ФЕДЕРАЛЬНОЕ АГЕНСТВО ПО ОБРАЗОВАНИЮ

КЕМЕРОВСКИЙ ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ
ПИЩЕВОЙ ПРОМЫШЛЕННОСТИ

Кафедра прикладной математики и информатики

Иванец Г.Е., Ивина О.А.

ИНФОРМАТИКА

ПРОГРАММИРОВАНИЕ В СИСТЕМЕ TURBO PASCAL

Лабораторный практикум
для студентов, обучающихся по направлению подготовки
260000 «Технология продовольственных продуктов и потребительских товаров»

Кемерово 2009

УДК 681.3(075)
ББК 32.97я7
И18

Рецензенты

Ю.Н. Захаров, зав. кафедрой «Вычислительная математика» Кемеровского государственного университета, д-р физ. – мат. наук, профессор

А.В. Иголинский, технический директор по информационным технологиям ЗАО «НТК ГрадЛАН»

Рекомендовано редакционно-издательским советом Кемеровского технологического института пищевой промышленности

Иванец Г.Е., Ивина О.А.

И18 Информатика. Программирование в системе Turbo Pascal: лабораторный практикум / Иванец Г.Е., Ивина О.А; Кемеровский технологический институт пищевой промышленности. - Кемерово, 2009. – 94с.

ISBN

Предназначено для студентов, обучающихся по направлению подготовки 260000 «Технология продовольственных продуктов и потребительских товаров», изучающих программирование в системе Turbo Pascal в курсе информатики; может быть полезно для преподавателей при подготовке лекций и проведении практических занятий и лабораторных работ.

УДК 681.3(075)
ББК 32.97я7

IBSN

Введение

Информатика – это дисциплина, изучающая информационные процессы, связанные с накоплением, хранением, преобразованием (обработкой) и передачей информации. Для решения этих задач необходимо знать как аппаратные средства современных персональных компьютеров, так и прикладное программное обеспечение. Исходя из определения информационных процессов, можно условно классифицировать прикладные программы для их изучения. Инструментальные среды занимают важное место в этой классификации.

Инструментальные средства (среды) - это интегрированные системы для производства программных продуктов. С точки зрения определения сущности информационных процессов эти системы предназначены для преобразования информации, а точнее, данных.

В данной методической разработке представлен комплексный подход к изучению этого раздела дисциплины «Информатика». Методическая разработка по содержанию является лабораторным практикумом. Каждая тема сопровождается кратким теоретическим изложением, формулировкой заданий, контрольными вопросами, примерами выполнения заданий. Такой комплексный подход позволит студентам относительно самостоятельно выполнять индивидуальные задания. Изучение программирования в курсе «Информатика» является обязательным элементом в процессе обучения, т.к. оно способствует развитию логического мышления студентов и поможет будущим специалистам решать инженерно – технические задачи различной сложности. Кроме того, студенты, знающие алгоритмический язык, легко осваивают при необходимости, другие языки, поскольку их связывает единая алгоритмