очищает текущее окно.

LineTo(X,Y:integer) - процедура проводит прямую линию из точки, где находится текущий указатель, в точку с координатами (X,Y) текущим цветом. Текущий указатель перемещается в точку (X,Y).

LineRel(Dx,Dy:integer) - процедура проводит прямую линию из точки, где находится текущий указатель, в точку с приращением координат на Dx (по X) и на Dy (по Y) текущим цветом. Текущий указатель перемещается в конец линии.

Line(X1,Y1,X2,Y2:integer) - процедура проводит прямую линию из точки с координатами (X1,Y1) в точку (X2,Y2) текущим цветом. Положение текущего указателя не изменяется.

MoveTo(X,Y:integer) - процедура перемещает текущий указатель в точку (X,Y).

SetColor(Color:word) - процедура устанавливает цвет выводимого изображения, задаваемы параметром Color.

SetBkColor(Color:word) - процедура устанавливает цвет фона.

Rectangle(X1,Y1,X2,Y2:integer) - процедура рисует прямоугольник с координатами (X1,Y1) - верхний левый угол и (X2,Y2) - нижний правый угол.

Bar(X1,Y1,X2,Y2:integer) - процедура рисует закрашенный прямоугольник с координатами (X1,Y1) - верхний левый угол и (X2,Y2) - нижний правый угол, используя стандартный цвет закраски.

Circle(X,Y:integer; Radius:word) - процедура рисует окружность с центром (X,Y) и радиусом Radius.

Arc(X,Y:integer; St,End,Radius:word) - процедура рисует дугу окружности от угла St до End с центром в точке (X,Y) и радиусом Radius.

Ellipse(X,Y:integer; St,End,XRadius,Yradius:word) - процедура рисует дугу эллипса с центром (X,Y) и с радиусами XRadius (по оси X), Yradius (по оси Y) от начального угла St до конечного угла End.

OutText(Text:string) - процедура выводит на экран строку Text, начиная с текущей позиции указателя. Текущий указатель перемещается в конец строки.

OutTextXY(X,Y:integer; Text:string) - процедура выводит на экран строку Text, начиная с точки (X,Y). Положение текущего указателя не изменяется.

CloseGraph - процедура восстанавливает режим, существовав-ший до инициализации графики.

Лабораторная работа №1
“Сервисная оболочка операционной системы MS DOS Norton Commander”

Цель работы - знакомство с программой Norton Commander и режимами ее работы.

Краткое описание программы Norton Commander

Запуск Norton Commander осуществляется набором в командной строке NC. Часто запуск программы Norton Commander производится при включении ЭВМ и загрузки в оперативную память операционной системы MS DOS.

Для выхода из Norton Commander надо нажать клавишу [F10]. В центре экрана появится запрос на подтверждение того, что вы хотите выйти из Norton Commander. Чтобы выйти, нажмите [ENTER] или "Y". Чтобы отменить выход, нажмите [ESC] или "N".

После запуска Norton Commander в верхней части экрана появляются два прямоугольных окна, ограниченные двойной рамкой (далее эти окна будут называться панелями). Ниже этих панелей располагается командная строка с обычным приглашением DOS. Там можно вводить обычные команды DOS. Если выполняемая процедура выдала сообщения, которые потом закрыли панели Norton Commander, его можно посмотреть, выключив панели с помощью 'Ctrl-O' и затем также включить.

Еще ниже располагается строка, напоминающая значения функциональных клавиш Norton Commander. Вид экрана при работе с Norton Commander показан на рисунке.

Общий вид панелей Norton Commander

В каждой панели Norton Commander может содержаться:

- оглавление каталога на диске;
- дерево каталогов на диске;
- сводная информация о диске и каталоге на другой панели.

Если в панели содержится оглавление каталога, то наверху панели выводится имя этого каталога. Если в панели содержится дерево каталогов на диске, сверху выводится "TREE". Если в панели содержится информация о диске и каталоге, сверху панели выводится "INFO".

Имена файлов в оглавлении каталога выводятся строчными буквами, а подкаталоги - прописными буквами. Справа от имени подкаталога изобра

жается <SUB-DIR>.

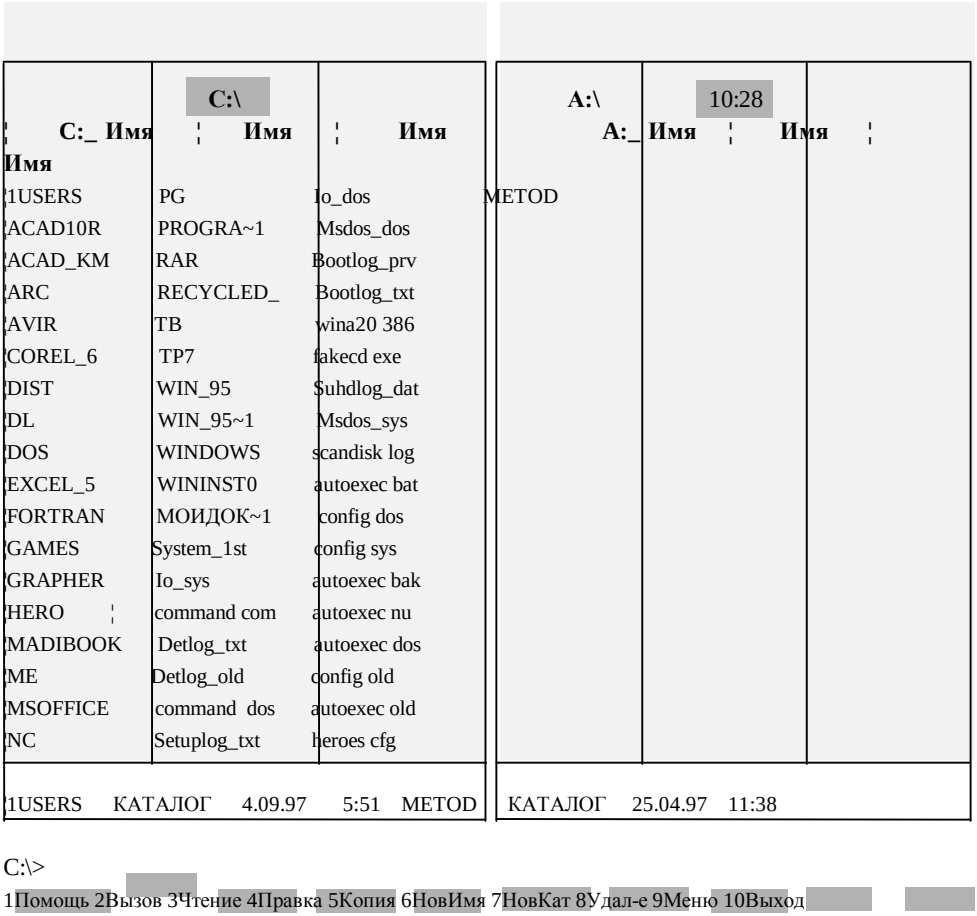


Рис.1.

Самую верхнюю строку в оглавлении занимает ссылка на родительский каталог (разумеется, для корневого каталога диска эта строка отсутствует). В поле имени для родительского каталога изображается "..", справа изображается <UP-DIR>.

Один из файлов или каталогов на экране выделен серым цветом (На монохромном дисплее - инверсным изображением). Будем называть такой файл или каталог выделенным.

Клавишами перемещения курсора [↑], [↓], [PgUh], [PgDn] можно перемещать выделенный участок на экране, выделяя другой файл или каталог.

Клавишей [Tab] можно перевести выделенный участок на другую панель Norton Commander.

Если выделить какой-нибудь подкаталог и нажать [Enter], то Norton Commander "войдет" в этот подкаталог и выведет его оглавление.

Чтобы перейти в родительский каталог, надо выделить его и нажать <ENTER>.

Функциональные клавиши

В самой нижней строке экрана перечислены команды, выполняемые при нажатии функциональных клавиш. Например, для копирования выбранного файла нужно нажать клавишу <F5>. Теперь рассмотрим все команды подробно:

F1 - <Help> (помощь)

При нажатии этой клавиши на экране появляется подсказка, описывающая назначение клавиш и команд.

F2 - <User> (Меню пользователя)

С помощью этой клавиши пользователь может вызвать созданное им самим дополнительное меню команд, выполняемых при нажатии любых клавиш. Это меню может содержать команды, наиболее часто используемые вами при работе и избавит вас от их постоянного набора.

F3 - <View> (Просмотр)

Эта команда позволяет просматривать на экране файлы, не изменяя их. Преимущество просмотра файлов с помощью этой команды перед текстовыми редакторами в том, что она позволяет просматривать файлы любой длины, делать это быстро, исключает возможность случайной порчи файла при просмотре.

Текст можно перемещать вверх, вниз, влево и вправо. В самой верхней строке экрана расположена строка, которая содержит следующую информацию (слева направо):

- имя просматриваемого файла (<View:...>);
- номер крайней левой позиции на экране (<Col...>);
- объем файла в байтах (<...Bytes>);
- часть просмотренного файла (...%).

В самой нижней строке экрана содержится строка, которая поясняет назначение функциональных клавиш: - F7 - <Search> (найти), позволяет найти в просматриваемом тексте данную строку символов, искомая строка набирается непосредственно после нажатия <F7>;

- F10- <Quit> (выход), команда окончания просмотра. Нажатием <Shift-F3> можно посмотреть любой файл, имя которого вы запросите.

F4 - <Edit> (Редактирование).

По этой команде вызывается встроенный в Norton Commander редактор текста, который обладает ограниченными возможностями редактирования, но вызывается очень быстро.

Выберите файл и нажмите <F4>, вы увидите содержимое этого файла , которое можно редактировать. Нажав клавишу <F1> вы можете получить подсказку по работе в редакторе. В самой верхней строке экрана расположена следующая информация (слева направо):

- имя редактируемого файла (<Edit:...>);
- '*' - звездочка, если файл изменялся после вызова редактора;
- номер строки, в которой находится курсор (<Line...>);
- номер позиции, в которой находится курсор (<Col...>);
- объем свободной памяти для редактирования файла (<...free>);
- десятичный код (или обозначение) символа, находящегося над курсором.

В самой нижней строке экрана расположена информация, которая поясняет назначение функциональных клавиш в данном редакторе:

- F1 - <Help> (помощь),выводит на экран подсказку, описывающую назначение клавиш и команд.
- F2 - <Save> (сохранить), сохраняет содержимое редакти-руемого файла на диске;
- F7 - <Search> (найти),позволяет найти в редактируемом тексте заданную строку символов. Искомая строка набирается непосредственно после нажатия;
- F10 - <Quit> (выход), команда выхода из редактора. Если после выхода из редактора вы не изменяли файл, то по команде <Quit> вы выйдете из редактора. Если файл изменялся, появится красная таблица со словами <Save>, <Don't save>, <Continue editing>.

Выбрав соответствующее слово, вы можете выйти из редактора с сохранением файла, без сохранения и вернуться обратно в редактор. Нажатием <Shift-F4> можно редактировать любой файл.

F5 - <Copy> (Копирование)

Эта команда позволяет копировать файлы или группы выбранных файлов. Чтобы скопировать файл, выберите его или группу файлов, нажмите <F5>, наберите имя накопителя и каталога и нажмите <Enter>. Нажатием можно скопировать любой файл и каталог, имя которого вы наберете, также в <'to'> любой каталог.

F6 - <RenMov> (Переименование и перемещение файлов)

Этой командой вы можете переименовать выбранный файл или переместить его(или группу файлов). Чтобы переименовать файл, выделите его курсором, нажмите <F6>, наберите новое имя файла(не указывая имя накопителя и каталога) и нажмите <Enter>. Чтобы переместить файл, выделите его или группу файлов, нажмите <F6>, наберите имя накопителя и каталога и нажмите <Enter>. Нажатием <Shift-F6> можно переместить или переименовать любой файл или каталог, имя которого вы наберете, также в <'to'> любой каталог или имя.

F7 - <MkDir> (Создать новый каталог)

Эта команда создает новый подкаталог того каталога, в котором вы находитесь в данный момент. Нажатием <Shift-F7> можно создать любой каталог, имя которого вы наберете.

F8 - <Delete> (Удалить)

С помощью этой команды можно удалить файл, группу файлов или пустой каталог. Перед удалением выдается запрос на подтверждение удаления (красная рамка со словами 'Ok' и 'Cancel'), если вы уверены, нажмите <Enter>, нет - <Esc>. Нажатием <Shift-F8> можно удалить любой файл и каталог, имя которого вы наберете.

F9 - <PullDn> (Меню Norton Commander)

По этой команде вызывается меню Norton Commander в самой верхней строке экрана, с помощью которого можно получать информацию, изменять режимы работы и выполнять другие действия.

F10 - <Quit> (Выход)

Эта команда позволяет выйти из Norton Commander.

Меню Norton Commander

Если вы нажмете клавишу <F9>, в верхней строке появится строка меню со следующими словами (слева направо):

- <Left>, определяет вид и содержание левой панели;
- <Files>, выполнение операций над файлами;
- <Commands>, дополнительные команды;
- <Options>, определяет параметры;

- <Right>, определяет вид и содержание правой панели. Если вы клавишами перемещения курсора влево и вправо выберете одно из этих слов и нажмете <Enter>, то из выбранной позиции <выпадет> еще одно меню, в котором вы уже можете выбрать непосредственно исполняемую команду, перемещая курсор клавишами клавишами перемещения курсора вверх и вниз и нажав потом клавишу <Enter>.

Теперь более подробно о каждой позиции меню и командах, содержащихся в них:

<Left>(<Right>) - левая (правая)

С помощью данного меню можно выбрать следующие варианты внешнего вида каждой из панелей и его содержимого (выбранные выриантs помечаются галочками):

1) <Brief> - высвечиваются имена файлов и их расширения;

2) <Full> - высвечиваются имена файлов с расширениями, их размеры, даты и времена создания;

3) <Info> - высвечивается информация о текущем накопителе и каталоге;

4) <Tree> - высвечивается иерархическое дерево каталогов;

5) <On/Off> (Ctrl-F2) - позволяет включать и выключать панель на экране;

Следующие пять команд позволяют при выводе на экран сортировать файлы по:

6) - имени (<Name>);

7) - расширению (<extension>);

8) - дате и времени создания или последнего изменения (<Time>);

9) - размеру (<Size>);

10) (<Unsorted>) отмена сортировки, то есть файлы будут высвечиваться в порядке, в котором они расположены на накопителе;

11) <Re-read> - повторное считывание содержимого накопите-ля или каталога (например, при замене дискеты);

12) <Drive..>(Alt-F2) - смена накопителя. При выборе этой команды высвечивается список накопителей, из которого выберите нужный и нажмите <Enter>.

<Files>

Команды этого меню полностью повторяют команды функциональных клавиш <F1>-<F10>.

<Commands> (Команды):

1) <NCD tree> - позволяет сменить текущий каталог на любой другой с помощью иерархического дерева каталогов. После выбора данной команды на экране появится <дерево>, в котором выберите каталог и нажмите <Enter>. Для ускоренного поиска каталога наберите на клавиатуре его имя.

2) <Find file> - поиск файла с заданным именем по всем каталогам данного накопителя. Выберите эту функцию, наберите имя файла или модель поиска (например, *.exe- все файлы с расширением <.exe>) и нажмите <Enter>. Найденные файлы будут высвечиваться на экране. После окончания поиска вы сможете выбрать нужный файл стрелками перемещения курсора вверх и вниз нажав затем <Enter>. После этого Вы выйдете в тот каталог, где находится файл.

3) <History> - показывает несколько предыдущих команд DOS, набранных Вами. Из них Вы можете выбрать необходимую (если есть) и выполнить ее вновь, нажав <ENTER>.

4) <Ega line> - если Вы имеете на компьютере контроллер дисплея типа EGA, то эта команда позволит Вам высвечивать на экране 43 строки информации вместо обычных 25.

5) <Swap panel> - меняет местами панели экрана.

6) <Panel on/off> - включает и выключает панели на экране (если надо посмотреть что "под ними").

7) <Compare directories> - сравнить каталоги. Сделайте так, чтобы содержимое каталогов, которые Вы хотите сравнить, располагались в панелях экрана. Затем выберите данную команду. Теперь на экране в каждом из каталогов будут выделены те файлы, которые отсутствуют в другом.

8) <Menu file edit> - позволяет редактировать содержимое файлов, определяющих функции пользователя и настройку Norton Commander по вашему желанию. После выбора данной команды на экране появится таблица со словами, определяющими следующие варианты:

а) редактирование файла, содержимое которого определяет набор команд пользователя вызываемых при нажатии <F2> данный файл имеет имя <NC.MNU> и находится в корневом каталоге. Этот файл вызывается из корневого каталога в том случае, если в текущем каталоге такого файла нет. Если такого файла в корневом каталоге нет, то при вызове данной функцией его можно создать, выбрав в появившейся в этом случае таблице вариант <New-File>. В противном случае нажмите <Esc>. Команда, выполняемая при нажатии какой-либо клавиши, определяется следующим образом. В файле <NC.MNU> в начале строки наберите символ, соответствующий выбранной клавише, например <P> (латинское). Затем через пробел название команды, например <Вызов PCTOOLS>. Эта строка будет высвечиваться на экране при вызове меню по клавише <F2>. Теперь в следующей

строке наберите саму команду так, как Вы ее набрали бы в DOS. Теперь при нажатии клавиши <P> в меню, вызываемому по <F2>, будет запускаться программа <PCTOOLS>.

б) <Local> - то же, что и <Main>, но редактируется или создается файл с именем <NC.MNU>, находящийся в текущем каталоге. Это дает возможность пользователю определять набор команд, необходимых ему в данном каталоге, а соответственно, и с данными типами файлов.

в) <Cansel> - отмена редактирования.

г) <Extension file edit> - редактирование <NC.EXT> файла, определяющего операции, выполняемые над файлами с определенными расширениями при нажатии клавиши <Enter>. После выбора данной команды на экране появляется файл <NC.EXT>, который Вы сможете отредактировать по следующим правилам.

Сначала выберите расширение файлов, операцию с которыми Вы хотели определить, после него поставьте двоеточие. Затем в поле пробела наберите команду, реализующую необходимую операцию.

<Option> (меню определяет внешний вид экрана Norton Commander):

1) <Color> - выбор цветов экрана. После выбора данной команды появляется таблица со следующими вариантами:

<B&W> - переводит дисплей в черно-белый режим;

<Color> - переводит дисплей в цветной режим;

<Lap-top> - предназначена для переносных компьютеров с жидкокристаллическими дисплеями.

2) <Auto menus> - определяет появление меню команд пользователя при запуске Norton Commander.

3) <Path prompt> - определяет появление имени каталога в строке для команд DOS.

4) <Key bar> - позволяет убрать самую нижнюю строку подсказки.

5) <Full screen> - выбор высоты панелей во весь экран или в его половину.

6) <Mini status> - выводит внизу панели новую информацию о выбранном файле или об объеме группы выбранных файлов, если файлы представлены на панели в кратком виде.

7) <Ins move done> - определяет, двигается курсор вниз при выделении файла клавишей <Ins> или нет.

8) <Clock> - позволяет вывести текущее время в правый верхний угол экрана.

9) <Editor> - позволяет выбрать редактор, вызываемый для редактирования выбранного файла нажатием клавиши <F4>. После выбора данной

операции выберите из появившейся таблицы слово 'Built in', если хотите пользоваться встроенным в Norton Commander редактором. Если вы хотите пользоваться другим редактором, выберите 'External', затем выберите имя каталога и название файла редактора, после него через пробел поставьте '!.!', что означает 'редактирование файла с данным именем и расширением'.

10) <Save setup> - сохраняет выбранную конфигурацию Norton Commander, то есть при следующем запуске Norton Commander будет иметь точно такой же вид, как и в момент сохранения конфигурации.

Задание к работе

1. Изучить управление панелями (Ctrl-O, Ctrl-P, Ctrl-U, Ctrl-F1, Ctrl-F2, Alt-F1, Alt-F2, Tab).
2. Вывести информационную панель (Ctrl-L).
3. Создать подкаталог группы D:\USERS\BARMIN\1013NN (F7, Enter).
4. Создать в подкаталоге группы два именных подкаталога D:\USERS\BARMIN\1013NN\SUBDIR1 и SUBDIR2 (F7, Enter).
5. Создать в каждом именном подкаталоге по текстовому файлу (Shift-F4, New-file, Enter).
6. Сохранить созданные файлы (F2), присвоив им имена с расширением .txt, и выйти из режима редактирования (Esc).
7. Скопировать созданный файл из своего именного подкаталога в другой именной подкаталог (F5).
8. Просмотреть скопированные файлы (F3).
9. Переименовать скопированные файлы (F6).
10. Переслать переименованные файлы в другой именной подкаталог (Shift-F6).
11. Удалить все созданные файлы и подкаталоги (F8, Enter).
12. Просмотреть меню Norton Commander (F9).

Лабораторная работа №2
“Интегрированная среда программирования Turbo Pascal 7.0”

Цель работы - знакомство с интегрированной средой и режимами ее работы.

Краткое описание системы программирования Turbo Pascal (версия 7.0)

Среда программирования Turbo Pascal представляет собой интегрированную среду, включающую экранный редактор, компилятор, редактор связей и отладчик. Интегрированная среда позволяет набирать тексты программ с использованием встроенного редактора текстов, компилировать их, выполнять, проводить отладку программ.

Загрузка системы Turbo Pascal осуществляется при запуске файла Turbo.exe, который находится в директории пакета Turbo Pascal (TP). После выполнения загрузки на экране дисплея появляется основной экран интегрированной среды (рис.1), состоящий из трех частей: строки главного меню, поля экрана, строки состояния.

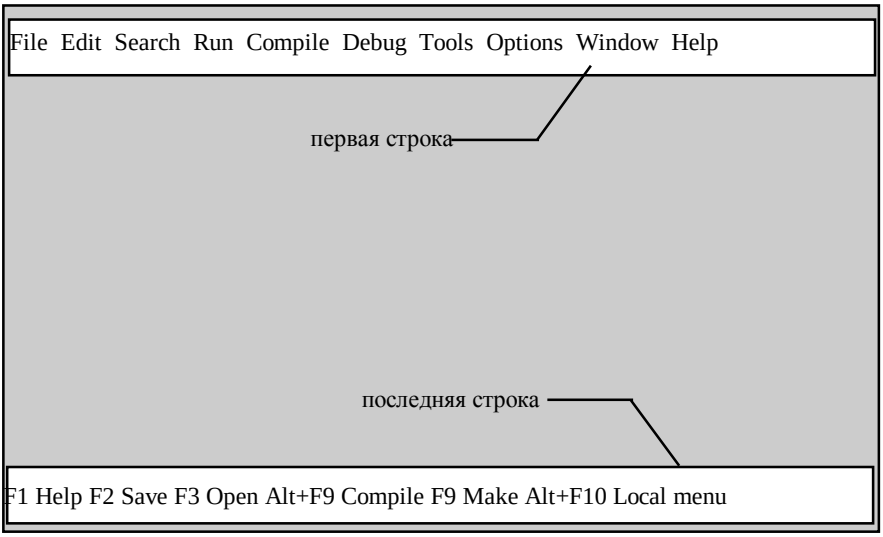


Рис.1

Первая строка содержит все команды главного меню. Вход в главное меню осуществляется с помощью функциональной клавиши F10. В послед-

ней строке экрана приведены основные доступные в каждый текущий момент функциональные клавиши с указанием их назначения. Содержимое строки состояния меняется при изменении режима работы среды.

Главное меню системы обеспечивает следующие режимы работы:

File - команды управления файлами;

Edit - команды текстового редактора для работы с блоками;

Search - команды поиска нужного текста или ошибок в файле;

Run - команды выполнения или пошагового выполнения программы;

Compile - компиляция программы на диск или в память;

Debug - команды отладки программы (оценивать выражения, изменять данные, устанавливать точки прерывания и окно просмотра значений переменных);

Tools - команды трассировки и инструментальные программные средства пользователя;

Options - установка и изменение режимов работы для компилятора, редактора, мыши, отладчика и т.д.

Window - команды работы с окнами (открывать, размещать, просматривать);

Help - вызов информации о работе системы (помощь).

В данном учебном пособии полностью или частично описаны локальные меню следующих режимов работы: File, Edit, Run, Debug, Window.

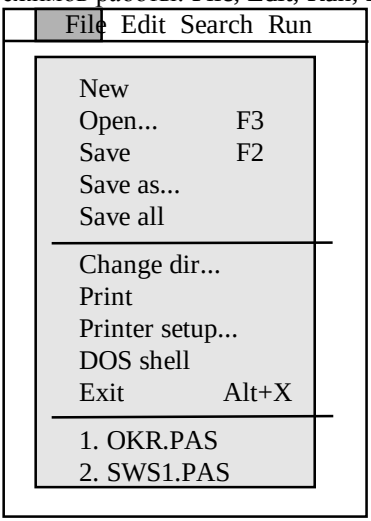


Рис. 2

Локальное меню режима **File** приведено на рис. 2 и содержит следующие команды:

New - создать новый файл в новом окне редактирования. При выборе этой команды на экране появляется окно редактирования (рис. 3).

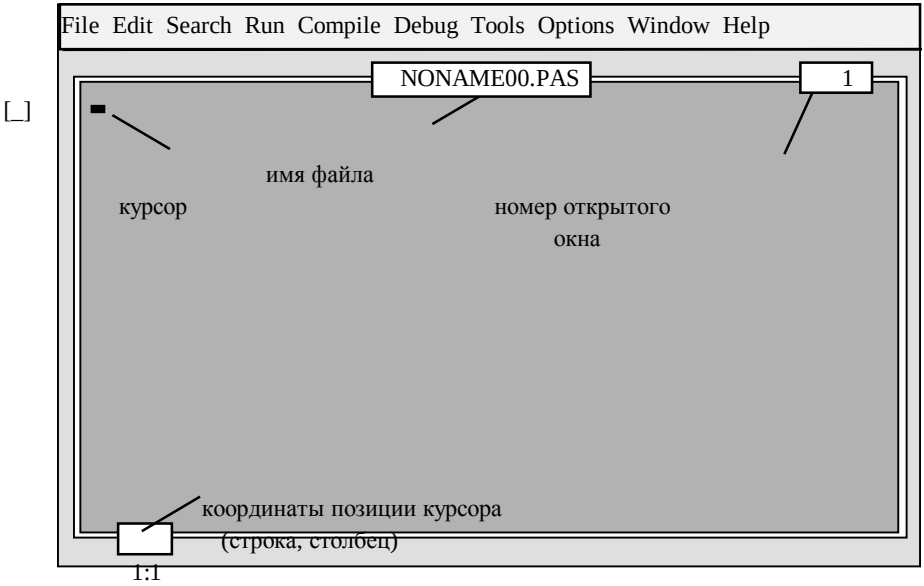


Рис. 3

Имя фала в данном случае Noname00.pas, так как файл еще не записан на диск с определенным именем.

Open (F3) - обнаруживать и открывать файл в окне редактирования. Выбор этой команды ведет за собой появление на экране диалогового окна “Open a file” (рис. 4).

1 - заголовок меню; 2 - строка ввода имени нужного файла; 3 - содержание открытого в данный момент каталога; 4 - строка состояния; 5 - открыть файл с именем, набранным в строке 2, в новом окне редактирования; 6 - загрузить в окно редактирования файл, выбранный курсором в активном окне 3; 7 - закрыть диалоговое окно без изменений; 8 - просмотр помощи о работе в диалоговом окне.

При открытии диалогового окна курсор ожидает ввода имени файла в строке 2. Если необходимо провести поиск нужного файла на диске, нажмите клавишу “Tab” и окно 3 станет активным. Перемещение курсора в окне 3 осуществляется при помощи клавиш со стрелками. При подведении курсора к очередному имени файла,

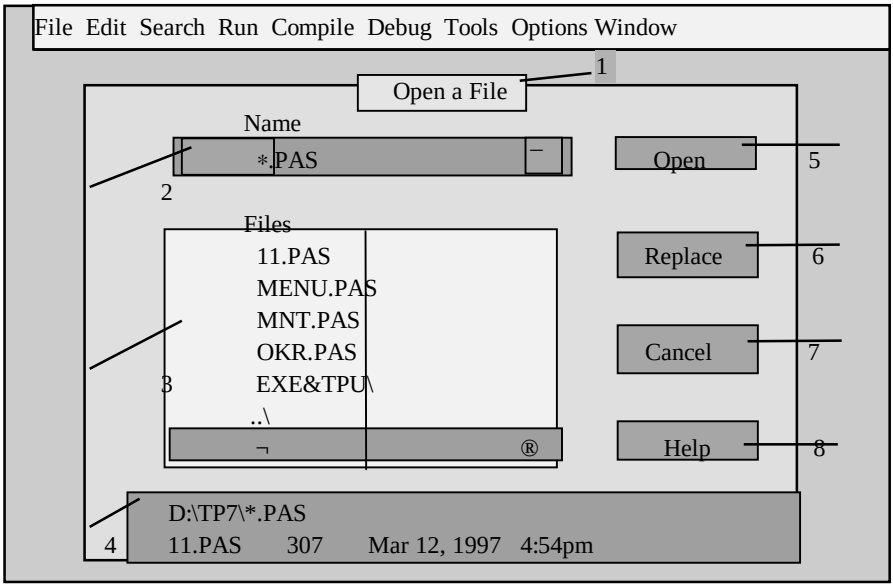


Рис. 4

в строке состояния 4 появляется информация о месте расположения этого файла, его размерности, дате и времени создания. Для подтверждения ввода имени файла или выбора его на диске нажмите клавишу “Enter”, либо при помощи клавиши “Tab” перейдите к окнам 5 или 6. Для выхода из диалогового окна нажмите клавишу “Esc” либо перейдите к окну 7.

Save (F2) - сохранить файл, находящийся в активном окне редактирования.

Save as - сохранить находящийся в активном окне редактирования файл под другим именем либо перенести его в другой каталог или на другой диск. При выборе этой команды на экране появляется диалоговое окно “Save file as”. Работа в нем не отличается от описанной выше работы в диалоговом окне “Open a file” (рис. 4).

Change dir - перейти в другой каталог или на другой диск. Выбор этой команды ведет за собой появление на экране диалогового окна, содержащего дерево каталогов и подкаталогов на активном диске и включающего возможность перехода на другой диск.

Print - печатать содержание файла в активном окне редактирования.

Printer setup - задать фильтр для вывода текста на принтер, тип принтера и возможность выделения различными шрифтами элементов программы.

DOS shell - временный выход в DOS. Для возвращения в Turbo Pascal наберите слово “Exit” в командной строке DOS.

Exit (ALT+X) - выход из системы Turbo Pascal.

Далее в режиме File предложен список файлов, открывавшихся при последнем запуске Turbo Pascal.

Локальное меню режима **Edit** приведено на рис. 5 и содержит следующие команды:

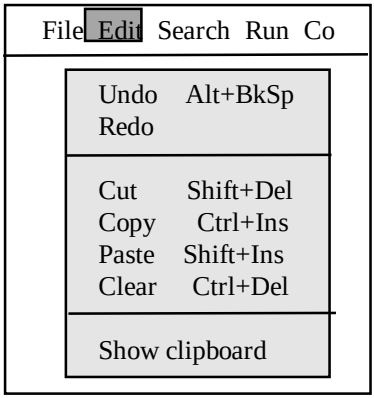


Рис. 5

Undo (Alt+BkSp) - откат, отмена эффекта выполнения предыдущей команды или нескольких предыдущих команд и восстановление состояние обрабатываемого текста.

Redo - возврат, команда, противоположная Undo.

Cut (Shift+Del) - удалить выделенный блок текста из программы и поместить его в буфер обмена (карман) Clipboard.

Copy (Ctrl+Ins) - копировать выделенный блок текста программы без удаления его из текущего файла в буфер обмена.

Paste (Shift+Ins) - вставить блок текста из буфера обмена в текущий файл, начиная с позиции, указанной курсором.

Clear (Ctrl+Del) - удалить выделенный блок текста без записи его в буфер обмена.

Show Clipboard - открыть окно буфера обмена. При выборе этой команды в окне редактирования появляется содержимое буфера обмена. Clipboard можно редактировать так же, как и любой программный файл.

Для выделения блока текста программы необходимо подвести курсор к началу блока, нажать клавишу “Shift” и, удерживая ее, воспользоваться клавишами со стрелками. Отмена выделения производится аналогично.

Локальное меню режима **Run** приведено на рис. 6 и содержит следующие команды:

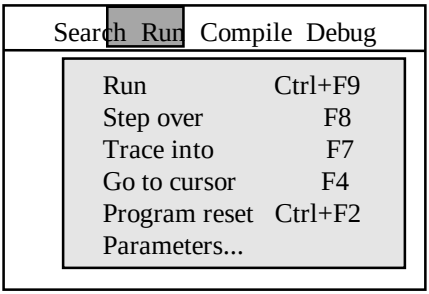


Рис. 6

Run (Ctrl+F9) - выполнить программу, находящуюся в активном окне редактирования.

Step over (F8) - выполнить следующий оператор программы без входа внутрь подпрограммы.

Trace into (F7) - выполнить следующий оператор программы с возможностью выполнения операторов внутри подпрограммы.

Go to cursor (F4) - начать выполнение программы с оператора, в котором находится курсор.

Program reset (Ctrl+F2) - прервать сеанс отладки программы и освободить память.

Parameters - задать параметры программе так же, как они задаются при запуске программы с помощью командной строки.

Локальное меню режима **Compile** приведено на рис.7 и содержит следующие команды:

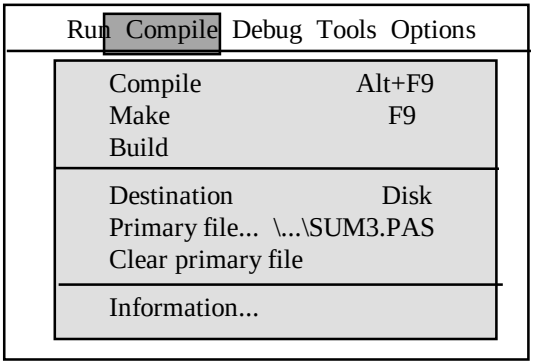


Рис.7

Compile (Ait+F9) - трансляция только программы, указанной в primary file (или находящейся в редакторе, если в primary file отсутствует имя файла). Данная команда позволяет получить также EXE-файл программы, записав его на диск. Для этого необходимо предварительно переключить команду Destination в режим Disk (см. ниже).

Make (F9) - трансляция программы, указанной в primary file (или находящейся в редакторе), и перетрансляция всех модулей пользователя, подсоединенных к данной программе, в которые были внесены изменения.

Build - трансляция программы, указанной в primary file (или находящейся в редакторе), и перетрансляция всех модулей пользователя, подсоединенных к данной программе, вне зависимости от того, были ли внесены в них изменения.

Destination - определяет, где будет сохраняться построенный EXE-файл: в памяти (Memory) или на диске (Disk).

Primary file - определяет файл, который будет компилироваться. Если он не указан, то компилируется файл, загруженный для редактирования.

Clear primary file - выгружает файл, предварительно посланный в primary file.

Information - вывод информации о текущем каталоге, файле, размере файла и программного кода, кода завершения программы, размере стека и данных.

Локальное меню режима **Debug** приведено на рис. 8 и содержит следующие команды:

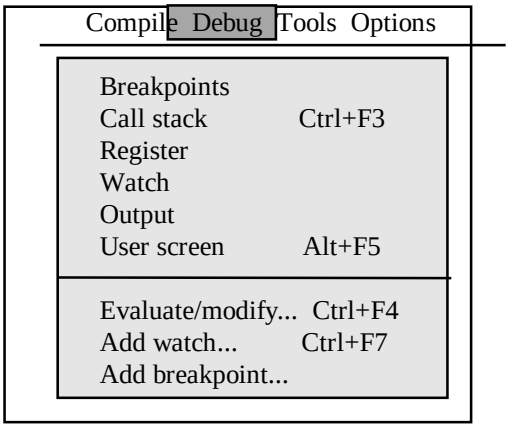


Рис. 8

Breakpoints - установить условные точки прерывания программы, просмотреть и редактировать их. Точка прерывания - контрольная точка программы, в которой ее выполнение прерывается и управление передается отладчику.

Call stack (Ctrl+F3) - открыть окно обращений к процедурам и функциям в текущей точке программы, с целью просмотра значений параметров выделенной процедуры или функции.

Register - открыть окно регистров (внутренних запоминающих устройств процессора или адаптера для временного хранения обрабатываемой или управляющей информации) и их разрядности.

Watch - открыть окно просмотра текущих значений выбранных переменных или выражений программы. Окно просмотра дает возможность наблюдать за изменением значений этих выражений или переменных в процессе пошагового выполнения программы (команды “Trace into” и “Step over” режима Run).

Outptut - открыть окно отображения данных. Окно Outptut показывает текст любой последней команды DOS или текст сгенерированный из вашей программы (не графический).

User screen (Alt+F5) - просмотр полного экрана пользователя (окно Output в полноэкранном режиме).

Evaluate/modify (Ctrl+F4) - команда дает возможность задать переменную или выражение, для которого следует вычислить значение, или задать новое значение для переменной.

Add watch (Ctrl+F7) - вставить наблюдаемое выражение или переменную в окно Watch.

Add breakpoint - определить новую точку прерывания.

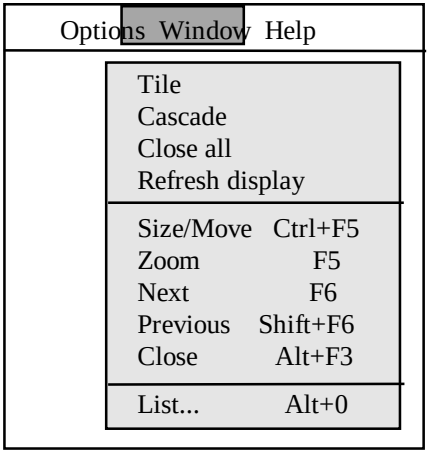


Рис. 9

Локальное меню режима **Window** приведено на рис. 9 и содержит следующие команды:

Tile - упорядочить все открытые окна на рабочем экране в виде непересекающихся окон.

Cascade - упорядочить все открытые окна на рабочем экране в каскадном порядке.

Close all - закрыть все открытые окна.

Refresh display - команда восстановления экрана.

Size/Move (Ctrl+F5) - изменить размеры и позицию на экране активного окна.

Zoom (F5) - увеличить или восстановить размеры активного окна.

Next (F6) - сделать следующее окно активным.

Previous (Shift+F6) - сделать предыдущее окно активным.

Close (Alt+F3) - закрыть активное окно.

List (Alt+O) - показать список всех открытых окон.

Задания к работе

Задание А.

1. Войти в интегрированную среду Turbo Pascal 7.0 (D:\USERS\TP\Turbo.exe).

2. Войти в окно редактирования (F10/File).

3. Сменить каталог на D:\USERS\GR(..номер вашей группы..) (Change dir).

4. Создать новый файл исходной программы по заданию преподавателя (NEW).

5. Вызвать компилятор (Compile/Compile).

6. Исправить ошибки.

7. Запустить выполнение программы (Run/Run).

8. Просмотреть результаты выполнения программы на пользовательском экране (Debug/User screen).

9. Сохранить выполненную программу (File/Save).

10. Распечатать программу (File/Print).

11. Сохранить созданную программу под другим именем (File/Save as).

Задание Б.

1. Открыть созданный программный файл (File/Open).

2. Выделить блок текста программы (Shift/→↓).

3. Удалить выделенный блок текста из программы в карман Clipboard (Edit/Cut).

4. Вставить блок текста из кармана в место, указанное курсором (Edit/Paste).

5. Копировать вставленный блок текста на старое место (Edit/Copy).

6. Удалить выделенный блок текста (Edit/Clear).

7. Отменить предыдущую команду (Edit/Undo).

8. Возвратить предыдущую команду (Edit/Redo).

9. Открыть окно кармана (Edit/Show Clipboard).

10. Получить информацию о программном файле (Compile/Information).

11. Просмотреть другие команды режимов (Debug, Window).

Пример программного файла

```
Program PRIM1;
{Лабораторная работа №1; студент группы ***** Иванов И.И.}
Var
  A, X, Z, Y, P, C: real;
Begin
    writeln('Введите A, X, Z ');
    readln(A,X,Z);  {Ввод исходных данных}
    writeln('A=',A:6:3);  {Вывод введенных значений
```

```
      writeln('X=',X:6:3);    исходных данных}
      writeln('Z=',Z:6:3);
C:=SIN(X*X)/COS(X*X);  {Вычисление тангенса}
Y:=A*C*SQR(X)+SQRT(Z*Z/(A*A+X*X)); {Вычисление значения
                                    функции У}
P:=LN(A+X*X)+SQR(SIN(Z/A));  {Вычисление значения
                                    функции Р}
      writeln('Y=',Y:8:3,'  P=',P:8:3);  Вывод вычисленных
                                          значений У,Р}
End.
```

Лабораторная работа № 3
“Программирование алгоритмов линейной структуры”

Цель работы - овладение практическими навыками разработки и программирования вычислительного процесса линейной структуры и навыками по отладке и тестированию программ.

Задания для самостоятельной подготовки

1. Изучить:
- запись констант, переменных, стандартных функций;
- правила записи арифметических выражений;
- арифметический оператор присваивания;
- организацию простейшего ввода-вывода данных.
2. Разработать алгоритм решения в соответствии с заданием.
3. Составить программу решения задачи.
4. Подготовить тестовый вариант исходных данных и вычислить для них вручную или с помощью микрокалькулятора значения вычисляемых в программе величин.

Задание к работе

1. Вычислить на ЭВМ значения переменных, указанных в таблице (вариант задается преподавателем), по заданным расчетным формулам и наборам исходных данных. На печать вывести значения вводимых исходных данных и результаты вычислений, сопровождая вывод наименованиями выводимых переменных.

2. Модифицировать программу, а затем выполнить ее на ЭВМ таким образом, чтобы вывод вычисленных значений переменных осуществлялся в соответствии со следующей разметкой строк:

```
*****************************
*  Результаты вычислений   *
*****************************
                                < пропуск первой строки >
< имя > = ·····
                                        **************
                                         < имя > = ·····
                                                  *************
```

Таблица

Вариант задания	Расчетные формулы	Значения исходных данных
1	2	3
1	$a = \dfrac{2\cos(x - p\ /\ 6)}{1/\ 2 + \sin^2 y}$; $b = 1 + \dfrac{z^2}{3 + z^2/\ 5}$	x=1,426 y=1,220 z=3,5
2	$g = \left\lvert x^{y/\ x} - \sqrt[3]{y/\ x}\right\rvert$; $c = (y - x)\dfrac{y - z/\ (y - x)}{1 + (y - x)^2}$	x=1,825 y=18,225 z=-3,298
3	$s = 1 + x + \dfrac{x^2}{2!} + \dfrac{x^3}{3!} + \dfrac{x^4}{4!}$; $d = x(\sin x^3 + \cos^2 y)$	x=0,335 y=0,025
4	$y = e^{-bt}\sin(at + b) - \sqrt{\lvert bt + a\rvert}$ $s = b\sin(at^2\cos 2t) - 1$	a=-0,5 b=1,7 t=0,44
5	$v = \sqrt{x^2 + b} - b^2\sin^3(x + a)/\ x$ $y = \cos^2 x^3 - x/\sqrt{a^2 + b^2}$	a=1,5 b=15,5 x=-2,9
6	$s = x^3\,\mathrm{tg}^2(x + b)^2 + a/\sqrt{x + b}$; $Q = \dfrac{bx^2 - a}{e^{ax} - 1}$	a=16,5 b=3,4 x=0,61
7	$R = x^2(x + 1)/\ b - \sin^2(x + a)$ $s = \sqrt{xb/\ a} + \cos^2(x + b)^3$	a=0,7 b=0,05 x=0,5
8	$y = \sin^3(x^2 + a)^2 - \sqrt{x/\ b}$ $z = \dfrac{x^2}{a} + \cos(x + b)^3$	a=1,1 b=0,004 x=0,2

Продолжение таблицы

1	2	3
9	$f = \sqrt[3]{m \operatorname{tg} t + \lvert c \sin t \rvert}$ $z = m \cos(bt \sin t) + c$	m=2 c=-1 t=1,2 b=0,7
10	$y = b \operatorname{tg}^2 x - \dfrac{a}{\sin^2(x/a)}$ $d = ae^{-\sqrt{a}} \cos(bx/a)$	a=3,2 b=17,5 x=-4,8
11	$f = \ln(a + x^2) + \sin^2(x/b)$ $z = e^{-cx} \dfrac{x + \sqrt{x + a}}{x - \sqrt{\lvert x - b \rvert}}$	a=10,2 b=9,2 x=2,2 c=0,5
12	$y = \dfrac{a^{2x} + b^{-x} \cos(a + b)x}{x + 1}$ $R = \sqrt{x^2 + b} - b^2 \sin^3(x + a)/x$	a=0,3 b=0,9 x=0,61
13	$z = \sqrt{ax \sin 2x + e^{-2x}(x + b)}$ $v = \cos^2 x^3 - x/\sqrt{a^2 + b^2}$	a=0,5 b=3,1 x=1,4
14	$U = \dfrac{a^2 x + e^{-x} \cos bx}{bx - e^{-x} \sin bx + 1}$ $f = e^{2x} \ln(a + x) - b^{3x} \ln(b - x)$	a=0,5 b=2,9 x=0,3
15	$z = \dfrac{\sin x}{\sqrt{1 + m^2 \sin^2 x}} - cm \ln mx$ $s = e^{-ax}\sqrt{x + 1} + e^{-bx}\sqrt{x + 1{,}5}$	m=0,7 c=2,1 x=1,7 a=0,5 b=1,08

Пример выполнения работы.

Вычислить на ЭВМ значения *y* и *p*, используя расчетные формулы:

$$y = a \operatorname{tg}^3 x^2 + \sqrt{\frac{z^2}{a^2 + x^2}},$$

$$p = \ln(a + x^2) + \sin^2 \frac{z}{a}$$

при значениях a=0.59, z=-4.8, x=2.1.

Схема алгоритма решения представлена на рис. 3.1.

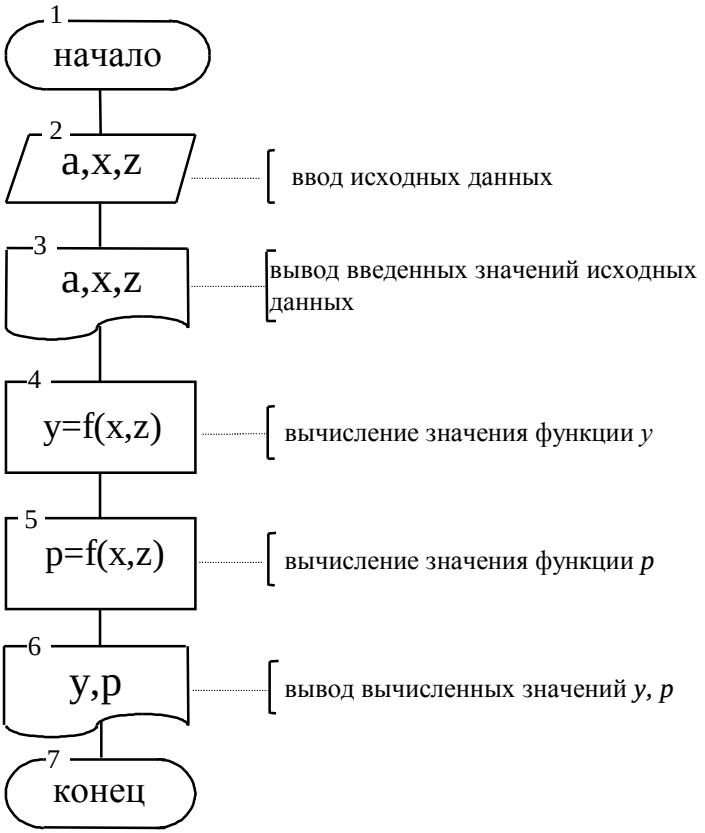


Рис. 3.1

Программа, реализующая схему алгоритма на языке Pascal, имеет вид:

```
Program PRIM3;
{Лабораторная работа №3; студент группы *** Иванов И.И.}
Var
  A, X, Z, Y, P, C: real;
```

```
Begin
        writeln('Введите A, X, Z ');
        read(A,X,Z);  {Ввод исходных данных}
        writeln('A=',A:6:3);  {Вывод введенных значений
        writeln('X=',X:6:3);    исходных данных}
        writeln('Z=',Z:6:3);
C:=SIN(X*X)/COS(X*X);  {Вычисление тангенса}
Y:=A*C*SQR(C)+SQRT(Z*Z/(A*A+X*X)); {Вычисление значения
                                    функции У}
P:=LN(A+X*X)+SQR(SIN(Z/A));  {Вычисление значения
                                функции Р}
     writeln('Y=',Y:8:3,'  P=',P:8:3);  Вывод вычисленных
                                значений У,Р}
End.
```

В качестве тестового набора исходных данных используем следующие значения переменных: a=1, z=1, x=0.5. Тогда вычисляемые значения y и p будут равны:

$$y = 1tg^3(0{,}5)^2 + \sqrt{\frac{1}{1+(0{,}5)^2}} = 0{,}911;$$

$$p = \ln(1+0{,}25) + \sin^2(1) = 0{,}223.$$

Вычисленные значения y и p следует сравнить со значениями, вычисленными на ЭВМ.

Лабораторная работа № 4
“Программирование алгоритмов разветвляющейся структуры”

Цель работы - овладение практическими навыками разработки, программирования вычислительного процесса разветвляющейся структуры, получение дальнейших навыков по отладке и тестированию программы.

Задания для самостоятельной подготовки

Задание А.

1. Изучить возможности языка программирования для реализации:

- условной и безусловной передачи управления;
- вычислительного процесса разветвляющейся структуры.

2. Разработать алгоритм решения в соответствии с заданием.

3. Составить программу решения задачи.

4. Подготовить тесты (число тестов равно числу ветвей вычислительного процесса) для проверки правильности функционирования программы.

Задание Б.

1. Изучить возможности языка программирования для реализации:
- вычислительных процессов с известным числом повторений в цикле;
- приема программирования - табулирования функции от одного аргумента (вычисление значений функции при изменении значения аргумента в заданном диапазоне с шагом Δx).

2. Разработать алгоритм табулирования функции, определенной в задании А.

3. Составить программу табулирования функции.

Задания к работе

Задание А.

1. Вычислить значение функции, заданной в таблице (в соответствии с вариантом задания). Осуществить вывод значений выводимых исходных данных и результат вычисления значений функции, сопровождая вывод наименованиями переменных.

2. Выполнить программу на ЭВМ и протестировать все ветви алгоритма.

Т а б л и ц а

Вар. задан.	Функция	Условие	Исход-ные данные	Диапазон и шаг изменения аргумента
1	2	3	4	5
1	$y = \begin{cases} at^2 \ln t \\ 1 \\ e^{at} \cos bt \end{cases}$	$1 \le t \le 2$ $t < 1$ $t > 2$	a=-0,5 b=2	$t \in [0,5;3]$ Δt=0,15

Продолжение таблицы

1	2	3	4	5
2	$y=\begin{cases} px^2-7/x^2 \\ ax^3+7\sqrt{x} \\ \lg(x+7\sqrt{x}) \end{cases}$	$x<1,3$ $x=1,3$ $x>1,3$	$a=1,5$	$x\in[0,8;2]$ $\Delta x=0,1$
3	$v=\begin{cases} ax^2+bx+c \\ a/x+\sqrt{x^2+1} \\ (a+bx)/\sqrt{x^2+1} \end{cases}$	$x<1,2$ $x=1,2$ $x>1,2$	$a=2,8$ $b=-0,3$ $c=4$	$x\in[1;2]$ $\Delta x=0,05$
4	$Q=\begin{cases} px^2-7/x^2 \\ ax^3+7\sqrt{x} \\ \ln(x+7\sqrt{\lvert x+a\rvert}) \end{cases}$	$x<1,4$ $x=1,4$ $x>1,4$	$a=1,65$	$x\in[0,7;2]$ $\Delta x=0,1$
5	$y=\begin{cases} 1,5\cos^2 x \\ 1,8ax \\ (x-2)^2+6 \\ 3\operatorname{tg} x \end{cases}$	$x<1$ $x=1$ $1<x<2$ $x>2$	$a=2,3$	$x\in[0,2;2,8]$ $\Delta x=0,2$
6	$p=\begin{cases} x^3\sqrt{x-a} \\ x\sin ax \\ e^{-ax}\cos ax \end{cases}$	$x>a$ $x=a$ $x<a$	$a=2,5$	$x\in[1;5]$ $\Delta x=0,5$
7	$Q=\begin{cases} bx-\lg bx \\ 1 \\ bx+\lg bx \end{cases}$	$bx<1$ $bx=1$ $bx>1$	$b=1,5$	$x\in[0,1;1]$ $\Delta x=0,1$
8	$y=\begin{cases} \sin x\lg x \\ \cos^2 x \end{cases}$	$x>3,5$ $x\le 3,5$	–	$x\in[2;5]$ $\Delta x=0,25$

Продолжение таблицы

1	2	3	4	5
9	$f = \begin{cases} \lg(x+1) \\ \sin^2\sqrt{\lvert ax\rvert} \end{cases}$	$x>1$ $x\le 1$	$a=20,3$	$x\in[0,5;2]$ $\Delta x=0,1$
10	$z = \begin{cases} (\ln^3 x + x^2)/\sqrt{x+t} \\ \sqrt{x+t} + \sqrt{x} \\ \cos x + t\sin^2 x \end{cases}$	$x<0,5$ $x=0,5$ $x>0,5$	$t=2,2$	$x\in[0,2;2]$ $\Delta x=0,2$
11	$s = \begin{cases} \dfrac{a+b}{e^x+\cos x} \\ (a+b)/(x+1) \\ e^x + \sin x \end{cases}$	$x<2,8$ $2,8\le x<6$ $x\ge 6$	$a=2,6$ $b=-0,39$	$x\in[0;7]$ $\Delta x=0,5$
12	$y = \begin{cases} a\lg x + \sqrt[3]{\lvert x\rvert} \\ 2a\cos x + 3x^2 \end{cases}$	$x>1$ $x\le 1$	$a=0,9$	$x\in[0,8;2]$ $\Delta x=0,1$
13	$v = \begin{cases} \dfrac{a}{i} + bi^2 + c \\ i \\ ai + bi^3 \end{cases}$	$i<4$ $4\le i\le 6$ $i>6$	$a=2,1$ $b=1,8$ $c=-20,5$	$i\in[0;12]$ $\Delta i=1$
14	$z = \begin{cases} a\sin\left(\dfrac{i^2+1}{n}\right) \\ \cos\left(i+\dfrac{1}{n}\right) \end{cases}$	$\sin\dfrac{i^2+1}{n} > 0$ $\sin\dfrac{i^2+1}{n} < 0$	$a=0,3$ $n=10$	$i\in[1;10]$ $\Delta i=1$
15	$v = \begin{cases} \sqrt{at^2 + b\sin t + 1} \\ at + b \\ \sqrt{at^2 + b\cos t + 1} \end{cases}$	$t<0,1$ $t=0,1$ $t>0,1$	$a=2,5$ $b=0,4$	$t\in[-1;1]$ $\Delta t=0,2$

Задание Б.

1. Модифицировать программу таким образом, чтобы вычислялось многократно значение функции при изменении аргумента в указанном диапазоне и с заданным шагом (табл.). Организовать вывод значения аргумента и вычисленного значения функции в виде таблицы.

Таблица функции У(Х)

Х	У
...	...
...	...

2. Выполнить на ЭВМ модифицированную программу.

Пример выполнения работы

Задание А. Вычислить на ЭВМ значение функции

$$s = \begin{cases} at + b, & \text{если } at<1; \\ \cos at, & \text{если } at=1; \\ e^{-at}\cos at, & \text{если } at>1, \end{cases}$$

для a=1.3, b=1.29, t=0.38.

Схема алгоритма решения представлена на рис.1.

Программа, реализующая на языке Pascal схему алгоритма, представленную на рис. 1, имеет вид:

```
Program PRIM4A;
{Лабораторная работа № 4а}
{Студент группы ******* Иванов И.И.}
Var  A, B, T, S: real;
Begin
     writeln('Введите A, B, T ');
     read(A,B,T);  {Ввод исходных данных}
S:=A*T+B;  {Вычисление значения функции s при любом значении
                                   произведения at}
     if A*T=1 then S:=COS(A*T);  {Вычисление значения функции s
                                   при условии at=1}
     if A*T>1 then S:=EXP(–A*T)*COS(A*T);  {Вычисление значения
                                   функции s при условии at>1}
     writeln('A=',A:8:3, 'B=',B:8:3, 'T=',T:8:3);  {Вывод введенных
                                   значений исходных данных}
     writeln('Результат S=',S:8:3)  {Вывод вычисленного значения
                                   функции s}
End.
```

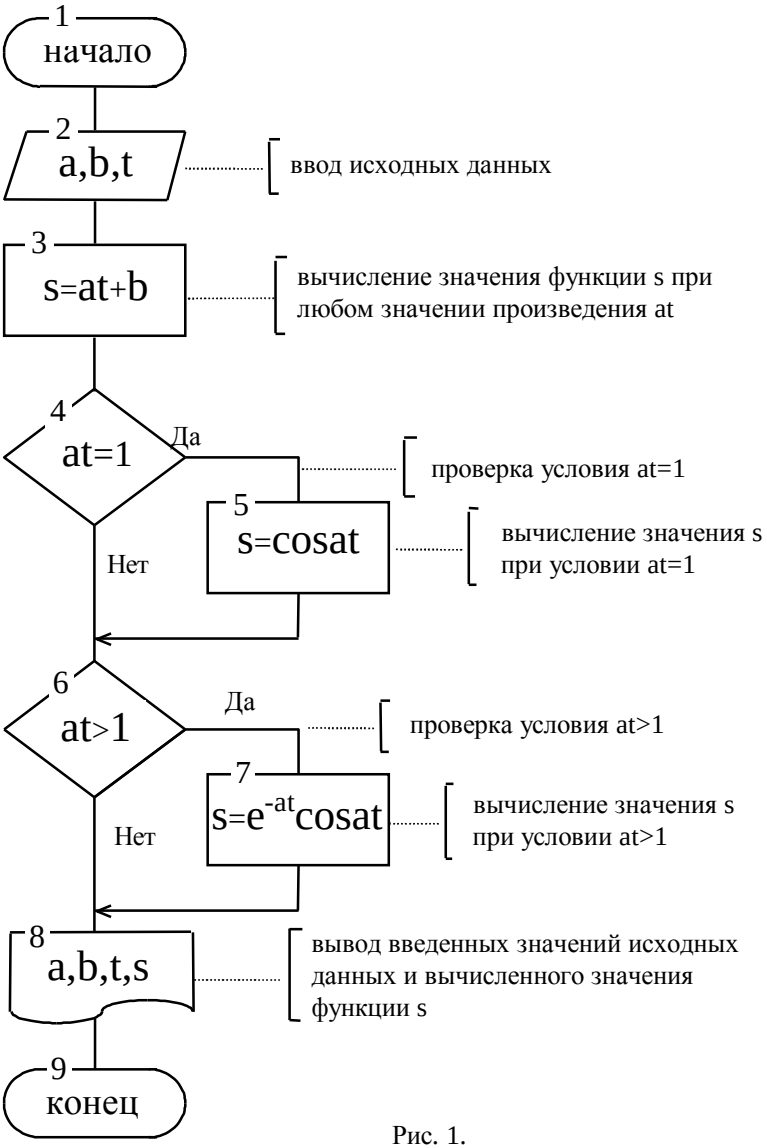


Рис. 1.

В качестве тестовых наборов исходных данных примем следующие тесты:

1) a=1, b=1, t=0.5, s=1·0.5+1=1.5;

2) a=1, b=1, t=1, s=cos(1)=0.5403;

3) a=2, b=1, t=1, $s=e^{-2}\cos(2)=-0.0563$.

Задание Б. Вычислить на ЭВМ значение функции, указанной в задании А при изменении аргумента t в диапазоне $t\in[0.1;\ 2.1]$ с шагом 0.1. Вывод значений t и s выполнить в виде таблицы.

Схема алгоритма решения приведена на рис. 2.

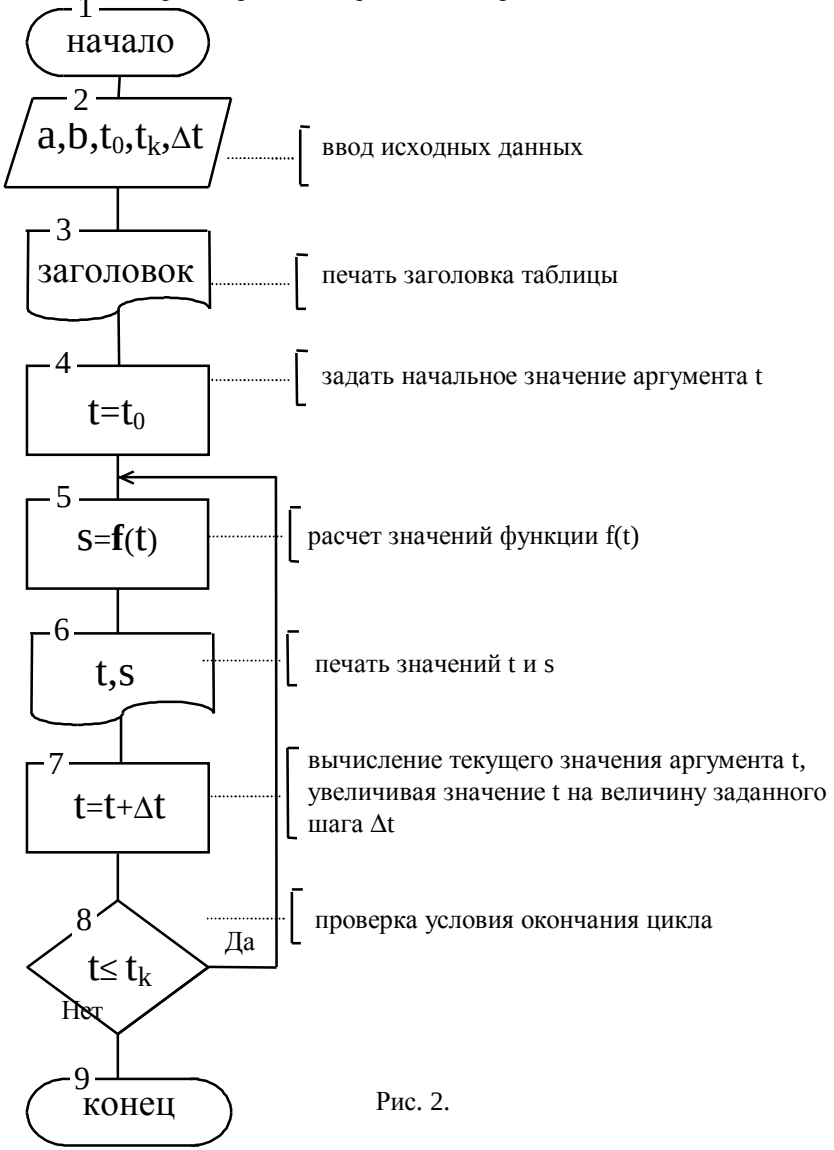


Рис. 2.

Программа, реализованная в соответствии со схемой алгоритма (рис. 2), имеет вид:

```
Program PRIM4B;
{Лабораторная работа № 4б}
{Студент группы ****** Иванов И.И.}
Label 10;
Var
  A, B, T, S: real;
  TO, TK, DT: real;
Begin
    writeln('Введите A, B, TO, TK, DT ');
    read(A,B,TO,TK,DT);  {Ввод исходных данных}
    writeln('Таблица функции S(T)');   {Печать заголовка
    writeln('    T              S(T) ');        таблицы}
    T:=TO;  {Присвоить t начальное значение заданного диапазона}
10:     S:=A*T+B;
        if A*T=1 then S:=COS(A*T); {Вычисление значения функции s
                                    при условии at=1}
        if A*T>1 then S:=EXP(–A*T)*COS(A*T); {Вычисление значения
                                    функции s
                                    при условии
                                    at>1}
        writeln(T:10:3,'    ',S:10:3);  {Вывод на печать значений t и s
                                    в виде табли-
цы}
        T:=T+DT    {Увеличение значения аргумента t на величину
                                    заданного шага}
    if T≤TK then GoTo 10  {Проверка условия окончания цикла при
                                достижении t конечного значения
                                заданного диапазона}
End.
```

Лабораторная работа №5
“Программирование алгоритмов циклической структуры”

Цель работы - овладение практическими навыками разработки и программирования алгоритмов циклической структуры, приобретение дальнейших навыков по отладке программ.

Задания для самостоятельной подготовки

Задание А.

1. Изучить:

- организацию алгоритмов циклической структуры;
- возможности языка программирования для построения циклов;
- приемы программирования - вычисление определенных интегралов по методу прямоугольников и трапеций.

2. Разработать алгоритм решения в соответствии с заданием.

3. Составить программу решения задачи.

4. Для контроля вычислений определить математическое выражение точного значения интеграла и включить вычисление его в программу.

Задание Б.

1. Изучить:

- способы описания размеров массивов на языке программирования;
- способы ввода и вывода массивов;
- реализацию на языке программирования приемов накопления суммы или произведения, запоминания результатов, нахождения наибольшего и наименьшего.

2. Разработать алгоритм решения в соответствии с заданием.

3. Составить программу решения задачи.

4. Подготовить тест для проверки программы.

Задание к работе

Задание А.

Вычислить на ЭВМ значение интеграла

$$z = \int_a^b f(x)\,dx,$$

приведенного в табл. 1, на заданном отрезке интегрирования $[a,b]$ (в соответствии с вариантом задания). Считать заданным число разбиений отрезка интегрирования n и метод численного решения. Включить в программу

вычисление точного значения интеграла по первообразной функции, приведенной в табл. 2. На печать вывести приближенное, точное значения интеграла и относительную погрешность вычисления в процентах.

Примечание. Требуемая точность, указанная в табл.1, в данной работе не используется.

Таблица 1

Вар. задания	Подынтегральная функция f(x)	Метод численного решения	Число разбиений n	Отрезок интегрирования $[a,b]$	Требуемая точность ε
1	2	3	4	5	6
1	$(\ln^2 x)/x$	Трапеций	60	[1;4]	10^{-4}
2	$\frac{1}{x^2}\sin\frac{1}{x}$	Прямоугольников	50	[1;2,5]	$0{,}5\cdot10^{-3}$
3	$x^x(1+\ln x)$	Трапеций	40	[1;3]	10^{-4}
4	$\cos x$	*	60	$[0;\pi/2]$	10^{-4}
5	$\sin^2 x$	*	60	$[0;\pi/2]$	$0{,}5\cdot10^{-3}$
6	$e^x \sin x$	*	100	[0;1]	10^{-4}
7	$((\ln x)/x)^2$	Прямоугольников	50	[1;2,5]	10^{-4}
8	$x\,\mathrm{arctg}\,x$	Трапеций	50	[0;3]	$0{,}5\cdot10^{-3}$
9	$1/\sqrt{9+x^2}$	Прямоугольников	100	[0;2]	10^{-5}
10	$e^x \cos^2 x$	Трапеций	60	$[0;\pi]$	10^{-4}
11	$x^3/(3+x)$	Прямоугольников	80	[1;2]	$0{,}5\cdot10^{-4}$
12	$x^2\ln(x^2+1)$	Трапеций	50	[1;2]	10^{-4}
13	$x\left(\frac{e^x-e^{-x}}{2}\right)$	Прямоугольников	50	[0;2]	10^{-4}
14	$x^2\sin^2 x$	Трапеций	100	[1; 2]	10^{-4}
15	$\frac{\sqrt{x}}{(x^2+1)}$	*	50	[1; 2]	$0{,}5\cdot10^{-3}$

В табл. 2 приведены выражения для вычисления первообразных функций

$$F(x)\Big|_a^b = \int_a^b f(x)\,dx = F(b) - F(a).$$

Таблица 2

Вар. задания	Первообразные функции $F(x) = \int f(x)\,dx.$	Вар. задания	Первообразные функции $F(x) = \int f(x)\,dx.$
1	$(\ln^3 x)/3$	9	$\ln(x + \sqrt{x^2 + 9})$
2	$\cos(1/x)$	10	$\frac{e^x}{5}(\cos^2 x + \sin 2x + 2)$
3	x^x	11	$\frac{x^3}{3} - \frac{3}{2}x^2 + 9x - 27\ln\lvert x+3\rvert$
4	$\sin x$	12	$\frac{x^3}{3x}\ln(x^2+1) - \frac{2x^3}{9} + \frac{2x}{9} - \frac{2}{3}\operatorname{arctg} x$
5	$\frac{x}{2} - \frac{\sin 2x}{4}$	13	$x\operatorname{ch} x - \operatorname{sh} x$
6	$\frac{e^x}{2}(\sin x - \cos x)$	14	$\frac{x^3}{6} - \left(\frac{x^2}{4} - \frac{1}{8}\right)\sin 2x - \frac{x\cos 2x}{4}$
7	$-\frac{1}{x}(\ln^2 x + 2\ln x + 2)$	15	$-\frac{1}{2\sqrt{2}}\ln\left(\frac{x + \sqrt{2x} + 1}{x - \sqrt{2x} + 1}\right) -$ $-\frac{1}{\sqrt{2}}\operatorname{arctg}\left(\frac{\sqrt{2x}}{x-1}\right)$
8	$\frac{(x^2+1)}{2}\operatorname{arctg} x - \frac{x}{2}$		

Задание Б.

1. Обработать на ЭВМ массив в соответствии с вариантом задания, указанного в табл. 3.

Таблица 3

Вар. задан	Массив	Действия	Условия и ограничения
1	2	3	4
1	**X**(100)	Вычислить сумму и количество элементов массива **X**	$0 \le x_i \le 1$
2	**A**(80)	Вычислить среднее арифметическое значение элементов массива **A**	$a_i > 0$
3	**X**(70)	Переписать элементы массива **X** в массив **Y** и подсчитать их количество	$-1 \le x_i \le 1$
4	**B**(50)	Определить максимальный элемент массива **B** и его порядковый номер	$x_i > 0$

Продолжение таблицы 3

1	2	3	4
5	**C**(40)	Вычислить минимальный элемент массива **C** и его номер	x_i <0
6	**D**(80)	Найти максимальный и минимальный элементы массива **D** и поменять их местами	-
7	**Y**(20)	Вычислить среднее геометрическое элементов массива **Y**	y_i >0
8	**Z**(30)	Расположить в массиве **R** сначала положительные, а затем отрицательные элементы массива **Z**	-
9	**N**(50)	Определить сумму элементов массива **N**, кратных трем	$n_i/3*3=n_i$
10	**X**(N)	Вычислить сумму и количество элементов массива **X**	x_i >0, N≤30
11	**A**(N)	Найти среднее геометрическое элементов массива **A**	a_i >0, N≤50
12	**X**(N)	Переписать в массив **Y** подряд положительные элементы массива **X**	x_i >0, N≤40
13	**X**(N)	Переписать подряд в массив **Y** положительные и в массив **Z** отрицательные элементы массива **X**	N≤40
14	**B**(K)	Определить максимальный элемент массива **B** и его порядковый номер	x_i <0, K≤40
15	**C**(K)	Определить минимальный элемент массива **C** и его порядковый номер	-1≤x_i≤1, K≤20

2. Проверить правильность выполнения программы с помощью тестового варианта.

Пример выполнения работы

Задание А.

Вычислить на ЭВМ методом трапеций значение интеграла

$$z = \int_a^b \frac{\sin^2 x}{1 + 2k\cos x + k^2}\,dx \text{ для } a = 0; b = \pi;$$

k=0,5, разбивая отрезок интегрирования на 60 частей. Для контроля вычислить точное значение интеграла и оценить на ЭВМ относительную погрешность метода.

Вычислить точное значение интеграла. Для $|k| \le 0{,}9$

$$\int_0^p \frac{\sin^2 x}{1 + 2k\cos x + k^2}\,dx = \frac{p}{2}.$$

Схема алгоритма решения задачи представлена на рис.1.

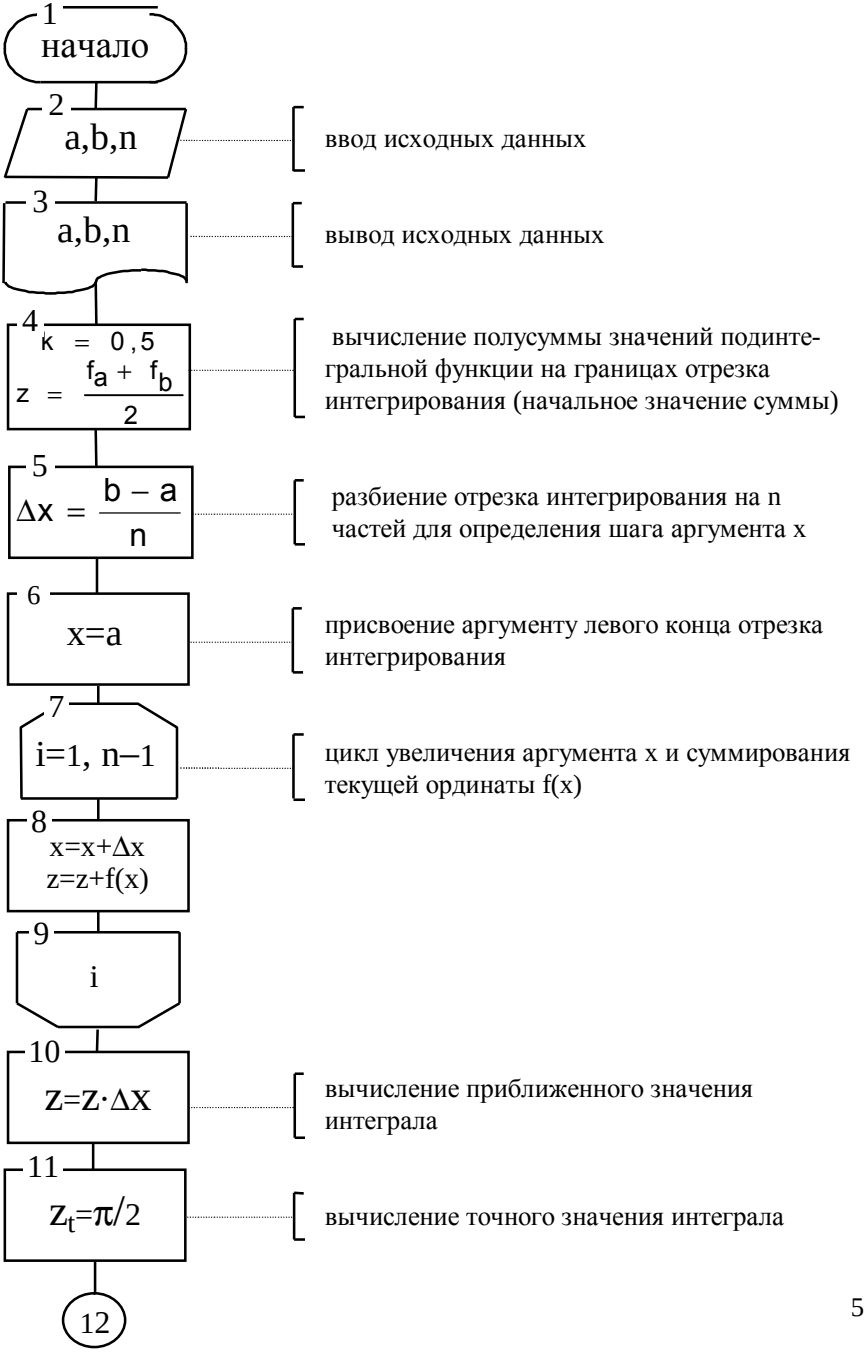


Рис. 1.

Программа, реализующая схему алгоритма (рис. 1), имеет вид:

```
Program PRIM5a;
{Лабораторная работа № 5а}
{студент группы ***** Иванов И.И.}
Var  A, B, K, X, Z, ZT, DZ, DX :real;
     N, I:integer;
Begin
      Writeln(‘ Введите A,B,N ‘ );
      Read(A,B,N);   {Ввод исходных данных}
      Writeln(‘Вывод исходных данных’);
      Writeln(‘A=‘, A, ‘B=‘, B, ‘N=‘, N);
      K:=0,5;
      Z:=(SQR(SIN(A))/(1+2 * K * COS(A)+K * K)+SQR(SIN(B))/(1+
        +2 * K * COS(B)+K * K))/2; {Начальное значение суммы}
      DX:=(B-A)/N;  X:=A;       {Определение шага аргумента
                                  и присвоение ему значения а}
      For i:=1 to N-1 do   {Организация цикла суммирования}
            begin
                  X:=X+DX;
                  Z:=Z+SQR(SIN(X))/(1+2 * K * COS(X)+K * K)
            end
      Z:=Z * DX;       ZT:=PI/2; {Вычисление приближенного и точного
                                                значения интеграла}
      DZ:=ABS(ZT-Z) * 100/ZT;       {Вычисление относительной
                                                погрешности}
      Writeln(‘Приближенное значение Z=‘, Z:9:5 );  {Вывод
      Writeln(‘Точное значение ZT=‘, ZT:9:5);        результатов}
```

```
    Writeln('Относительная погрешность ', DZ:9:4)
End.
```

Задание Б.

Вычислить на ЭВМ наибольший элемент массива $X_1, X_2, \ldots, X_n$ и его порядковый номер; $n \leq 30$. Проверить правильность программы на тесте при $n=3$ и следующих элементах массива (1.5, 4.3, 2.4). При выполнении задания необходимо использовать прием нахождения наибольшего. Для этого перед циклом следует задать начальное значение наибольшего, равное первому элементу массива, а в цикле сравнивать наибольший с текущим элементом массива; в том случае, если текущий элемент больше наибольшего из предыдущих, то считать его наибольшим. Для нахождения порядкового номера наибольшего элемента массива необходимо перед циклом задать его начальное значение, равное 1, а в цикле всякий раз, когда текущий элемент массива больше наибольшего, считать номером наибольшего номер текущего элемента массива. Схема алгоритма решения представлена на рис. 2.

Программа, реализующая алгоритм (рис.2), имеет вид:

```
Program  PRIM5b;
{Лабораторная работа № 5b}
{студент группы ***** Иванов И.И.}
Const      NN=30;
Var
    N,I,IMAX:integer;
    XMAX:real;
    X:array [1..NN]       of     real;
Begin
    Write('Введите значение N');
    Read(N);   {Ввод размерности массива}
    For i:=1  to  N  do  Read(X[i]);   {Ввод массива}
    Writeln('Вывод исходных данных:');
    Writeln(N);
    For i:=1 to N do  Writeln(X[ i ]);    {Вывод массива}
    XMAX:=X[1];  {Начальное значение наибольшего элемента
                                  массива}
    IMAX:=1;   {Начальное значение порядкового номера
                    наибольшего элемента массива}
    For i:=2 to N do   {Цикл перебора элементов массива}
        if  X[i]>XMAX        then
            begin
            XMAX:=X[i];
            IMAX:=i
            end;
    Write('XMAX=',XMAX,'':4,'IMAX=',IMAX)  {Вывод результатов}
```

End.

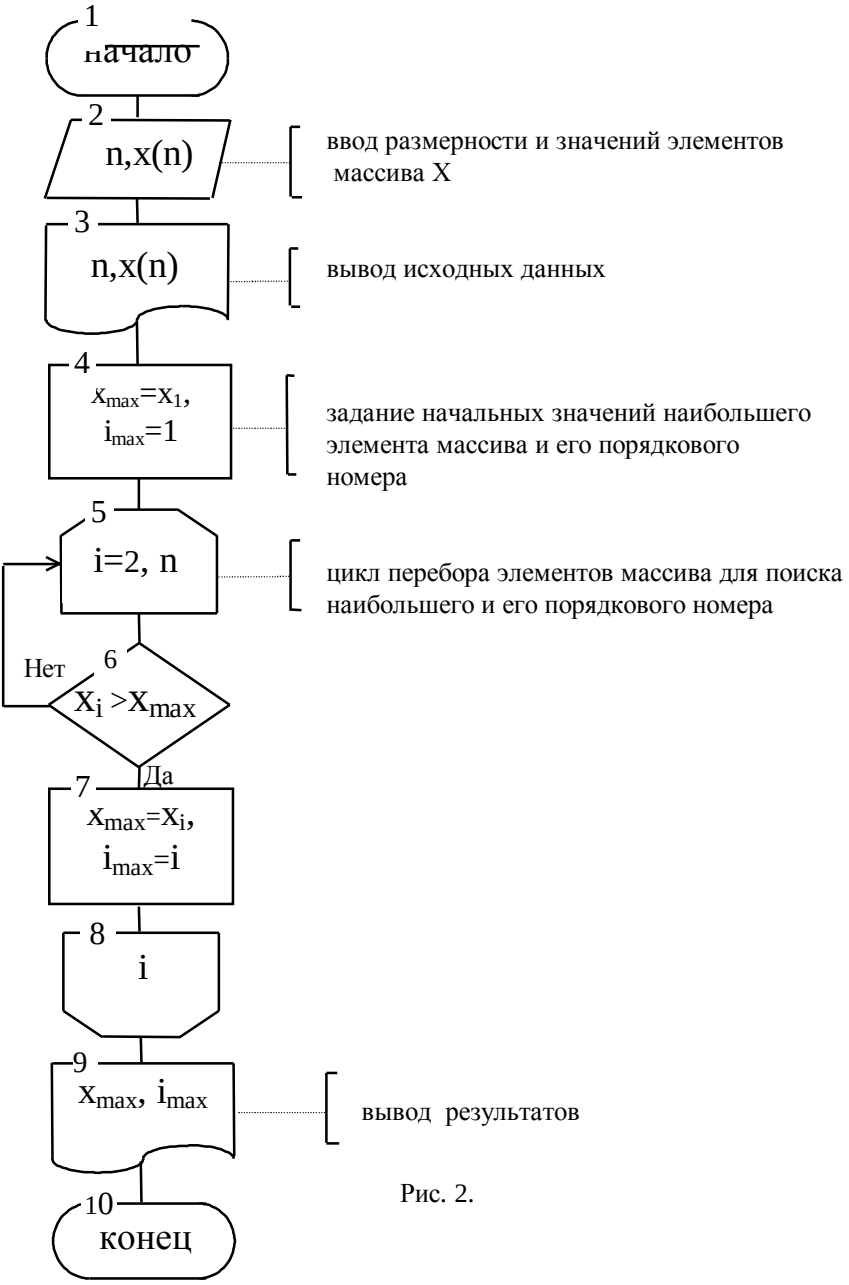

1
начало
2
n,x(n)
ввод размерности и значений элементов массива X
3
n,x(n)
вывод исходных данных
4
$x_{max}=x_1$,
$i_{max}=1$
задание начальных значений наибольшего элемента массива и его порядкового номера
5
i=2, n
цикл перебора элементов массива для поиска наибольшего и его порядкового номера
Нет
6
$x_i > x_{max}$
Да
7
$x_{max}=x_i$,
$i_{max}=i$
8
i
9
x_{max}, i_{max}
вывод результатов
10
конец

Рис. 2.

Лабораторная работа №6
“Программирование алгоритмов со структурой вложенных циклов”

Цель работы - овладение навыками алгоритмизации и программирования вычислительных структур с вложенными циклами.

Задание для самостоятельной подготовки

Задание А.

1. Изучить:

- организацию вычислительных структур с вложенными циклами;
- возможности языка программирования по организации таких структур;
- прием программирования для вычисления определенного интеграла с заданной точностью.

2. Разработать алгоритм решения задачи в соответствии с заданием А.

3. Составить программу решения задачи.

Задание Б.

1. Изучить:

- прием программирования - нахождение экстремума функции с заданной точностью.

2. Разработать алгоритм решения в соответствии с заданием Б.

3. Составить программу решения задачи.

4. Для контроля вычислить точное значение экстремума заданной функции (в таблице указано точное значение аргумента x, при котором достигается экстремум).

Задание к работе

Задание А.

Вычислить на ЭВМ с заданной точностью ε значение опреде-ленного интеграла

$$z = \int_a^b f(x)\,dx,$$

приведенного в табл. 1, лаб. №5. Исходными данными для решения считать значения отрезка интегрирования $[a,b]$, точность вычисления ε и метод численного решения. Включить в программу вычисление точного значения интеграла по первообразной функции из табл. 2, лаб. №5, оценить абсолютную погрешность метода.

Примечание. Значение числа разбиений, указанное в табл. 1, лаб. №5, в данной работе не используется.

Задание Б.

Вычислить на ЭВМ с заданной точностью ε экстремум функции, приведенной в таблице ниже (в соответствии с вариантом задания). На печать вывести вычисленное конечное значение экстремума и значение аргумента, при котором оно достигается.

Таблица

Вар. задания	Вид функции $y = f(x)$	Вид экстремума	Диапазон изменения аргумента $[a,b]$	"Грубое" значение шага h	Точность вычисления экстремума ε
1	2	3	4	5	6
1	$2 + x - x^2$	Максимум 0,5	[0; 1,0]	0,15	10^{-5}
2	$(1-x)^4$	Минимум 1,0	[0,2; 1,5]	0,25	$0{,}5\cdot10^{-4}$
3	$\cos x + \operatorname{ch} x$	Минимум 0,0	[–0.8; 0,4]	0,25	10^{-5}
4	$x^{1/3}(1-x)^{2/3}$	Максимум 0,333333	[0,1; 0,6]	0,1	10^{-5}
5	$x^3 - 6x^2 + 9x + 4$	Максимум 1,0	[0,2; 1,5]	0,3	10^{-5}
6	$x^3 - 6x^2 + 9x + 4$	Минимум 3,0	[2; 4]	0,3	$0{,}5\cdot10^{-5}$
7	$2x^2 - x^4$	Минимум 0,0	[-2; 0,8]	0,15	10^{-4}
8	$\dfrac{x^2 - 3x + 2}{x^2 + 2x - 1}$	Минимум 1,4	[1; 2]	0,15	$0{,}5\cdot10^{-4}$
9	$x\sqrt[3]{x-1}$	Минимум 0,75	[0,1; 1,2]	0,2	10^{-5}
10	xe^{-x}	Максимум 1,0	[0,1; 1,5]	0,25	10^{-5}
11	$(\ln^2 x)/x$	Максимум 7,389	[6; 8]	0,15	10^{-5}
12	$x + 1/x$	Минимум 1,0	[0,1; 1,5]	0,2	10^{-4}
13	$\operatorname{arctg} x - \dfrac{1}{2}\ln(1+x^2)$	Максимум 1,0	[0,15; 1,5]	0,2	10^{-5}

Продолжение таблицы

1	2	3	4	5	6
14	$\lvert x\rvert e^{-\lvert x-1\rvert}$	Максимум -1,0	[-2; -0,5]	0,15	10^{-5}
15	$(\ln^2 x)/\sqrt{x}$	Минимум 1,0	[0,1; 1,9]	0,2	10^{-4}

Пример выполнения работы

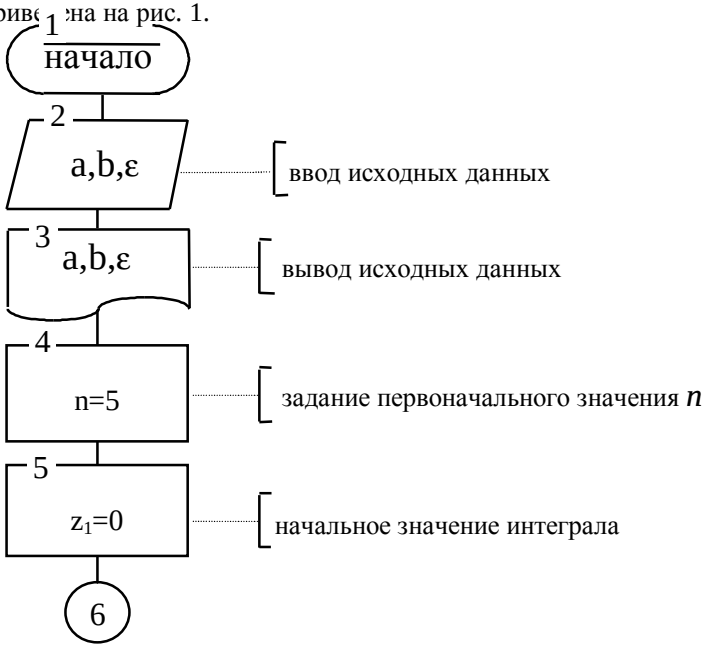
Задание А.

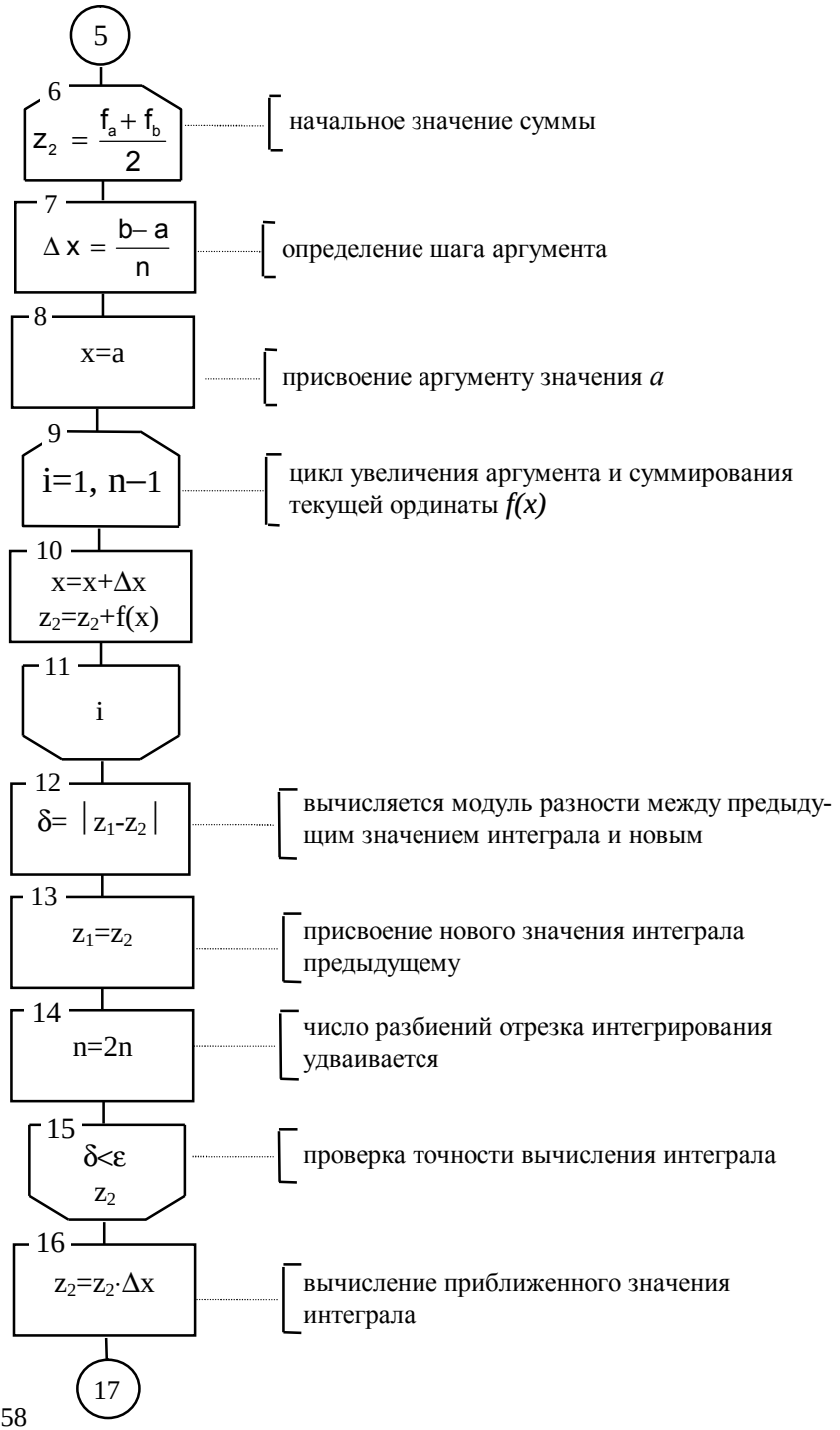
Вычислить на ЭВМ методом трапеций значение интеграла

$$z = \int_a^b \frac{\sin^2 x}{1 + 2k\cos x + k^2} dx$$

с точностью $\varepsilon=10^{-4}$ для a=0; b=π; k=0,5. Для контроля вычислить точное значение интеграла и оценить абсолютную погрешность вычисления.

В примере выполнения лабораторной работы № 5 приведены оценка точного значения данного интеграла, схема алгоритма и программа вычисления для фиксированного значения числа разбиений n отрезка интегрирования. Схема алгоритма решения по методу удвоения числа разбиений n приведена на рис. 1.




5
6
$z_2 = \frac{f_a + f_b}{2}$
начальное значение суммы
7
$\Delta x = \frac{b - a}{n}$
определение шага аргумента
8
x=a
присвоение аргументу значения *a*
9
i=1, n–1
цикл увеличения аргумента и суммирования текущей ординаты *f(x)*
10
x=x+Δx
$z_2=z_2+f(x)$
11
i
12
$\delta = |z_1 - z_2|$
вычисляется модуль разности между предыдущим значением интеграла и новым
13
$z_1=z_2$
присвоение нового значения интеграла предыдущему
14
n=2n
число разбиений отрезка интегрирования удваивается
15
δ<ε
z_2
проверка точности вычисления интеграла
16
$z_2=z_2 \cdot \Delta x$
вычисление приближенного значения интеграла
17

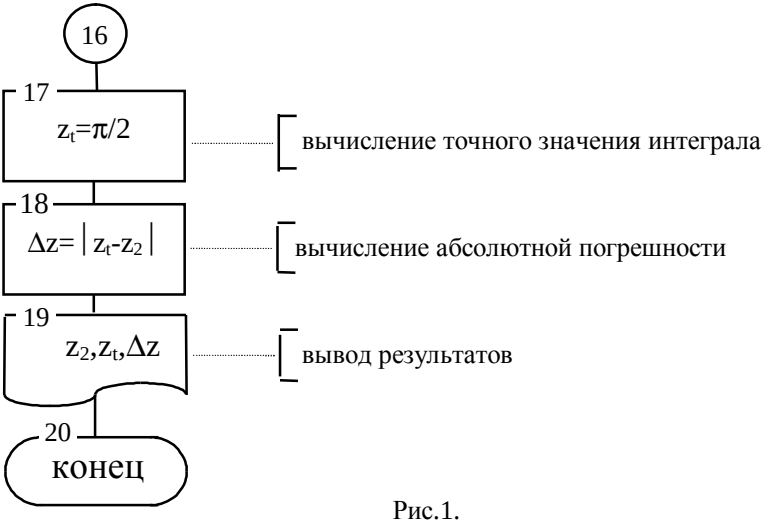


Рис.1.

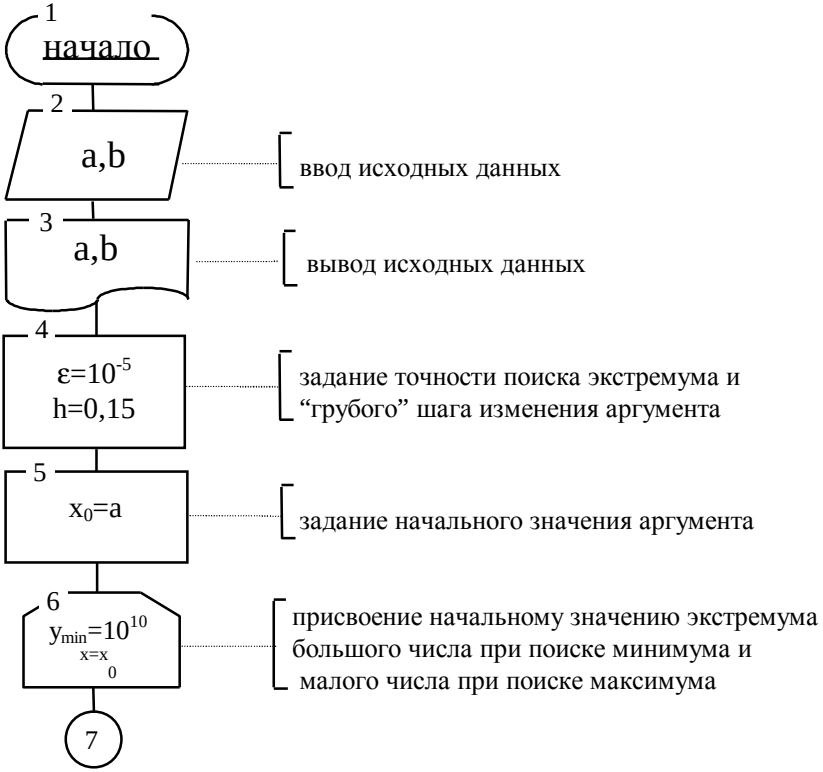
Программа, реализованная в соответствии с алгоритмом (рис. 1), имеет вид:

```
Program PRIM6a;
{Лабораторная работа № 6а}
{Студент группы ****** Иванов И.И.}
Var  A, B, K, X, Z1, Z2, Z3:real;
     DZ, DELTA, DX, ZT, EPS:real;
     N, I:integer;
Begin
 Writeln('Введите A,B,EPS ' );
 Read(A,B,EPS); {Ввод исходных данных}
 Writeln('Вывод исходных данных ',A,B,EPS);
 K:=0.5; N:=5; {Число разбиений отрезка интегрирования}
 Z1:=0; {Начальное значение интеграла}
Repeat  {Цикл проверки точности вычисления интеграла}
 Z2:=(SQR(SIN(A))/(1+2*K*COS(A)+K*K)+ {Начальное значение
   +SQR(SIN(B))/(1+2*K*COS(B)+K*K))/2;       суммы}
DX:=(B-A)/N; {Определение шага аргумента}
 X:=A; {Присвоение аргументу значения а}
     For i:=1 to N-1 do {Организация цикла суммирования}
     begin
          X:=X+DX;
          Z2:=Z2+SQR(SIN(X))/(1+2*K*COS(X)+K*K)
     end;
 DELTA:=ABS(Z1-Z2); {Вычисление модуля разности}
 Z1:=Z2; {Присвоение нового значения интеграла предыдущему}
```

```
  N:=N*2 {Удвоение числа разбиений отрезка интегрирования}
Until  DELTA<EPS; {Проверка точности вычисления интеграла}
  Z2:=Z2*DX; {Приближенное значение интеграла}
  ZT:=PI/2; {Вычисление точного значения интеграла}
  DZ:=ABS(ZT-Z2); {Вычисление абсолютной погрешности}
  Writeln('Приближенное значение Z= ', Z2:9:5);  {Вывод
  Writeln('Точное значение ZT= ', ZT:9:5);          результатов}
  Writeln('Абсолютная погрешность ', DZ:9:4)
End.
```

Задание Б.

Вычислить на ЭВМ с заданной точностью $\varepsilon=10^{-5}$ экстремум функции $y=x\,(x-1)^2\,(x-2)^3$ на интервале [-0,3; 0,5]. Вид экстремума - минимум. "Грубое" значение шага изменения аргумента принять равным 0,15. На печать вывести вычисленное значение экстремума и значение аргумента, при котором оно достигается. Схема алгоритма решения приведена на рис. 2.




6
7
y=f(x)
8
Нет
y<ymin
Да
9
ymin=y
xmin=x
x=x+h
10
y>ymin
y
11
DY = |y − ymin| / y
12
DY≤ε
Да
Нет
13
x0=xmin-h
14
h=h/2
y
15
xmin,ymin
16
конец
цикл вычисления наименьшего значения функции с текущим значением шага h изменения аргумента x
проверка условия прохождения экстремальной точки функции или функция в заданном интервале монотонно убывающая
относительная погрешность определения экстремума функции
проверка условия достижения требуемой точности
задание нового начального значения аргумента в окрестности минимума функции
дробление шага изменения аргумента
вывод результатов

Рис.2

Программа реализующая схему алгоритма (рис.2.), имеет вид:

```
Program PRIM6b;
{Лабораторная работа № 6б}
{Студент группы ****** Иванов И.И.}
Label  M1;
Const EPS = 1E - 5;
Var  A, B, XO, XMIN, Y, YMIN, X, H, DY:real;
Begin
      Writeln('Введите A, B ');
      Read(A, B);   {Ввод исходных данных}
      Write(A,B);  {Вывод исходных данных}
      XO:=A;  H:=0.15;  {Задание начального значения аргумента и
                              шага его изменения}
      While TRUE  do   {Начало внешнего цикла}
        begin
        YMIN:=1E10;   X:=XO;
          Repeat  {Начало вложенного цикла}
                Y:=X*SQR(X-1)*SQR(X-2)*(X-2);
                if  Y<YMIN  then
                      begin
                      YMIN:=Y;   XMIN:=X
                      X:=X+H;
                      if  X>=B  then Writeln('Отсутствие экстремума');
                      end;
            Until  Y>YMIN;  {Конец вложенного цикла}
        DY:=ABS(Y-YMIN)/Y;
        if  DY<=EPS  then  GoTo M1;  {Условие выхода из внешнего
                                                          цикла}
        XO:=XMIN-H;   H:=H/2;
        end;  {Конец внешнего цикла}
M1:  Writeln('XMIN=', XMIN:8:3, '  YMIN = ', YMIN:8:5)   {Вывод
                                                          результатов}
End.
```

В соответствии с принципами структурного программирования для организации итерационного внешнего цикла используется оператор цикла WHILE, а для внутреннего цикла - оператор REPEAT-UNTIL.

Для контроля вычислений определим экстремум функции $y=x(x-1)^2(x-2)^3$ на интервале [-0,3;0,5]. Минимум функции равен $y_{min=}-0,76$.

Лабораторная работа N 7
“Обработка матриц”

Цель работы - овладение навыками алгоритмизации и программирования структур с вложенными циклами, навыками использования приемов программирования во вложенных циклах, способами ввода и вывода матриц.

Задания для самостоятельной подготовки

1.Изучить:

- правила организации вложенного цикла с учетом порядка перебора элементов матрицы;
- правила использования приемов программирования в структурах с вложенными циклами;
- способы ввода и вывода матриц, имеющиеся в языке программирования.

2. Разработать алгоритм решения в соответствии с заданием.

3. Составить программу решения задачи.

4. Подготовить тестовый вариант программы и исходных данных.

Задание к работе

1. Обработать на ЭВМ матрицу в соответствии с вариантом задания, указанного в таблице. Вывести на печать результаты и исходную матрицу в общепринятом виде.

2. Проверить правильность выполнения программы с помощью тестового варианта.

Пример выполнения работы

Выполнить на ЭВМ решение задачи. Записать в массив **B**(N,KMAX) положительные элементы строк матрицы **A**(N,M) ($N \le 20$, $M \le 10$) до первого отрицательного, где KMAX - наибольшее значение числа положительных элементов в строке до первого отрицательного. Вывести на печать сформированную матрицу **B**. На печать выводить только те элементы, которые записаны в матрицу **B**.

Таблица

Вариант задания	Имя матрицы и размеры	Действия	Условия и ограничения
1	2	3	4
1	**A**(10,15)	Вычислить и запомнить сумму и число положительных элементов каждого столбца матрицы. Результаты отпечатать в виде двух строк.	$a_{ij} > 0$
2	**A**(N,M)	Вычислить и запомнить суммы и числа элементов каждой строки матрицы. Результаты отпечатать в виде двух столбцов.	$N \le 20$ $M \le 15$
3	**B**(N,N)	Вычислить сумму и число элементов матрицы, находящихся под главной диагональю и на ней.	$N \le 12$
4	**C**(N,N)	Вычислить сумму и число положительных элементов матрицы, находящихся над главной диагональю.	$c_{ij} > 0$ $N \le 12$
5	**D**(K,K)	Записать на место отрицательных элементов матрицы нули и вывести ее на печать в общепринятом виде.	$K \le 10$
6	**D**(10,10)	Записать на место отрицательных элементов матрицы нули, а на место положительных - единицы. Вывести на печать нижнюю треугольную матрицу в общепринятом виде.	
7	**F**(N,M)	Найти в каждой строке матрицы максимальный и минимальный элементы и поместить их на место первого и последнего элемента строки соответственно. Матрицу напечатать в общепринятом виде.	$N \le 20$ $M \le 10$
8	**F**(10,8)	Транспонировать матрицу и вывести на печать элементы главной диагонали и диагонали, расположенной под главной. Результаты разместить в двух строках.	
9	**N**(10,10)	Для целочисленной матрицы найти для каждой строки число элементов, кратных пяти, и наибольшее из полученных результатов.	n_{ij} /5*5=n_{ij}

Продолжение таблицы

1	2	3	4
10	**N**(10,10)	Из положительных элементов матрицы **N** сформировать матрицу **M** (10, KMAX), располагая их в строках матрицы подряд, где KMAX - максимальное число положительных элементов строки матрицы **N**. Записать нули на место отсутствующих элементов. Отпечатать обе матрицы в общепринятом виде.	
11	**P**(N,N)	Найти в каждой строке наибольший элемент и поменять его местами с элементом главной диагонали. Отпечатать полученную матрицу в общепринятом виде.	N ≤ 15
12	**R**(K,N)	Найти наибольший и наименьший элементы матрицы и поменять их местами.	K ≤ 20 N ≤ 10
13	**S**(25,8)	Вывести исходные данные в первые 24 строки и первые 7 столбцов. Вычислит среднее арифметическое значение элементов каждой строки и записать его в 8-й столбец, а также среднее арифметическое каждого столбца и записать его в 25-ю строку. Отпечатать полученную матрицу в общепринятом виде.	
14	**T**(N,M)	Найти строку с наибольшей и наименьшей суммой элементов. Вывести на печать найденные строки и суммы их элементов.	N ≤ 20 M ≤ 15
15	**V**(15,10)	Упорядочить по возрастанию элементы каждой строки матрицы. Отпечатать полученную матрицу в общепринятом виде.	

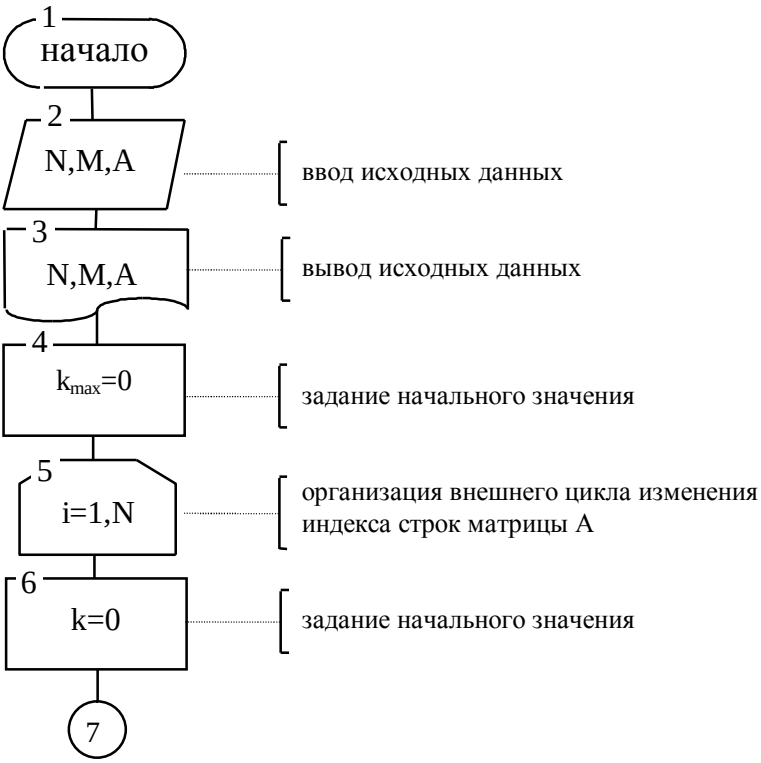
Алгоритм решения следующий. Организовать вложенный цикл для перебора элементов исходной матрицы **A** по строкам. Для этого во внешнем цикле следует изменять индекс строки, а во внутреннем - индекс столбца; во внутреннем цикле - находить и записывать в соответствующую строку матрицы **B** положительные элементы строки до первого отрицательного, а также подсчитывать число положительных элементов K в этой строке. Для подсчета числа положительных элементов необходимо перед внутренним циклом задать его начальное значение, равное 0, а внутри цикла считать число таких элементов, используя оператор присваивания K=K+1.

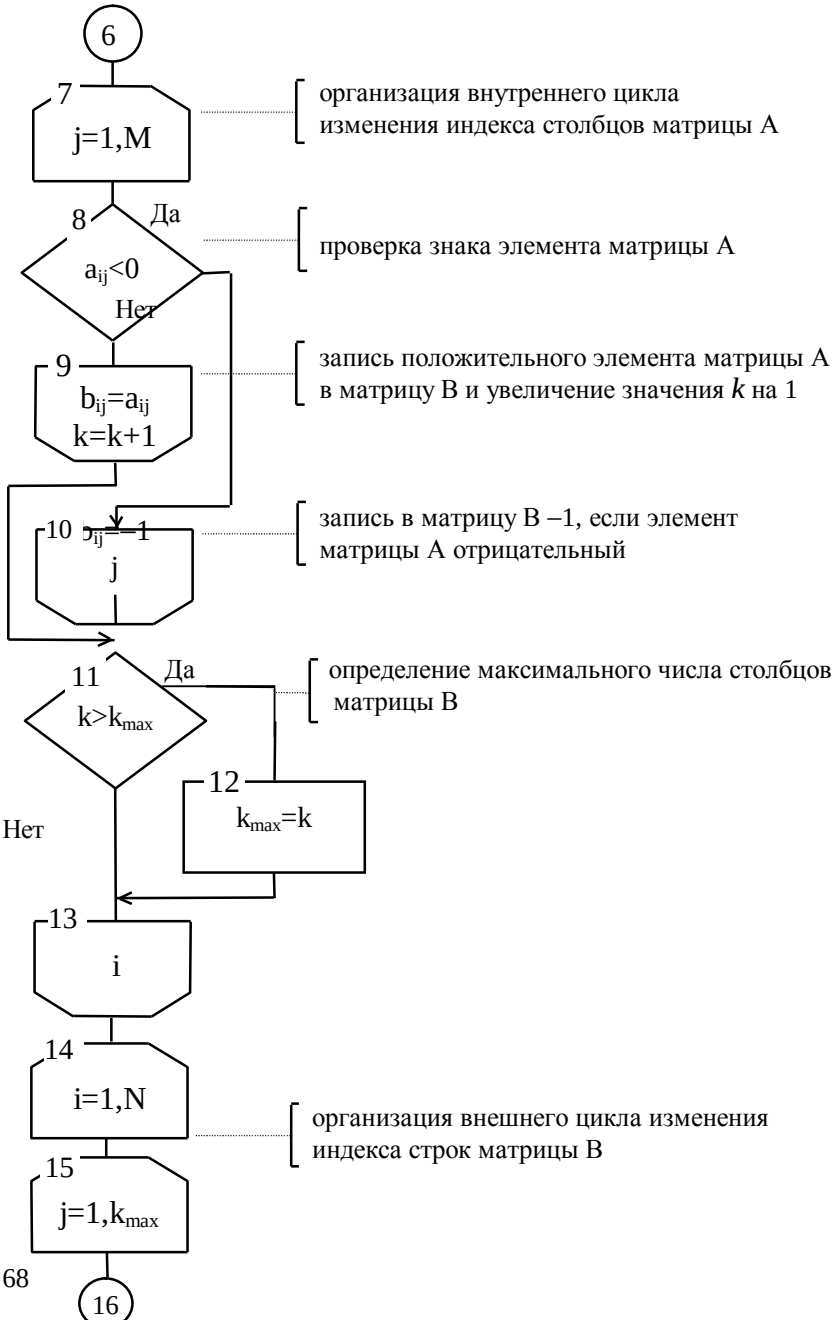
Если положительных элементов в строке нет, то К=0, если в строке элементы положительны, то К=N.

Как только встретиться отрицательный элемент в строке матрицы, необходимо записать в матрицу **B** вместо него -1 и выйти из внутреннего цикла. Во внешнем цикле следует найти наибольшее значение из всех К, вычисленных для отдельных строк. Для этого перед внешним циклом необходимо задать начальное значение KMAX , например, равное нулю, а внутри внешнего цикла сравнить К с KMAX и находить наибольшее из них.

Таким образом, матрица **B** имеет размер N×KMAX. В некоторых строках матрицы **B** будет записано элементов меньше, чем KMAX. По условию задачи на печать необходимо вывести только элементы, записанные в матрицу **B**. Наличие в строках матрицы - 1 указывает на окончание вывода элементов этой строки.

Схема алгоритма решения представлена на рисунке 7.1.




6
7
j=1,M
организация внутреннего цикла изменения индекса столбцов матрицы А
8
Да
$a_{ij}<0$
Нет
проверка знака элемента матрицы А
9
$b_{ij}=a_{ij}$
k=k+1
запись положительного элемента матрицы А в матрицу В и увеличение значения k на 1
10
$b_{ij}=-1$
j
запись в матрицу В –1, если элемент матрицы А отрицательный
11
Да
$k>k_{max}$
определение максимального числа столбцов матрицы В
12
$k_{max}=k$
Нет
13
i
14
i=1,N
организация внешнего цикла изменения индекса строк матрицы В
15
$j=1,k_{max}$
16

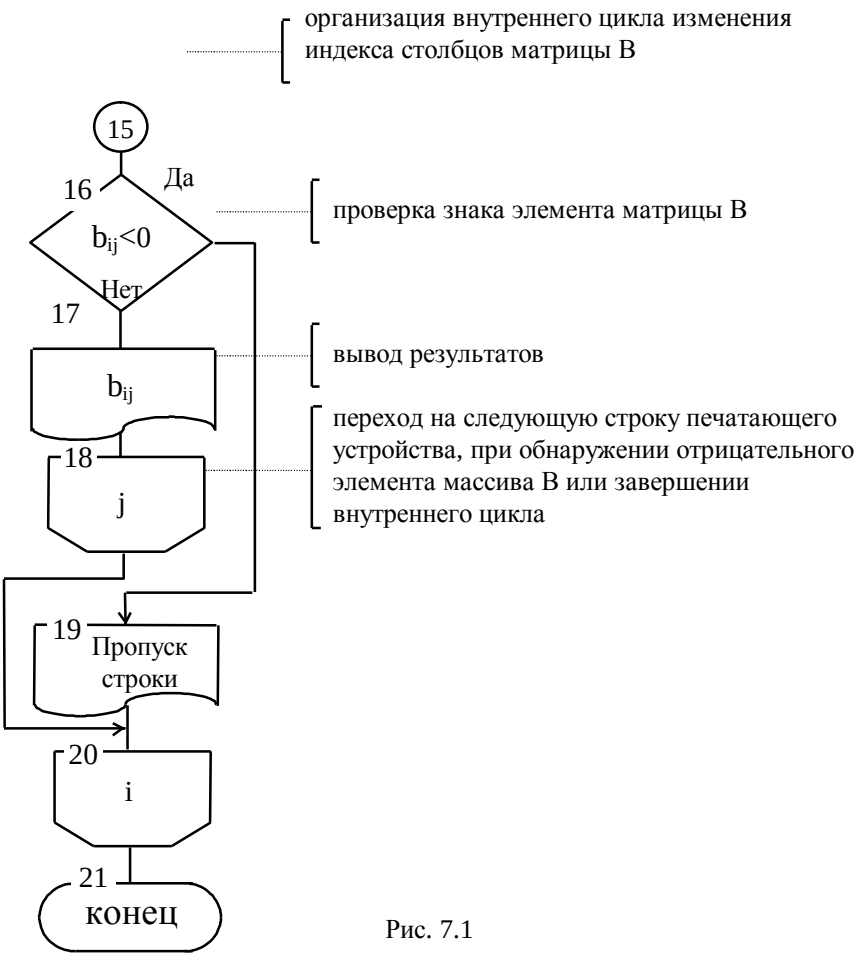


Рис. 7.1

Программа, реализующая алгоритм решения задачи на языке Pascal, имеет вид:

```
Program PRIM7;
{Лабораторная работа № 7}
{Студент группы ******* Иванов И.И.}
Label 10;
Const N=20;  M=10;
Var I, J, NR, MR, K, KMAX:Integer;
```

```
    A, B:array [1..N,1..M] of real;
Begin
    Writeln('Введите значение NR,MR');
    Read(NR,MR); {Ввод числа строк и столбцов матрицы A}
    Writeln(Введите значения матрицы A');
    for  i:=1 to NR  do
    for  j:=1 to MR  do
        begin
            Read(A[i,j]); {Ввод матрицы A}
            Writeln(A[i,j])  {Вывод исходных данных}
        end;
    KMAX:=0; {Задание начального значения}
    for  i:=1  to  NR  do  {Организация внешнего цикла}
        begin
            K:=0; {Задание начального значения}
            for  j:=1 to MR do {Организация внутреннего цикла}
              if A[i,j]>0 then {Проверка знака элемента матрицы A}
                 begin
                    B[i,j]:=A[i,j] {Запись положительных элементов
                                    матрицы A в B и увеличение K на
1}
                    K:=K+1;
                 end
               else
                 begin
                    B[i,j]:=-1; {Запись в матрицу B  –1}
                    Goto 10
                 end;
          10: if  K>KMAX   then  KMAX:=K; {Определение макси-
                                  мального числа столбцов матрицы B}
        end;
    for  i:=1 to NR  do {Организация цикла вывода результатов}
        for  j:=1 to KMAX  do
            if  B[i,j]>0  then
                Write(B[i,j])
            else
                Writeln
End.
```

Тестовая проверка. Выполнить программу при N=4 и M=4, приняв следующие значения исходной матрицы **A**:

$$\begin{pmatrix} 1.5 & 2.0 & -1.0 & 2 \\ 3.3 & 4.4 & 5.5 & 6.6 \\ -2 & 3. & 4. & 5. \\ 0.0 & -2. & 7. & -4 \end{pmatrix}$$

В результате выполнения программы выводится матрица в виде:

1.5 2.0

3.3 4.4 5.5 6.6

<пустая строка>

0.0

Правильность выполнения программы легко устанавливается сравнением исходной матрицы **A** с результатом печати матрицы **B**. В тестовом наборе данных рассмотрены следующие случаи:

1) в строке есть отрицательный элемент;

2) в строке нет отрицательных элементов;

3) первый элемент строки - отрицательный.

Использование указанных случаев позволяет убедиться в работоспособности программы при любых наборах данных.

Лабораторная работа N 8
“Программирование с использованием подпрограмм пользователя”

Цель работы - овладение навыками алгоритмизации и программирования задач с использованием подпрограмм пользователя различных видов, овладение навыками написания подпрограмм и обращения к ним, выбора параметров подпрограмм.

Задания для самостоятельной подготовки

1. Изучить:
   - правила записи подпрограмм различных видов и способов обращений к ним;
   - способы передачи параметров в подпрограмму;
   - правила записи программ, использующих подпрограммы различных видов;
   - порядок выполнения программ, использующих подпрограммы.
2. Разработать алгоритм решения в соответствии с заданием.
3. Составить программу решения задачи.
4. Подготовить тестовый вариант программы и исходных данных.

Задание к работе

Задание А.

1. Выполнить на ЭВМ программу, использующую подпрограмму-функцию, в соответствии с заданием, указанным в табл. 1.

Таблица 1

Вар. задан.	Условие задачи	Примечания
1	2	3
1	Вычислить большие корни квадратных уравнений $x^2 - ax + b = 0$ $cy^2 - dy - f = 0$	Все корни действительные
2	Подсчитать число точек, находящихся внутри круга радиусом r с центром в начале координат; координаты заданы массивами **X**(100), **Y**(100)	Расстояний точки от начала координат вычислять в подпрограмме
3	Определить периметры треугольников, заданных координатами их вершин XA(5), XB(5), XC(5) YA(5), YB(J), YC(5)	Длину стороны треугольников вычислять в подпрограмме

Продолжение таблицы 1

1	2	3
4	Подсчитать число точек, находящихся внутри круга радиусом r с центром в точке с координатами (1,1); координаты заданы массивами **X**(80), **Y**(80).	Расстояние точки от центра круга определять в подпрограмме
5	Вычислить $z = \frac{v_1 + v_2 + v_3}{3}$, где v_1, v_2, v_3 - объемы шаров с радиусами r_1, r_2, r_3 соответственно	v_i вычислять в подпрограмме
6	Вычислить суммы положительных элементов массивов **X**(N), **Y**(M), **Z**(K).	N≤60, M≤60 K≤70
7	Вычислить среднее арифметическое положительных элементов для массивов **A**(N1), **B**(N2), **C**(N3).	N1≤100 N2≤100 N3≤100
8	Подсчитать количество элементов матриц **X**(10,15) и **Y**(20,12), удовлетворяющих условиям $0 \le x_{ij} \le 1$ и $0 \le y_{ij} \le 1$.	
9	Вычислить суммы положительных элементов каждой строки для матриц **A**(10,12) и **B**(15,10).	
10	Вычислить $z = \frac{x_{m1} + x_{m2}}{2}$, где x_{m1} и x_{m2} - наименьшие элементы массивов **X1**(70), **X2**(80).	
11	Вычислить суммы элементов главных диагоналей матриц **A**(N,N), **B**(M,M).	M≤20 N≤30
12	Вычислить $z = \frac{s_1 + s_2}{2}$, где s_1 - сумма положительных элементов массива **X**(50); s_2 - сумма отрицательных элементов масси-ва **Y**(60).	Обе суммы вычислять в одной подпрограмме
13	Подсчитать число нулевых элементов для матриц **A**(N,M) и **B**(M,N).	M≤20 N≤20
14	Вычислить сумму элементов нижних треугольных матриц для матриц **A**(15,15) и **B**(20,20).	
15	Определить число положительных эле-ментов до первого отрицательного в массивах **X**(40), **Y**(50), **Z**(N).	N≤50

2. Проверить правильность выполнения программы с помощью тестового варианта.

Задание Б.

1. Выполнить на ЭВМ программу, использующую подпрограмму-процедуру в соответствии с номером, указанным в табл. 2.

Таблица 2

Вар. Задан.	Условие задачи	Примечания
1	2	3
1	Вычислить $z = \frac{s_1 + s_2}{k_1 k_2}$, где s_1 и k_1- сумма и количество положительных элементов мас-сива **X**(N); s_2 и k_2 - сумма и количество положительных элементов массива **Y**(M).	M≤100 N≤100
2	Вычислить $z = \frac{e^{s1} + e^{s2}}{k_1 k_2}$, где s_1 и k_1- сумма и количество положительных элементов массива **X**(100); s_2 и k_2 - сумма и количество положительных элементов массива **Y**(80).	Обе суммы вычислять в одной подпрограмме
3	Вычислить и запомнить суммы поло-жительных элементов каждой строки матрицы **A**(10,20), **B**(15,10).	
4	Вычислить $z = (x_1 + y_1) / (x_2 - y_2)$, где x_1 и x_2 - корни уравнения $2x^2+x-4=0$, y_1 и y_2 - корни уравнения $ay^2+2y-1=0$	Все корни действительные
5	Найти наибольшие элементы и их порядковые номера массивов **X**(N) и **Y**(M).	N≤80 M≤70
6	Переписать положительные элементы массива **X**(100) и **Y**(80) в массив **Z** подряд.	Запись в массив **Z** выполнить в подпрограмме
7	Найти наименьшие элементы и номера строк и столбцов, в которых они расположены, для матриц **A**(10,15) и **B**(15,12).	
8	Вывести на печать элементы целочисленных матриц **N**(5,8) и **M**(10,6), кратные трем.	
9	Вычислить $z = \frac{\sum_{i=1}^{40} \sin x_i + \sum_{i=1}^{50} \cos y_i}{\sum_{i=1}^{40} \lvert x_i \rvert}$, где x_i и	Все суммы вычислять в одной подпрограмме

	y_i заданы массивами.	

Продолжение таблицы 2

1	2	3
10	Вычислить $z = \frac{x_{max} - y_{min}}{2}$, где x_{max} - максимальный элемент массива **X**(50); y_{min} - минимальный элемент массива **Y**(40)/	x_{max} и y_{min} вычислять в одной подпрограмме
11	Вычислить и запомнить количество отрицательных элементов каждого столбца для матриц **A**(10,10), **B**(15,20).	
12	Вычислить суммы элементов верхней треугольной матрицы для матриц **A**(10,10), **B**(15,15).	
13	Найти средние значения и стандартные отклонения для элементов массивов **X**(N), **Y**(M).	N≤100 M≤100
14	Вычислить суммы и количества элементов, находящихся в интервале от a до b для матриц **X**(10,8) и **Y**(10,12).	
15	Преобразовать массивы **X**(50) и **Y**(60), расположив в них подряд только положитель-ные элементы. Вместо остальных элементов записать нули.	

2. Проверить правильность выполнения программы с помощью тестового варианта.

Пример выполнения работы

Задание А.

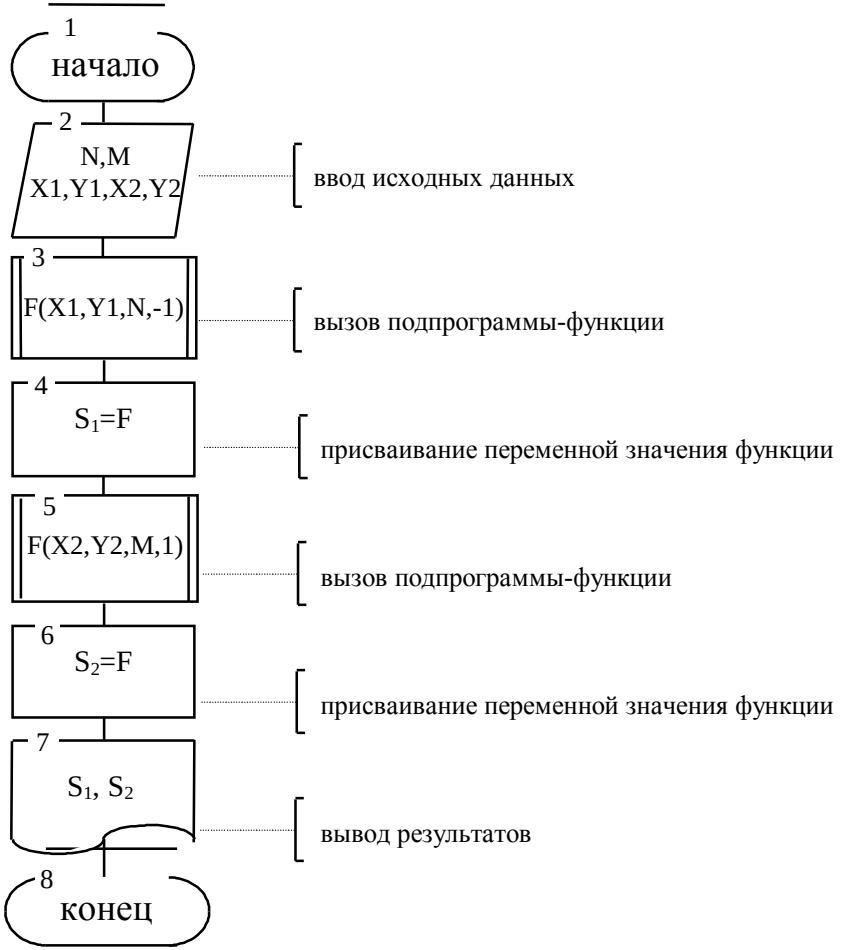
1. Выполнить на ЭВМ решение задачи. Определить ближайшую к началу координат точку, находящуюся в верхней полуплоскости, и наиболее удаленную точку, лежащую в нижней полуплоскости. Координаты точек, находящихся в верхней полуплоскости, заданы массивами **X1**(N) и **Y1**(N), а лежащих в нижней полуплоскости - массивами **X2**(M) и **Y2**(M), где N≤40, M≤60.

Для каждой точки верхней полуплоскости следует определить расстояние от начала координат. Из этих расстояний необходимо найти наименьшее. Такие же действия выполнить для точек, находящихся в нижней полуплоскости, однако найти наибольшее расстояние от начала координат.

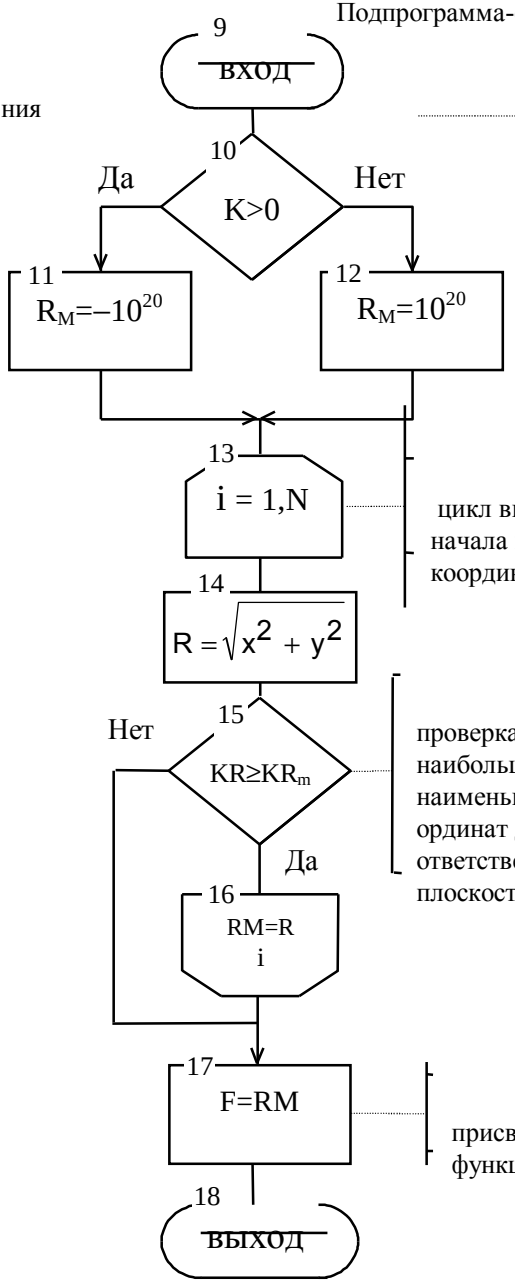
Вычисление расстояний от начала координат и нахождение наименьшего или наибольшего из них выполним в подпрограмме-функции.

Использование одной подпрограммы для нахождения наибольшего или наименьшего значений потребует введения дополнительного параметра, который необходим для проверки условия K∗R>K∗RM. Если K=1, то условие R>RM используется для нахождения наибольшего; если K=-1, то условие R<RM используется для нахождения наименьшего.

В подпрограмму необходимо передать массивы координат точек, их размер, а также параметр К, который может принимать значения +1 или -1. Результат, полученный в подпрограмме-функции, присваивается ее имени. Схема алгоритма решения задачи представлена на рис. 8.1.



Подпрограмма-функция



задание начального значения R_M от K

цикл вычисления расстояния R от начала координат до точки, заданной координатами массивов X и Y

проверка условия при K=1 нахождения наибольшего, а при K=-1 нахождения наименьшего расстояния от начала координат до точки, расположенной соответственно в нижней и верхней полуплоскостях

присвоение имени подпрограммы-функции результата вычисления

Рис. 8.1

2. Проверить правильность выполнения программы для массивов, заданных значениями

X1={-4, 0, 3}; **Y**1={0, 5, 4}; **X**2={-4, 0, 2}; **Y**2={-2, -5, -1},

для которых наименьшее расстояние от начала координат в верхней полуплоскости s_1=4, а наибольшее - в нижней полуплоскости s_2=5.

Задание Б.

1. Решить на ЭВМ задачу. Переписать положительные элементы массивов **X**(N), **Y**(M) в массив **Z** подряд. Запись положительных элементов в массив осуществить в подпрограмме. Принять ограничения: N<=100 и M<=100.

В подпрограмме должна осуществляться запись положительных элементов исходного массива в массив результатов. Для этого в подпрограмму необходимо передать следующие параметры: имя и количество элементов исходного массива, имя и количество элементов результирующего массива. Поскольку в массив результатов **Z** записываются подряд положительные элементы из нескольких массивов, в списке параметров должны фигурировать также: входной параметр L - номер ячейки, начиная с которой необходимо осуществлять запись в массив результатов; выходной параметр К - номер ячейки, в которую записан последний положительный элемент исходного массива при предыдущем обращении к подпрограмме. После окончания записи элементов в массив результатов этот параметр определяет количество элементов, записанных в массив результатов.

При первом обращении к подпрограмме в нее необходимо передать имя массива **X**. Количество его элементов N; входной параметр L=1, если запись осуществляется в массив **Z**, начиная с элемента с индексом 1, имя выходного массива **Z**, количество его элементов N+M и получать в подпрограмме выходной параметр К, определяющий количество элементов, записанных в массив **Z**.

При втором обращении необходимо передать соответственно Y, M, L=K+1, Z, N+M, K. После второго обращения к подпрограмме К - суммарное количество элементов, записанных в массив **Z** после двух выполнений подпрограммы.

Схема алгоритма решений этой задачи имеет вид, представленный на рис. 2. В схеме алгоритма при первом обращении к подпрограмме параметр L задан равным нулю, так как в подпрограмме перед записью элемента в массив **Z** индекс К увеличивается на 1. По этой же причине при втором обращении параметру L присваивается значение К, т.е. меньшее на 1.

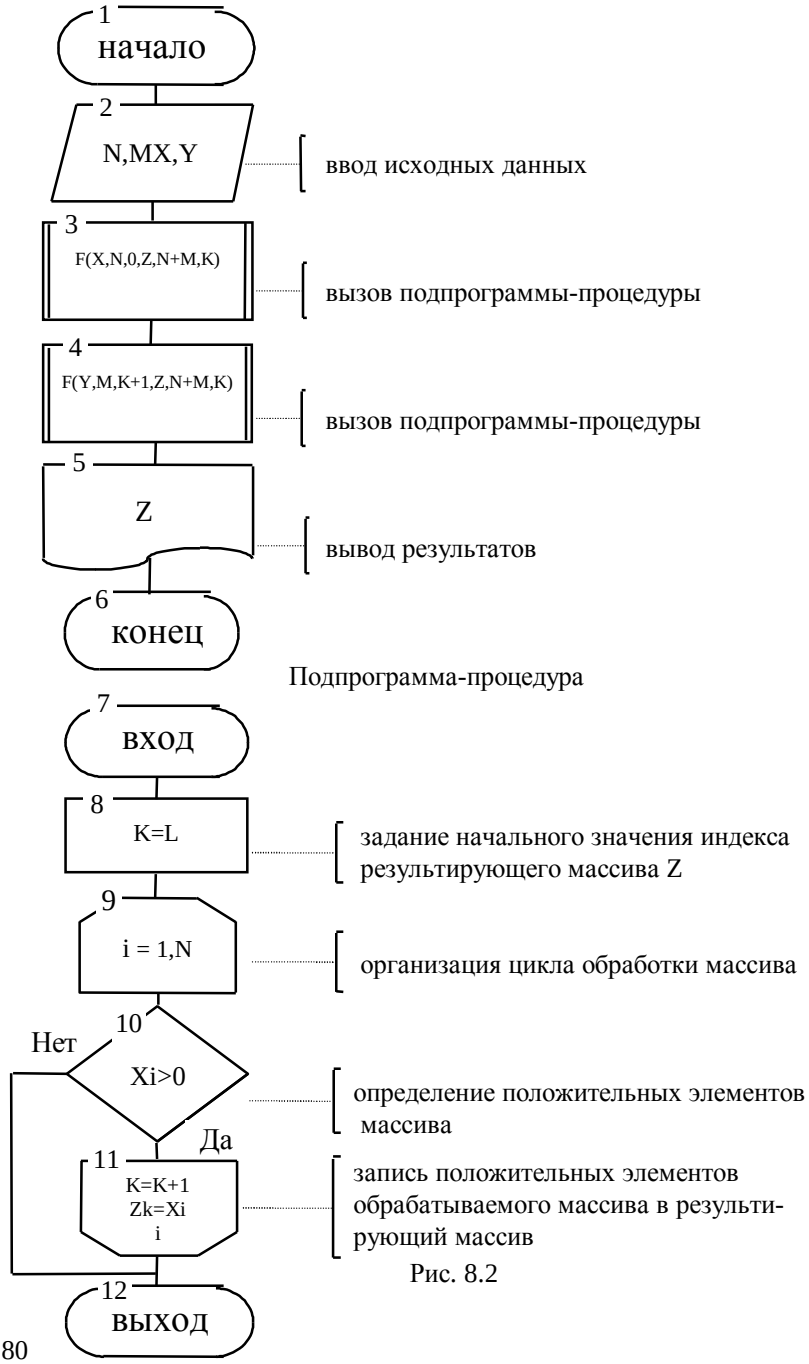

1
начало
2
N,MX,Y
ввод исходных данных
3
F(X,N,0,Z,N+M,K)
вызов подпрограммы-процедуры
4
F(Y,M,K+1,Z,N+M,K)
вызов подпрограммы-процедуры
5
Z
вывод результатов
6
конец
Подпрограмма-процедура
7
вход
8
K=L
задание начального значения индекса результирующего массива Z
9
i = 1,N
организация цикла обработки массива
10
Нет
Xi>0
определение положительных элементов массива
Да
11
K=K+1
Zk=Xi
i
запись положительных элементов обрабатываемого массива в результирующий массив
12
выход

Рис. 8.2

Программа, реализующая схему алгоритма (рис. 2) на языке программирования Pascal, имеет вид:

```
Program PRIM8B;
{Лабораторная работа № 8В}
{Студент группы ******* Иванов И.И.}
Const N=200;
Type
    Mas=array [1...N] of real;
Var
    i, NR, MR: integer;
    X, Y, Z: Mas;
Procedure ST (Var A,C:Mas; N,L,K:integer); {Подпрограмма-
                                            процедура}
    Var j:integer;
    Begin
        K:=L;  {Задание начального значения K}
        for j:=1 to N do  {Организация цикла}
          if A[j]>0 then
              begin
                 K:=K+1;
                 C[K]:=A[j]
              end;
    end;
Begin
    writeln('Введите значения NR и MR');
    read(NR,MR);  {Ввод размерностей массивов}
    writeln('Введите значения массива X');
    for i:=1 to NR do read(X[i]);  {Ввод координат точек в верхней
                                    полуплоскости}
    writeln('Введите значения массива Y');
    for i:=1 to MR do read(Y[i]);  {Ввод координат точек в нижней
                                    полуплоскости}
    ST(X,NR,0,Z,NR+MR,K);
    ST(Y,MR,K,Z,NR+MR,K);
    for i:=1 to K do
        write(Z[i])   {Вывод результатов}
End.
```

Данная конкретная программа в тестировании не нуждается, так как выведенные на печать результаты позволяют однозначно судить о правильности выполнения программы.

Лабораторная работа N 9
"Программирование с использованием модуля GRAPH для отображения графической информации на экране дисплея"

Цель работы - овладение методом построения графиков различных функций и практическими навыками работы в графическом режиме с использованием модуля Graph.

Задание для самостоятельной подготовки

1. Изучить возможности языка программирования для построе-ния графиков функций на экране дисплея с использованием модуля Graph;
2. Разработать алгоритм решения в соответствии с заданием.
3. Составить программу решения задачи.
4. Подготовить тестовый набор данных для контроля правиль-ности вычисления значений функции.

Задание к работе

Вывести на экран дисплея график функции $y=f(x)$, приведенной в таблице, в заданном диапазоне изменения аргумента x от a до b. Ширину поля графика и расположение его относительно края экрана задать самостоятельно. Предусмотреть после вывода графика на экран печать в виде таблицы вычисленных значений функции y с числом точек c, а также печать наибольшего и наименьшего значений функции.

Таблица

Вар. задания	Вид функции $y=f(x)$	Диапазон изменения аргумента		Число точек графика
		a	b	c
1	2	3	4	5
1	$\sin x+1$	$-\pi/2$	$+\pi/2$	30
2	$\cos x$	0	$3\pi/2$	40
3	$\lvert\sin x\rvert+\lvert\cos x\rvert$	0	π	40
4	$\lvert\sin x\rvert-\lvert\cos x\rvert$	0	π	40

5	$2\sin x+3\cos x$	$-\pi$	$+\pi$	50

Продолжение таблицы

1	2	3	4	5
6	$\sin x+\cos(2x)$	$-\pi$	$+\pi$	50
7	$2-\cos x$	0	$3\pi/2$	40
8	$\sin(\sqrt{2x})+\cos x$	0	2π	50
9	$2\sin(2x)+1$	$-\pi/2$	$+\pi/2$	50
10	$\sin x+\cos x-1$	$-\pi$	$+\pi$	40
11	$\sqrt{x^2+2}$	-3	5	40
12	$10/(1+x^2)$	-3	3	30
13	$(x-3)/(x^2+2)$	-1	4	50
14	$x\cos(2x)$	-1	4	50
15	$x^2 e^{-\lvert x\rvert}$	-1	3	40

Пример выполнения работы

Вывести на экран дисплея график функции $y=\sin(x)$ при изменении аргумента x от $a=-p$ до $b=p$. Число точек графика для вывода таблицы значений функции равно $c=40$.

Для нормальной работы модуля Graph необходимо выполнить следующие требования:

1) В разделе Uses подключить стандартную графическую библиотеку, например - Uses Graph. Предполагается, что в настройке оболочки Turbo Pascal маршрут к данному модулю указан верно.

2) Инициализировать режим графического отображения информации при помощи функции Initgraph, предварительно определив тип видеоадаптера.

По окончании работы с графической информацией функцией Closegraph деинициализируем режим графического вывода информации, т.е. перевести экран монитора в режим отображения алфавитно-цифровой информации.

3) При запуске программы в текущем каталоге должен находиться вспомогательный файл Egavga.bgi, или маршрут к нему должен быть указан.

Начнем написание тела программы с организации ввода исходных данных: a - начальное значение аргумента; b - конечное значение аргумента; c - количество точек графика для таблицы. Преобразуем переменные a и b из численного типа в строчный для последующего вывода на графический дисплей.

Определим тип видеоадаптера при помощи конструкции gd:=detect. Наиболее распространенным типом видеоадаптера в настоящее время является VGA - адаптер с параметрами 640 точек по горизонтали и 480 точек по вертикали.

При помощи функции Initgraph инициализируем режим графического вывода информации. Очистим графический экран функцией Cleardevice.

Вычислим вспомогательные коэффициенты my и mx для позиционирования осей X и Y, а также шаг изменения аргумента dx для вывода точек графика на экран дисплея, исходя из максимального количества отображаемых точек по горизонтали и вертикали (функции Getmaxx и Getmaxy).

Функцией Line вычертим ось X со стрелкой положительного направления. Выведем численное значение левой и правой границы графика (функция Outtextxy), используя ранее преобразованные в строчный тип переменные. Определим положение оси Y в зависимости от значений переменных a и b:

слева, если $a>0$;
справа, если $b<0$;
промежуточное, если $a<=0$ и $b>=0$.

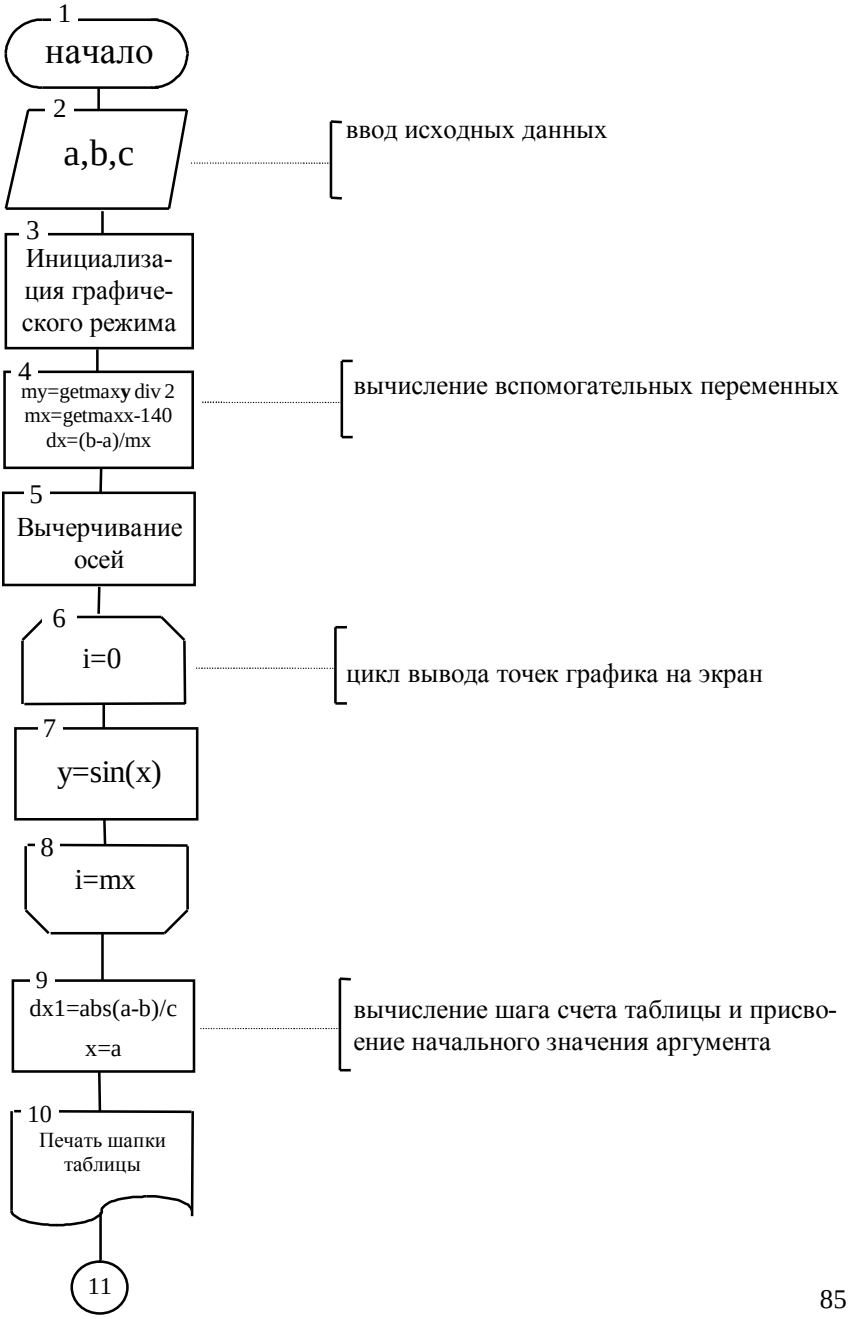
Организуем цикл вывода точек графика на графический экран монитора. Для чего вычислим Y - координаты (значение функции) графика при помощи формулы $y = \sin(i*dx+a)$, где i - номер текущей точки (изменяется от 0 до mx), dx - ранее вычисленное приращение аргумента, а - начальное значение аргумента. Выведем точку графика на экран при помощи функции Putpixel.

За циклом вывода точек графика на экран графического дисплея поставим конструкцию ch:=Readkey (ожидание нажатия любой клавиши) для просмотра графика.

Функцией closegraph деинициализируем режим графического вывода информации.

Организация вывода табличных значений функции рассмотрен в предыдущих работах и сложности не представляет.

Схема алгоритма решения задачи представлена на рисунке.


1
начало
2
a,b,c
ввод исходных данных
3
Инициализа-
ция графиче-
ского режима
4
my=getmaxy div 2
mx=getmaxx-140
dx=(b-a)/mx
вычисление вспомогательных переменных
5
Вычерчивание
осей
6
i=0
цикл вывода точек графика на экран
7
y=sin(x)
8
i=mx
9
dx1=abs(a-b)/c
x=a
вычисление шага счета таблицы и присво-
ение начального значения аргумента
10
Печать шапки
таблицы
11

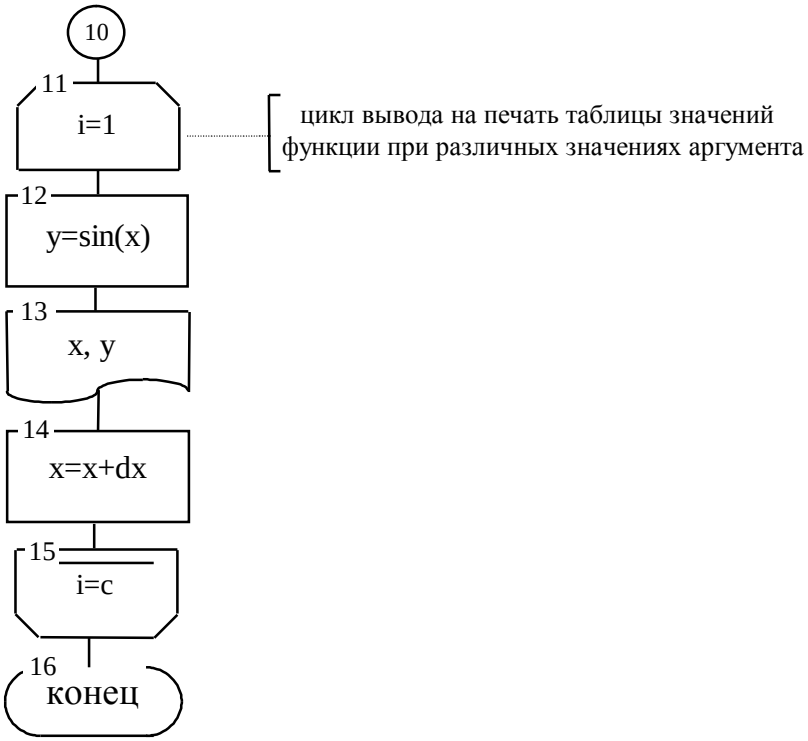


Рис. 9.1

Программа, реализующая алгоритм на языке программирования Pascal, имеет вид:

```
Program PRIM9;
{Лабораторная работа № 9}
{Студент группы ******* Иванов И.И.}

Uses Crt, Graph;

Var gd, gr, i, my, mx, c:integer;
    ch:char;
    dx, dx1, a, b, x, y:real;
    s1, s2:string;
```

```
      d:word;
Begin
Clrscr;   {Очистить экран в режиме работы модуля CRT}

          {Ввод исходных данных}

Write('Введите начальное значение аргумента a = ');
Readln(a);
Write('Введите конечное значение аргумента  b = ');
Readln(b);
Write('Введите количество точек графика для таблицы c = ');
readln(c);

       {Преобразование численного значения переменных в строковое
            для возможности вывода на экран в режиме модуля Graph}

Str(a:4:1, s1);
Str(b:4:1, s2);

gd:=detect;              {Определить тип видеоадаптера }
Initgraph(gd, gr, ' ');  {Инициализировать режим графического
                                 вывода информации}
Cleardevice;             {Очистить графический экран в режиме работы
                                 модуля Graph}

my:=getmaxy div 2;    {Вычислить вспомогательные переменные}
mx:=getmaxx-140;
dx:=(b-a)/mx;

Line(30, my, 590, my);            {Вычертить ось X}
Line(585, my-5, 590, my);
Line(590, my, 585, my+5);

Outtextxy(65, my+10, s1);         {Вывести численное значение левой
Outtextxy(getmaxx-75, my+10, s2);       и правой границы графика}

if a>0 then       {Определить положение оси Y слева, если a>0}
  begin
    Line(50, 30, 50, getmaxy-30);
    Line(45, 35, 50, 30);
    Line(55, 35, 50, 30)
  end;
```

```
if b<0 then   {Определить положение оси Y справа, если b<0}
  begin
    Line(580, 30, 580, getmaxy-30);
    Line(575, 35, 580, 30);
    Line(585, 35, 580, 30)
  end;
if (a<=0) and (b>=0) then       {Определить положение оси Y
                                автоматически, если a<=0 и b>=0}
  begin
    d:=round(70-a/dx);
    Line(round(70-a/dx), 30, round(70-a/dx), getmaxy-30);
    Line(round(70-a/dx)-5, 35, round(70-a/dx), 30);
    Line(d, 30, d+5, 35)
  end;
for i:=0 to mx do       {Цикл вывода точек графика на экран}
  begin
    y:=sin(i*dx+a);              {Вычисление значения функции}
    Putpixel(70+i, round(my-100*y), 5)    {Вывод точки графика на
                                           экран}
  end;
ch:=readkey;      {Ожидание нажатия любой клавиши}
Closegraph;       {Деинициализировать режим графического
                              вывода информации}

              {Печать таблицы значений функции}

dx1:=abs(a-b)/c;    {Вычисление шага счета таблицы}
x:=a;       {Присвоение начального значения аргумента}

              {Вывод шапки таблицы}

Writeln(' N    Значение     Значение ');
Writeln(' п/п   аргумента    функции  ');
Writeln('          X           Y     ');
Writeln('-----------------------------');

for i:=1 to c do     {Цикл вывода значений таблицы}
  begin
    y:=sin(x);
    Writeln(' ',i:3,'    ',X:7:3,'   ',Y:7:3);
    x:=x+dx1
```

```
  end;
Writeln('-----------------------------');
ch:=readkey {Ожидание нажатия любой клавиши}
End.
```

Лабораторная работа № 10
“Обработка файловых структур данных”

Цель работы - овладение навыками алгоритмизации и программирования файловых структур данных; проектирование структуры файла, вывод данных в файл, чтение данных из файла, а также получение практических навыков работы в текстовом режиме с использованием модуля CRT.

Задание для самостоятельной подготовки

1. Изучить:

- основную терминологию, связанную с файловыми структурами данных: файл и его структура, методы доступа, типизированный файл, запись конца файла для файлов с последовательным доступом;
- возможности языка программирования по обработке файла с последовательной организацией: запись данных в файл, чтение из файла, добавление записей в файл, корректировка записей и т.п.;
- основные стандартные процедуры модуля CRT, позволяющие управлять выводом на экран текстовой информации.

2. Разработать алгоритм решения в соответствии с заданием.
3. Составить программу решения задачи.
4. Подготовить тестовый вариант программы и исходных данных.

Задание к работе

Ввести в ЭВМ программу, вызова на экран дисплея меню (горизонтального либо вертикального), с помощью которого выполнится задание А или Б. Выбор конкретного режима этого меню осуществляется с помощью клавиш «↑», «↓», либо «←», «→». При нажатии этих клавиш строка, на которую падает выбор, должна выделяться цветом и фоном. Выбор фиксируется с помощью клавиши Enter. Задания А и Б оформляются в виде подпрограмм-процедур.

Задание А.

Написать подпрограмму создания файла в соответствии с вариантом задания, указанным в табл. 1(А). Прочитать и напечатать созданный файл.

Таблица 1

Вар. Задания	Условие задачи
1	А. Создать файл, содержащий сведения о месячной заработной плате рабочих завода. Каждая запись содержит поля - фамилия рабочего, наименование цеха, размер заработной платы за месяц. Количество записей - произвольное. Б. Вычислить общую сумму выплат за месяц по цеху X, а также среднемесячный заработок рабочего этого цеха. Напечатать для бухгалтерии ведомость для начисления заработной платы рабочим этого цеха.
2	А. Создать файл, содержащий сведения о количестве изделий, собранных сборщиками цеха за неделю. Каждая запись содержит поля: фамилия сборщика, количество изделий, собранных им ежедневно в течение шестидневной недели, т.е. раздельно - в Понедельник, Вторник и т.д. Количество записей - произвольное. Б. Написать программу, выдающую на печать следующую информацию: фамилию сборщика и общее количество деталей, собранное им за неделю; фамилию сборщика, собравшего наибольшее число изделий, и день, когда он достиг наивысшей производительности труда.
3	А. Создать файл, содержащий сведения о количестве изделий категорий A, B, C, собранных рабочим за месяц. Структура записи имеет поля: фамилия сборщика, наименование цеха, количество изделий по категориям, собранных рабочим за месяц. Количество записей - произвольное. Б. Считая заданными значения расценок S_A, S_B, S_C за выпол-ненную работу по сборке единицы изделия категорий A, B, C соответственно, выдать на печать следующую информацию: - общее количество изделий категорий A, B, C, собранных рабочим цеха X; - ведомость заработной платы рабочих цеха X; - средний размер заработной платы работников цеха X.
4	А. Создать файл, содержащий сведения о телефонах абонентов. Каждая запись имеет поля: фамилия абонента, год установки телефона, номер телефона. Количество записей - произвольное. Б. Написать программу, выдающую информацию следующего вида: - по вводимой фамилии абонента выдается номер телефона; - определяется количество установленных телефонов с XXXX года. Номер года вводится с терминала.

Продолжение табл. 1

5	А.	Создать файл, содержащий сведения об ассортименте игрушек в магазине. Структура записи: название игрушки, цена, количество, возрастные границы, например 2-5, т.е. от 2 до 5 лет. Количество записей - произвольное.
	Б.	Написать программу, в результате которой выдаются следующие сведения: - названия игрушек, которые подходят детям от 1 до 3 лет; - стоимость самой дорогой игрушки и ее наименование; - название игрушки, которая по стоимости не превышает x руб. и подходит ребенку в возрасте от a до b лет. Значения x, a, b ввести с терминала.
6	А.	Создать файл, содержащий сведения о сдаче студентами 1 курса сессии. Структура записи: индекс группы, фамилия студента, оценки по пяти экзаменам, признак участия в общественной работе: “1” - активное участие, “0” - неучастие. Количество записей - 30.
	Б.	Написать программу зачисления студентов группы X на стипендию. Студент, получивший все оценки “5” и активно участвующий в общественной работе, зачисляется на повышенную стипендию (доплата 50%), не активно участвует - доплата 25%. Студенты, получившие “4” и “5”, зачисляются на обычную стипендию. Студент, получивший одну оценку “3”, но активно занимающийся общественной работой, также зачисляется на стипендию, в противном случае зачисление не производится. Индекс группы вводится с терминала.
7	А.	Создать файл, содержащий сведения о сдаче студентами сессии. Структура записи: индекс группы, фамилия студента, оценки по пяти экзаменам и пяти зачетам (“з” означает зачет, “н” - незачет). Количество записей - 25.
	Б.	Написать программу, выдающую следующую информацию: - фамилии неуспевающих студентов с указанием индексов групп и количества задолженностей; - средний балл, полученный каждым студентом группы X, и всей группы в целом.
8	А.	Создать файл, содержащий сведения о личной коллекции книголюба. Структура записи: шифр книги, автор, название, год издания, местоположение (номер стеллажа, шкафа и т.п.). Количество записей - произвольное.
	Б.	Написать программу, выдающую следующую информацию: - местонахождение книги автора X названия Y. Значения X и Y ввести с терминала; - список книг автора Z, находящихся в коллекции; - число книг издания XX года, имеющееся в библиотеке.

Продолжение табл. 1

9	А.	Создать файл, содержащий сведения о наличии билетов и рейсах Белавиа. Структура записи: номер рейса, пункт назначения, время вылета, время прибытия, количество свободных мест в салоне. Количество записей - произвольное.
	Б.	Написать программу, выдающую информацию следующего вида: - время отправления самолетов в город X; - наличие свободных мест на рейс в город X с временем отправления Y. Значения X и Y вводятся по запросу с терминала.
10	А.	Создать файл, содержащий сведения об ассортименте обуви в магазине фирмы. Структура записи: артикул, наименование, количество, стоимость одной пары. Количество записей - произвольное. Артикул начинается с буквы Д для дамской обуви, М для мужской, П для детской.
	Б.	Написать программу, выдающую следующую информацию: - о наличии и стоимости обуви артикула X; - ассортиментный список дамской обуви с указанием наименования и имеющегося в наличии числа пар каждой модели.
11	А.	Создать два файла, содержащих сведения о десяти нападающих хоккейных команд “Динамо” и “Спартак” соответственно: имена нападающих, число заброшенных ими шайб, сделанных голевых передач, заработанное штрафное время.
	Б.	Написать программу, которая по данным, извлеченным из этих файлов, создает новый третий файл, содержащий имя, команду, сумму очков (голы+передачи) для шести лучших игроков обеих команд. Имена и показатели результативности хоккеистов вывести на экран.
12	А.	Создать файл, содержащий сведения о том, какие из пяти предлагаемых дисциплин по выбору желает слушать студент. Структура записи: фамилия студента, индекс группы, 5 дисциплин, средний балл успеваемости. Выбираемая дисциплина отмечается символом 1, иначе - пробел. Количество записей - 25.
	Б.	Написать программу, которая печатает список студентов, желающих слушать дисциплину X. Если число желающих превысит 8 человек, то отобрать студентов, имеющих более высокий средний балл успеваемости.
13	А.	Создать файл, содержащий сведения об отправлении поездов дальнего следования с Минского вокзала. Структура записи: номер поезда, станция назначения, время отправления, время в пути, наличие билетов. Количество - записей произвольное.
	Б.	Написать программу, которая позволяет получить следующую справочную информацию: - время отправления поездов в город X во временном интервале от A до B часов; - наличие билетов на поезд с номером XXX.

Продолжение табл. 1

Продолжение табл. 1

14	А.	Создать файл, содержащий сведения о сотрудниках института. Структура записи: фамилия работающего, название отдела, год рождения, стаж работы, должность, оклад. Количество записей - произвольное.
	Б.	Написать программу, которая позволяет получить следующую информацию: - список сотрудников пенсионного возраста на сегодняшний день с указанием стажа работы; - средний стаж работающих в отделе X.
15	А.	Создать файл, содержащий сведения о пациентах глазной клиники. Структура записи: фамилия пациента, пол, возраст, место проживания (город), диагноз. Количество записей - произвольное.
	Б.	Написать программу, выдающую следующую информацию: - количество иногородних, прибывших в клинику; - список пациентов старше X лет с диагнозом Y. Значения X и Y ввести с терминала.

Задание Б.

Написать процедуру обработки файла, созданного в задании А, в соответствии с заданием, указанным в табл.1(Б) для заданного варианта. Проверить правильность выполнения обеих программ с помощью тестового варианта исходных данных.

Пример выполнения работы

Выполнить на ЭВМ программу вывода на экран дисплея меню следующего вида:

Выберите режим
1. Создание файла 2. Обработка файловой информации 3. Окончание работы

Для вывода на экран изображения меню выделяется соответствующая типизированная константа-массив Stor. Каждый элемент этого массива содержит соответствующую строку из меню. В начале работы устанавливается текстовый режим работы с размером экрана 80×25. Затем экран очищается и определяется окно размером 6×38. Задается цвет и фон выводимых символов. Переменная W принимает значение, равное номеру выбранного режима (предполагается, что W вначале равно 1). Затем выводится

заглавие меню с приглашением выбрать нужный режим. Индикация строки, соответствующей выбранному из меню варианту, обеспечивается в цикле for..., в котором при условии T=W (где T - управляющая переменная цикла) изменяется цвет и фон выводимых символов. Затем ожидается нажатие клавиш «↑» и «↓» на клавиатуре. Если будет нажата клавиша «↑» и выделенной на этот момент будет первая строка, то произойдет переход к последней строке из меню и т.п. Как только будет нажата клавиша Enter, произойдет очистка экрана и выполнение программы перейдет к соответствующей процедуре.

Программа создания меню имеет вид:

```
Program PRIM10;
{Лабораторная работа № 10}
{Студент группы  *******  Иванов И.И.}
Uses Crt;  {Подсоединение модуля Crt}
Label M1,M2;
Type
     Zap=record
          Index:string[6];
          Fam:string[20];
          Marker: array [1..5] of byte
     end;
     Mas=array[1..3] of string[36];
Const   {Описание массива с изображением меню}
     Stor:Mas=('  1.  Создание файла              ',
               '  2.  Обработка файловой информации ',
               '  3.  Окончание работы            ');
Var
     Sessya:File of Zap;  {Объявление файловой переменной}
     X:Zap;
     w, t:byte;
     Kod:char;
Procedure Sozdan;  {Процедура создания файла Sessya (задание А)}
     Begin
          ...
     End;
Procedure Obrabotka;  {Процедура обработки файловой информации
                            (задание Б)}
     Begin
          ...
     End;
```

```
Begin  {Начало головной программы}
    TextMode(CO80);  {Установка текстового режима 80×25}
M2: ClrScr;  {Очистка экрана}
 Window(22,11,59,17);  {Организация окна меню}
 TextBackGround(14);  {Задание фона (14 - желтый)}
 TextColor(0);  {Задание цвета выводимых символов (0 - черный}
 ClrScr;  {Очистка установленного окна}
 w:=1;  {W - номер режима}
 GotoXY(11,2);
 Write('Выберите режим');  {Вывод заглавия меню}
{Выделение выбранной строки фоном и цветом}
 Kod:=' ';
 while Kod<>#13 do   begin  {Цикл выполняется до тех пор, пока не
                                нажата клавиша Enter}
     for t:=1 to 3 do
     begin
         if t=w then
          begin
              TextBackGround(1);  {Задание фона выбранной строки
                                      меню (1 - синий)}
              TextColor(15);  {Задание цвета символов выбранной
                                строки (15 - белый)}
          end
         else
          begin
              TextBackGround(14);
              TextColor(0);
          end;
       GotoXY(2,t+3);
       Write(Stor[t]);  {Вывод строк меню}
     end;
{Отображение выбора с помощью клавиш «↑» и «↓»}
     Kod:=ReadKey;  {Считывание символа}
     if Kod=#0 then  begin  {Если нажата функциональная клавиша}
         Kod:=ReadKey;  {Считывание второго байта}
         if Kod=#72 then
          begin  {Если нажата клавиша «↑»}
              if w>1 then w:=w-1
              else w:=3
```

```
            end;
          if Kod=#80 then
            begin  {Если нажата клавиша «↓» }
              if w<3 then w:=w+1
              else w:=1
            end
      end
  end;
  Window(1,1,80,25);  {Переход к полному экрану}
  TextBackGround(0);  {Восстановление фона (0 - черный)}
  TextColor(15);  {Восстановление цвета выводимых символов
                          (15 - белый)}
  ClrScr;
      if w=1 then Sozdan;  {Выполнение процедуры создания файла}
      if w=2 then Obrabotka;  {Выполнение процедуры обработки
                                файловой информации}
      if w=3 then Goto M1;
  Goto M2;
M1: End.
```

Задание А.

Выполнить на ЭВМ подпрограмму создания файла, содержащего сведения о сдаче студентами сессии. Структура записи содержит поля: индекс группы, фамилию студента, оценки по пяти экзаменам. Количество записей в файле произвольное. Подпрограмму оформить в виде процедуры и ввести в головную программу создания меню PRIM10. Идентификаторы и длины полей приведены в табл.2.

Таблица 2

Описание поля	Имя переменной	Тип данных	Пример
Индекс группы	INDEX	6 символов	101317
Фамилия	FAM	20 символов	Иванов П.В.
Оценки по пяти экзаменам	Массив MARKER (5)	Каждый элемент типа byte	54323

Для получения текущей записи организуем в процедуре запрос на ввод очередной порции информации с терминала в оперативную память ЭВМ, для чего воспользуемся обычными операторами языка для ввода данных. Полученную строку данных запишем в первую запись файла. Для этого используем оператор записи данных в файл, после чего запросим ввод второй строки данных с терминала в оперативную память ЭВМ. Организуем ее запись в файл. Этот процесс ввода с терминала и записи в файл будем про-

должать до тех пор, пока не будет получен с терминала признак окончания ввода данных. Для подсчета количества записей в файле введем счетчик К, значение которого будем увеличивать на 1 при каждой операции записи строки в файл. Для контроля после создания файла организуем чтение записей файла и вывод их на экран.

Процедура создания файла с именем SESSYA имеет вид:

```
Procedure Sozdan;
Var
  K:integer;
  Flag:byte;

Begin
  K:=0;
  Flag:=0;
  Assing(Sessya,'D:\Sessya.dat); {Связывание файловой перемен-
                ной Sessya с файлом Sessya.dat на диске D}
  Rewrite(Sessya); {Открыть файл Sessya для записи данных}
      Writeln('Введите текущую запись: индекс группы,);
      Writeln('фамилию и 5 оценок за экзамены');
      Writeln('Для окончания работы введите вместо');
      Writeln('индекса группы символ ######');
  While Flag=0 do  {Цикл ввода одной записи с экрана и записи ее в
                                файл}
  begin
       Read(X.Index);
           if X.Index='######' then
             Flag:=1  {Признак окончания ввода данных в файл Sessya}
           else
             begin
               Read(X.Fam); {Ввод полей записи с терминала}
               Read(X.Marker[1]);
               Read(X.Marker[2]);
               Read(X.Marker[3]);
               Read(X.Marker[4]);
               Read(X.Marker[5]);
               K:=K+1; Flag:=0;
               Write(Sessya,X) {Запись переменной X в файл Sessya}
               Writeln('Введите индекс группы')
             end;
  end; {Конец цикла}
  Writeln('В файле ',K ,' записей');
  readln;  {Ожидание нажатия клавиши Enter для выхода из режима
```

```
                просмотра пользовательского экрана}
  Close(Sessya); {Закрыть файл Sessya}
  Reset(Sessya); {Открыть файл Sessya для чтения}
     While not EOF(Sessya) do {Выполнять цикл чтения записей
        файла до тех пор, пока не будет обнаружен конец файла}
      begin
         Read(Sessya,X); {Считывание записи файла в переменную X}
             {Вывод переменной X типа "запись" на экран}
         Writeln('| ',X.Index,' | ',X.Fam,' | ',X.Marker[1],' ',
         X.Marker[2],' ',X.Marker[3],' ',X.Marker[4],' ',X.Marker[5])
      end;
  Writeln('Конец файла'); readln;
  Close(Sessya); {Закрыть файл Sessya}
End.
```

Тестовый вариант исходных данных для создания и обработки файла

101317	Сидоров В.Г.	44323
101317	Никифоров П.Д.	55343
101357	Федоров С.Г.	44444
101357	Фокин П.А.	55555
101317	Подгорный А.А.	55232
101317	Силаев Ф.Г.	55555
101317	Белов И.Ф.	55455
101317	Анисов Д.Ф.	42323
101317	Лукин В.Д.	33233
101317	Дедов П.Л.	33333

В качестве индекса группы Y ввести значение 101317.

Задание Б.

Написать подпрограмму зачисления на стипендию студентов группы Y. Размер обычной стипендии 400 тыс.руб. Студенту, получившему все пять оценок "5", назначается стипендия, повышенная на 50%; получившему оценки "4" и "5" - повышенная на 25%; студенту, получившему хотя бы одну оценку "2", стипендия не назначается. В остальных случаях назначается обычный размер стипендии. Подпрограмму оформить в виде процедуры и ввести в головную программу создания меню PRIM10.

Для выполнения задания Б необходимо: ввести с терминала индекс группы Y, для которой печатается ведомость зачисления на стипендию; организовать в цикле чтение текущей записи файла, проверять совпадение значения поля "индекс группы" со значением переменной Y. Если индексы не совпадают, то переходить к чтению следующей записи файла. Если индексы совпадают, то проводить зачисление на стипендию по следующему алгоритму: ввести счетчики количества отличных и хороших оценок K5 и K4, задав их начальное значение 0; организовать цикл на пять повторений

для проверки каждой оценки студента. Если текущая оценка “2”, то выходить из цикла, зачисления на стипендию не будет и переходить к чтению следующей записи файла; если оценка “5”, то увеличить счетчик К5 на 1 и продолжить цикл просмотра оценок; если оценка “не 5”, то проверить ее на значение “4”. Если оценка “4”, то счетчик К4 увеличить на 1 и продолжить цикл. После просмотра всех оценок проверить значение счетчика К5 - равно ли пяти? При положительном ответе назначить стипендию 400+400/2=600 тыс.руб. Выдать на печать фамилию студента, размер стипендии и перейти к чтению следующей записи; если К5 не равно пяти, то проверить равенство пяти суммы значений К5 и К4. При ответе “Да” назначить стипендию 400+400/4=500 тыс.руб., в противном случае - стипендию 400 тыс.руб. и перейти к печати строки документа о зачислении на стипендию, а затем - к чтению новой записи. При обнаружении конца файла завершить решение задачи. Процедура имеет вид:

```
Procedure Obrabotka;
Lable 2;
Var
      Y:string[6];
      i,K4,K5,N,j: integer;
      S:real;
Begin
  Assing(Sessya,’D:\Sessya.dat); {Связывание файловой перемен-
                ной Sessya с файлом Sessya.dat на диске D}
  Reset(Sessya); {Открыть файл Sessya для чтения}
  Writeln(‘Введите индекс группы ‘);
  Readln(Y);
  Writeln(‘Ведомость зачисления на стипендию студентов’);
  Writeln(‘                    группы ‘,Y);
  Writeln(‘|------|----------------------------------|--------------|‘);
  Writeln(‘| п/п |       Фамилия И.О.              |  (тыс.руб.) |‘);
  Writeln(‘|------|----------------------------------|--------------|‘);
  N:=0;  j:=0;
  While not EOF(Sessya) do  {Цикл обработки файловой информации
                              при отсутствии конца файла}
      begin
        Read(Sessya,X);     {Чтение из фала в переменную типа запись}
        if  X.Index=Y  then  begin   {Проверка индекса группы}
                  K4:=0;   K5:=0;
                  For i:=1 to 5 do  begin  {Цикл проверки оценок}
                        if X.Marker[i]=5  then  K5:=K5+1;
                        if X.Marker[i]=4  then  K4:=K4+1;
                        if X.Marker[i]=2  then  j:=1
                    end;
                  if j<>1 then      begin
                     S:=400;     {Назначение стипендии троечникам}
```

```
                    N:=N+1;
                      if K5=5 then
                       S:=600  {Назначение стипендии отличникам}
                      else
                       if (K5+K4)=5 then  S:=500;  {Назначение
                                          стипендии хорошистам}
  {Вывод на печать списка студентов, которым начислена стипендия}
                      Writeln(' | ',N:2,' |    ',X.Fam,'    | ',S:7:3,' |')
                   end;
            end;
       end;
  Writeln('Конец файла'); Readln; Close(Sessya) {Закрыть файл}
End.
```

Схема алгоритма всей программы создания меню и выполнения подпрограмм-процедур приведена на рисунке.

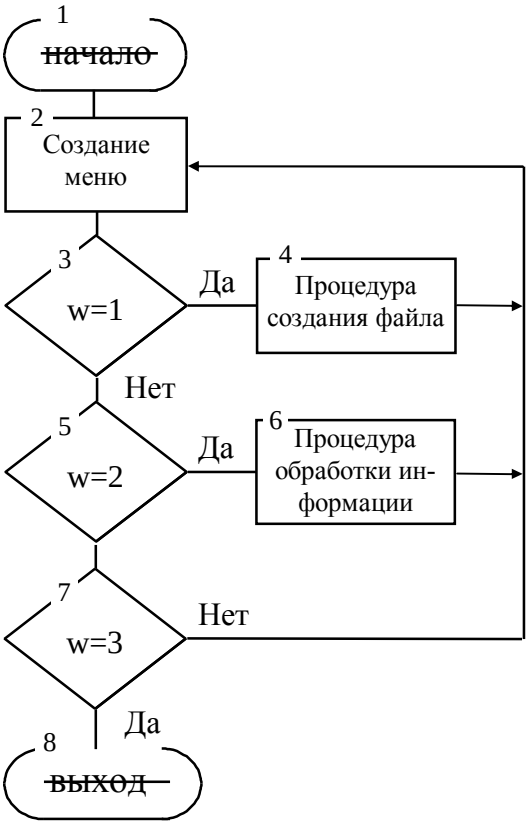


Рис. 10.1

ПРИЛОЖЕНИЕ 1

Список горячих клавиш Norton Commander

Tab - перейти на другую панель.

Enter - переход в другой каталог. Надо выделить этот каталог и нажать клавишу Enter.

Ctrl- - переход в корневой каталог.

Ctrl\PgUp - переход в надкаталог.

Ins - включить файл в группу, исключить файл из группы.

(+) - включить в группу файлы по маске. Нажать + на функциональной клавиатуре и ввести маску.

(-) - исключить из группы файлы по маске. Нажать - на функциональной клавиатуре и ввести маску.

Выбранные файлы изображаются желтым цветом на цветном и повышенной яркостью на монохромном дисплее.

Ctrl-O - убрать панели с экрана или вывести их на экран.

Ctrl-P - убрать одну из панелей (не текущую) с экрана или вывести панель на экран.

Ctrl-U - поменять панели местами.

Alt-F1 - вывести в левой панели оглавление другого диска.

Alt-F2 - вывести в правой панели оглавление другого диска.

Ctrl-F1 - убрать левую панель с экрана / вывести левую панель на экран.

Ctrl-F2 - убрать правую панель с экрана / вывести правую па-нель на экран.

Shift-F3 - (View) - просмотр файла. Имя файла запрашивается.

Shift-F4 - (Edit) - редактирование файла. Имя файла запрашивается.

Shift-F5 - (Copy) - копирование файла или группы файлов. Запрашивается, что и куда надо копировать.

Shift-F6 - (Renmov) - переименование файла (файлов) или каталога, пересылка файла (файлов) в другой каталог. Запрашивается, какие файлы и как (куда) переименовывать или пересылать.

Shift-F9 - сохранение текущих режимов Norton Commander.

Alt-F3 - (View) - просмотр файла с помощью встроенной программы просмотра Norton Commander.

Alt-F4 - (Edit) - редактирование файла с помощью альтернативного редактора (если F4 соответствует встроенный редактор Norton Commander, то Alt-F4 - внешний редактор, и наоборот).

Alt-F7 - (Search) - поиск файла на диске.

Alt-F8 - (History) - просмотр и повторное выполнение ранее введенных команд.

Alt-F9 - (EgaLn) - переключение с 25 на 43 строки на экране.

Alt-F10 - (Tree) - быстрый переход в другой каталог.

ПРИЛОЖЕНИЕ 2

Сообщения об ошибках на шаге компиляции языка Turbo Pascal.

Ниже приводятся коды ошибок и сообщения об ошибках, генерируемые компилятором языка Turbo Pascal. Кроме перевода сообщений в некоторых случаях даются необходимые пояснения, а также рекомендации по устранению ошибок.

1. **Out of memory** (выход за границы памяти). Компилятор извещает, что доступной памяти недостаточно для размещения программы. Чтобы устранить эту ошибку, рекомендуется удалить из памяти ранее загруженные, но не используемые в данный момент программы, или указать, что объектный код должен выводиться на диск. Если это не дает результата, следует разделить программу или модуль на большее число модулей.

2. **Identifier expected** (ожидается идентификатор).

Возможно, в качестве идентификатора использовано зарезервированное слово.

3. **Unknown identifier** (неизвестный идентификатор).

Идентификатор не объявлен.

4. **Duplicate identifier** (повторный идентификатор).

Идентификатор объявлен дважды.

5. **Syntax error** (синтаксическая ошибка).

Обнаружен символ, отсутствующий в алфавите языка.

6. **Error in real constant** (ошибка в записи константы вещественного типа).

7. **Error in integer constant** (ошибка в записи константы целого типа).

8. **String constant exceeds line** (длина строковой константы превышает максимально допустимую длину строки). Возможно ,отсутствует закрывающий апостроф.

9. **Too many nested files** (слишком много вложенных файлов: не более 15 уровней вложенности).

10. **Unexpected end of file** (несвоевременное появление признака конца файла). Возможно, не совпадает количество операторов begin и end, или не закрыт комментарий.

11. **Line too long** (слишком длинная строка). Длина строки превысила 126 символов.

12. **Type identifier expected** (ожидается идентификатор типа).

13. **Too many open files** (слишком много открытых файлов). Следует задать большее число файлов в CONFIG.SYS (в записи FILES = <число>).

14. **Invalid file name** (неправильно задано имя файла).

15. **File not found** (файл не найден).

16. **Disk full** (на диске нет свободного места).

17. **Invalid compiler directive** (неправильно записана директива компилятора).

18. **Too many files** (слишком много файлов).

19. **Undefined type in pointer definition** (необъявленный тип в объявлении указателя).

20. **Variable identifier expected** (ожидается идентификатор переменной).

21. **Error in type** (ошибка в объявлении типа).

22. **Structure too large** (структура слишком большая: допустимый размер 65520 байт области памяти для данных структурированного типа).

23. **Set base type out of range** (число значений базового типа для множества превышает допустимое: 256 значений).

24. **File components may not be files or objects** (компонентами файлов не могут быть файлы или объекты).

25. **Invalid string length** (недопустимая длина строки: свыше 255).

26. **Type mismatch** (несоответствие типов). Не соответствуют друг другу типы данных в выражении.

27.**Invalid subrange base type** (недопустимый базовый тип для интервального типа).

28. **Lower bound greater than upper bound** (нижняя граница больше верхней границы).

30. **Integer constant expected** (ожидается целая константа).

31. **Constant expected** (ожидается константа).

32. **Integer or real constant expected** (ожидается целая или вещественная константа).

33. **Type identifier expected** (ожидается идентификатор типа).

34. **Invalid function result type** (недопустимый тип результата функции).

35. **Label identifier expected** (ожидается идентификатор метки).

36. **Begin expected** (ожидается зарезервированное слово begin).

37. **End expected** (ожидается зарезервированное слово end).

38. **Integer expression expected** (ожидается выражение целого типа).

39. **Ordinal expression expected** (ожидается выражение порядкового типа).

40. **Boolean expression expected** (ожидается выражение булевого типа).

41. **Operand types do not match operator** (типы операндов не соответствуют оператору).

42. **Error in expression** (ошибка в выражении).

43. **Illegal assignment** (неправильное присваивание).

44. **Field identifer expected** (ожидается идентификатор поля записи).

45. **Object file too large** (объектный файл слишком большой: максимально допустимый размер 65520 байт).

46. **Undefined external** (не определена внешняя подпрограмма). Вероятно, во внешней подпрограмме отсутствует соответствующее определение PUBLIC.

47. **Invalid object file record** (нераспознаваемая запись объектного файла). Вероятно, объектный файл искажен.

48. **Code Segment too large** (кодовый сегмент слишком велик). Превышен максимально допустимый размер кода программы или модуля (65520 байтов).

49. **Data Segment too large** (сегмент данных слишком велик). Превышен максимально допустимый размер сегмента данных (65520 байтов).

50. **Do expected** (ожидается ключевое слово DO).

51. **Invalid PUBLIC definition** (неправильное определение public). Несоответствие определения **PUBLIC** в программе на Ассемблере и директивы external в программе или модуле на Паскале.

52. **Invalid EXTRN definition** (неправильное определение extrn). Вероятно, фрагмент программы на языке Ассемблер не объявлен в программе или модуле.

53. **Too many EXTRN definitions** (слишком много определений extrn). Максимально допустимое число определений EXTRN в Obj-файле равно 256.

54. **OF expected** (ожидается зарезервированное слово OF).

55. **INTERFACE expected** (ожидается зарезервированное слово interface).

56. **Invalid relocatable reference** (неправильно определена смещенная ссылка). Вероятно, во фрагменте программы на языке Ассемблера неправильно задано смещение адреса.

57. **THEN expected** (ожидается зарезервированное слово then).

58. **TO or DOWNTO expected** (ожидается зарезервированное слово to или downto).

59. **Undefined forward** (не завершено опережающее объявление).

60. **Too many procedures** (слишком много процедур: максимально допустимое количество 512).

61. **Invalid typecast** (неверно описанное преобразование типов).

62. **Division by zero** (деление на нуль).

63. **Invalid file type** (неправильно задан файловый тип).

64. **Cannot Read or Write variables of type** (нельзя читать или писать переменные этого типа). Предпринята попытка чтения/записи данных, не обрабатываемых процедурами Read/Readln и Write/Writeln.

65. **Pointer variable expected** (ожидается переменная ссылочного типа).

66. **String variable expected** (ожидается строковая переменная).

67. **String expression expected** (ожидается выражение строкового типа).

68. **Circular unit reference** (циклические ссылки модулей). Не допускается , чтобы два модуля ссылались друг на друга.

69. **Unit name mismatch** (неправильное имя модуля). Модуль, имя которого задано в директиве Uses, не найден.

70. **Unit version mismatch** (неверная версия модуля). Модуль, подключаемый к программе, был изменен после компиляции.

71. **Duplicate unit name** (имя модуля дублируется в директиве Uses).

72. **Unit file format error** (ошибка в спецификации файла модуля).

73. **Implementation expected** (ожидается зарезервированное слово implementation).

74. **Constant and case types do not match** (типы констант и селектора в операторе case не соответствуют друг другу).

75. **Record variable expected** (ожидается переменная типа “запись”).

76. **Constant out of range** (константа не укладывается в допустимый диапазон).

77. **File variable expected** (ожидается переменная файлового типа).

78. **Pointer expression expected** (ожидается выражение ссылочного типа).

79. **Integer or real expression expected** (ожидается выражение типа integer или real).

80. **Label not within current block** (метка находится вне текущего блока).

81. **Label already defined** (метка ранее уже объявлена).

82. **Undefined label in preceding statement part** (необъявленная метка в предыдущей части раздела операторов).

83. **Invalid @ argument** (неправильный аргумент оператора @).

84. **UNIT expected** (ожидается зарезервированное слово unit).

85. **“;” expected** (ожидается точка с запятой).

86. **“:“ expected** (ожидается двоеточие).

87. **“,“ expected** (ожидается запятая).

88. **“(“ expected** (ожидается “(“).

89. **“)” expected** (ожидается “)”).

90. **“=“ expected** (ожидается знак равенства).

91. **“:=“ expected** (ожидается знак присваивания).

92. **“[“ or “(.” expected** (ожидаются знаки “[“ или “(.”).

93. **“]” or “.)” expected** (ожидаются знаки “]” или “.)”).

94. **“.” expected** (ожидается точка).

95. **“..” expected** (ожидается “..”).

96. **Too many variables** (слишком много переменных).

Максимально допустимый размер памяти для размещения переменных равен 64 К:

97. **Invalid FOR control variable** (неправильная переменная цикла в операторе for).

98. **Integer variable expected** (ожидается переменная целого типа).

99. **File and procedure types are not allowed here** (в данном контексте файловый и процедурный типы недопустимы).

100. **String lengtch mismatch** (неправильная длина строки). Длина строковой константы не соответствует количеству элементов символьного массива.

101. **Invalid ordering of fields** (неправильный порядок следования полей).

102. **String constant expected** (ожидается константа строкового типа).

103. **Integer or real variable expected** (ожидается переменная типа integer или real).

104. **Ordinal variable expected** (ожидается переменная порядкового типа).

105. **INLINE error** (ошибка в директиве inline).

106. **Character expression expected** (ожидается выражение символьного типа).

107. **Too many relocation items** (слишком много перемещаемых элементов).

108. **Overflow in arithmetic operation** (переполнение при выполнении арифметической операции).

109. **No enclosing For, While or Repeat statement**. Использование операторов Break и Continue вне пределов цикла.

112. **Case constant out of range** (в операторе case константа не вписывается в допустимый диапазон).

113. **Error in statement** (ошибка в операторе).

114. **Cannot call an interrupt procedure** (не вызывается процедура прерывания).

116. **Must be in 8087 mode compile this** (для компиляции данной конструкции должен быть установлен режим сопроцессора). Данная конструкция может быть откомпилирована только в режиме {$N+}.

117. **Target address not found** (заданный адрес отсутствует). Не обнаружен оператор, расположенный по заданному адресу.

118. **Included files are not allowed here** (в заданном месте не допускается включение файла). Исходные файлы нельзя включать внутри раздела операторов.

119. **No inherited methods are accesible here**. Неправильное использование ключевого слова Inherited.

121. **Invalid Qualifier** (неправильно указан квалификатор).

122. **Invalid variable reference** (неправильная ссылка на переменную).

123. **Too many symbols** (слишком много символов). Длина текста программы или модуля превышает 64К символов.

124. **Statement part too large** (слишком велик раздел операторов). Превышен максимально допустимый размер раздела операторов программы (около 24К байт).

126. **Files must be var parameters** (файлы должны быть параметрами-переменными). Параметры файлового типа в объявлении подпрограмм должны описываться как переменные.

127. **Too many conditional symbols** (слишком много условных символов). Недостаточно памяти для размещения имен, указанных в директивах условной компиляции. Рекомендуется укоротить одно или несколько символических имен.

128. **Misplaced conditional directive** (несоответствие директив условной компиляции).

129. **ENDIF directive missing** (отсутствие директива {$ENDIF}).

130. **Error in initial conditional defines** (ошибка в установке условных определений). Условия компиляции, заданные в Otions/Compiler/Conditional defines, должны разделяться пробелами, запятыми или точками с запятой.

131. **Header does not match previous definition** (заголовок не соответствует предыдущему определению). Заголовок программы, заданный в секции связи модуля или в объявлении, использующем foward, не соответствует данному заголовку.

132. **Critical disk error** (серьезная ошибка дискового накопителя).

133. **Cannot evaluate this expression** (данное выражение невозможно вычислить).

134. **Expression incorrectly terminated** (неправильно завершено выражние). Вероятно, отсутствует точка с запятой.

135. **Invalid format specifier** (неправильная спецификация формата).

136. **Invalid indirect reference** (неправильная косвенная ссылка). Возможно, используется переменная типа absolute, базовая переменная которой не объявлена в данном модуле.

137. **Structured variables are not allowed here** (в данном контексте структурные переменные недопустимы).

138. **Cannot evaluate without System unit** (нельзя вычислить без модуля System).

139. **Cannot access this symbol** (невозможен доступ к данному символу). Доступ к некоторым идентификаторам, например, переменным, возможен только после начала фактического выполнения программы.

140. **Invalid floating-point operation** (неправильная операция с плавающей точкой). Операция над двумя значениями вещественного типа привела к переполнению или делению на нуль.

141. **Cannot compile overlay to memory** (нельзя компилировать код оверлейной программы в память).

142. **Procedure or function variable expected** (ожидается переменная типа procedure или function).

143. **Invalid procedure or function reference** (неправильная ссылка на процедуру или функцию).

144. **Cannot overlay this unit** (данный модуль нельзя сделать оверлейным). Предпринята попытка объявить оверлейным модуль, который был откомпилирован без директивы {$O+}.

145. **Too many nested scopes** (использование недопустимо большого числа вложенных элементов языка).

146. **File access denied** (попытка использовать файл, предназначенный только для считывания, как выходной).

147. **Object type expected** (ожидается объектный тип).

148. **Local object types are not allowed** (локальный объектный тип недопустим).

149. **Virtual expected** (ожидается зарезервированное слово virtual).

150. **Method identifier expected** (ожидается идентификатор метода).

Сообщения об ошибках на шаге выполнения

1. **Invalid function number** (неверный номер функции). Пред-принята попытка вызова несуществующей функции DOS.

2. **File not found** (файл не найден).

3. **Path not found** (путь не найден).

4. **Too many open** files (слишком много открытых файлов: должно быть не более 15).

5. **File access denied** (запрещен доступ к файлу). Предпринята попытка записи в файл, предназначенный только для считывания.

6. **Invalid file handle** (неправильный обработчик файла).

12. **Invalid file access code** (неправильный код доступа к файлу).

15. **Invalid drive number** (неправильный номер диска).

16. **Cannot remove current** directory (нельзя удалять текущую директорию).

17. **Cannot rename across drives** (при переименовании файла нельзя указывать другое устройство).

Ошибки ввода-вывода

100. **Disk read error** (ошибка чтения диска). Предпринята попытка чтения после конца файла.

101. **Disk write error** (ошибка записи на диск). Диск целиком заполнен.

102. **File not assigned** (файл не назначен).

103. **File not open** (файл не открыт).

104. **File not open for** input (файл не открыт для ввода).

105. **File not open for outinput** (файл не открыт для вывода).

106. **Invalid numeric format** (неправильный числовой формат). Числовое значение, считанное из текстового файла, имеет неправильный формат.

Серьезные ошибки

150. **Disk is write-protected** (диск защищен от записи).
151. **Unknown unit** (неизвестное устройство).
152. **Drive not ready** (дисковод не готов к работе).
153. **Unknown command** (неизвестная команда).
154. **CRC error in data** (ошибка в данных на диске).
156. **Disk seek error** (ошибка поиска на диске).
157. **Unknown media type** (неизвестный тип носителя).
158. **Sector not found** (сектор не найден).
159. **Print out of paper** (в принтере нет бумаги).
160. **Device write fault** (ошибка при записи на устройство).
161. **Device read fault** (ошибка устройства при чтении).
162. **Hardware failure** (отказ аппаратных средств).

Грубые ошибки

200. **Division by zero** (деление на нуль).
201. **Range check error** (выход за допустимые границы). Вычисляемое или присвоенное значение не укладывается в допустимый диапазон.
202. **Stack overflow** (переполнение стека).
203. **Heap overflow error** (переполнение области динамической памяти).
204. **Invalid pointer operation** (неправильная операция с указателем).
205. **Floating point overflow** (переполнение в операции с плавающей точкой).
206. **Floating point underflow** (потеря порядка в операции с плавающей точкой).
207. **Invalid floating point operation** (неправильная операция с плавающей точкой).

Список горячих клавиш

Таблица

Горячая клавиша	Функция	Опция меню
1	2	3
F1	Активизация окна помощи	
F2	Сохранение файла, находящегося в активном окне редактирования	File/Save
F3	Загрузка файла в активное окно редактирования	File/Open
F4	Выполнение программы до строки, помеченной курсором	Run/Goto Cursor
F5	Увеличение/уменьшение размеров активного окна	Window/Zoom
F6	Активация следующего активного окна	Window/Next
F7	Выполнение очередного оператора программы или подпрограммы	Run/Trace Into
F8	Выполнение очередного оператора программы	Run/Step Over
F9	Компиляция программы/модуля и возможно связанных с ними модулей	Compile/Make
F10	Возврат в главное меню	
Ctrl-F1	Выдача справки о языковой конструкции	Help/Topic Search
Ctrl-F2	Завершение отладки программы	Run/Program Reset
Ctrl-F3	Вывод на экран списка имен активных блоков	Debug/Call Stack
Ctrl-F4	Просмотр значения выражения, изменение значения переменной	Debug/Evaluate
Ctrl-F5	Изменение размера и положения активного окна	Window Size/move
Ctrl-F7	Добавление выражения в окно наблюдений	Debug/ Add Watch
Ctrl-F8	Установка или отмена точки останова	Debug/Add Breakpoint
Ctrl-F9	Запуск программы	Run/Run
Ctrl-Del	Удаление выделенного текста	Edit/Clear
Ctrl-Ins	Помещение выделенного текста в карман	Edit/Copy
Shift-Del	Удаление выделенного текста из файла и помещение его в карман	Edit/Cut
Shift-Ins	Помещение выделенного текста из карма-на в файл, находящийся в активном окне редактирования	Edit/Paste

Продолжение таблицы

1	2	3
Shift-F1	Вывести словарь контекстной помощи	Help/Index
Shift-F2	Фильтр	Tools/Grep
Shift-F3	Компилятор	Tools/Turbo Assembler
Shift-F4	Автономный отладчик	Tools/Turbo Debugger
Alt-F1	Восстановление содержимого предыдущего окна помощи	Help/Previous Topic
Alt-F3	Закрытие активного окна	Window/Close
Alt-F5	Активизация окна вывода	Window/User screen
Alt-F6	Активизация ранее активного окна	Window/ Previous
Alt-F7	Перейти к предыдущей строке окна сообщений	Tools/Goto Previous
Alt-F8	Определение места нахождения ошибки в исходном файле	Tools/Goto Next
Alt-F9	Компиляция файла, находящегося в активном окне редактирования	Compile/ Compile
Alt-Bksp	Отменить предыдущую операцию редактирования	Edit/Undo
Alt-1..9	Открыть окно с номером 1..9	
Alt-X	Выход из среды системы	File/Exit
Alt-0	Вывод на экран списка всех открытых окон	Window/List
Alt-C	Активизация меню Compile	Compile
Alt-D	Активизация меню Debug	Debug
Alt-E	Активизация меню Edit	Edit
Alt-F	Активизация меню File	File
Alt-H	Активизация меню Help	Help
Alt-O	Активизация меню Options	Options
Alt-R	Активизация меню Run	Run
Alt-S	Активизация меню Search	Search
Alt-W	Активизация меню Window	Window

ПРИЛОЖЕНИЕ 3

Пример оформления отчета

Белорусская государственная политехническая академия

Кафедра “Двигатели внутреннего сгорания”

Вычислительная техника и программирование

ОТЧЕТ

по лабораторной работе № 1
“Программирование алгоритма линейной структуры”

Выполнил:
студент группы 101NNN Фамилия И.О.

Проверил: Фамилия И.О.

Минск 1998

1. Цель работы - научиться составлять схему алгоритма решения задачи и программу на языке Pascal.

2. Задание к работе.

Вычислить высоты треугольника со сторонами *а*,*в*,*с*, используя формулы:

$$h_a = \frac{2}{a}\sqrt{p(p-a)(p-b)(p-c)};$$

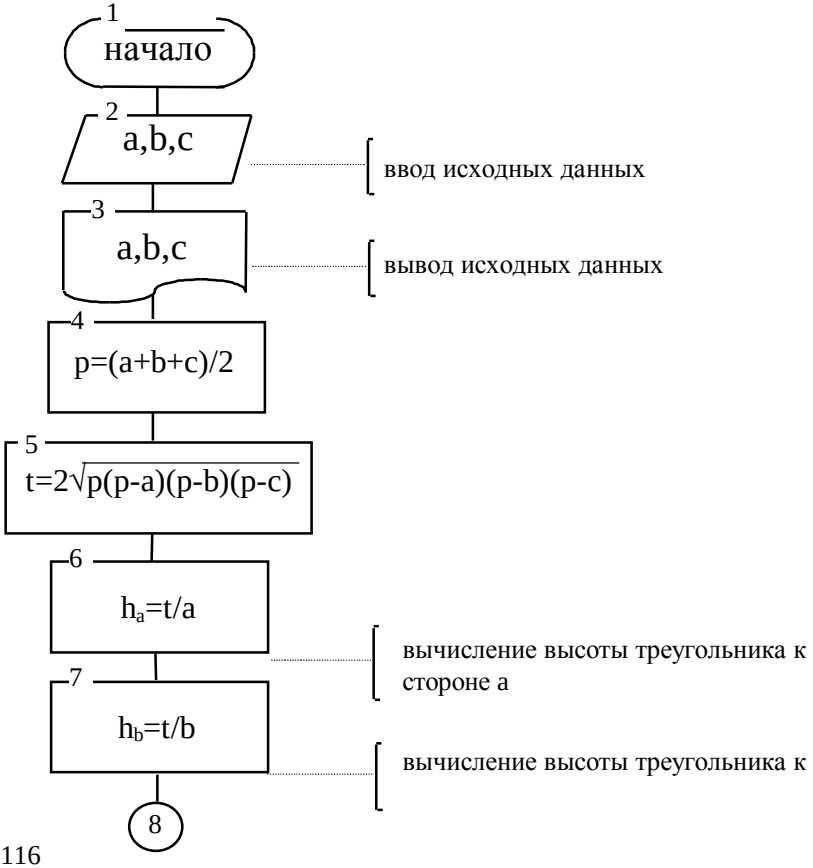
$$h_b = \frac{2}{b}\sqrt{p(p-a)(p-b)(p-c)};$$

$$h_c = \frac{2}{c}\sqrt{p(p-a)(p-b)(p-c)},$$

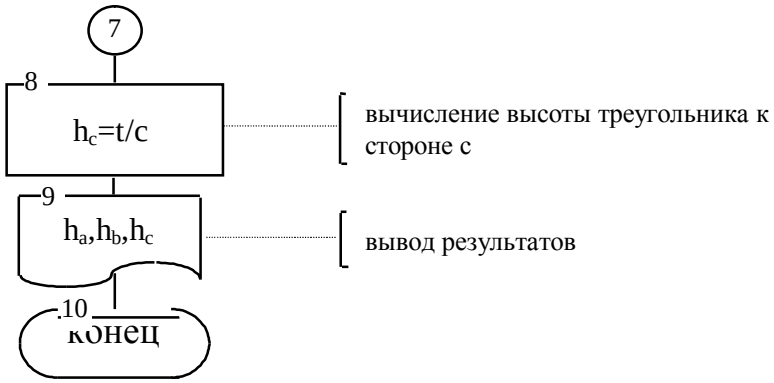
где $p = (a+b+c)/2$. Исходные данные *a*=2; *b*=3; *c*=4.

Результат вывести на экран дисплея (принтер).

3. Схема алгоритма решения задачи.



стороне b



4. Программа на языке Pascal (распечатка программы на принтере).

```
Program HTR;
{Лабораторная работа №1 “Алгоритм линейной структуры.”
Фамилия И.О. - группа 101NNN}
Var
A, B, C, P, T, HA, HB, HC: REAL;
Begin
Read(A,B,C);
Write(‘A=‘, A:2, ‘ ‘:3, ‘B=‘, B:2, ‘ ‘:3, ‘C=‘, C:2,);
P:=(A+B+C)/2;
T:=2*SQRT(P*(P-A)*(P-B)*(P-C));
HA:=T/A;
HB:=T/B;
HC:=T/C;
Writeln(‘HA=‘ ,HA);
Writeln(‘HB=‘ ,HB);
Writeln(‘HC=‘ ,HC)
End.
```

5. Результат (записывается с экрана дисплея или распечатывается на принтере).

Литература

1. Алексеев В.Е., Ваулин А.С., Петрова Г.Б. Вычислительная техника и программирование. Практикум по программированию: Практическое пособие/ Под ред. А.В. Петрова. - М.: Высш. шк., 1991. - 400 с.
2. Бородич Ю.С., Вальвачев А.Н., Кузьмич А.И. Паскаль для персональных компьютеров: Справ. пособие. - Мн.: Высш. шк.: БФ ГИТМП “НИКА”, 1991. - 365 с.
3. Вальвачев А.Н., Крисевич В.С. Программирование на языке ПАСКАЛЬ для персональных ЭВМ ЕС: Справ. пособие. - Мн.: Высш. шк., 1989. - 223 с.
4. ГОСТ 19.701 - 90. Схемы алгоритмов программ, данных и систем. Условные обозначения и правила выполнения . - М.: Издательство стандартов, 1991. - 26 с.
5. Епанешников А.М., Епанешников В.А. Программирование в среде Turbo Pascal 7.0. - 3-е изд., стер. - М.: “ДИАЛОГ-МИФИ”, 1996. - 288 с.
6. Сергиевский М.В., Шаламов А.В. ТУРБО ПАСКАЛЬ 7.0. Язык, Среда программирования. - М.: Машиностроение, 1994. - 254 с.
7. Фигурнов В.Э. IBM PC для пользователя. Изд. 5-е. - М.: Финансы и статистика, 1994. - 368 с.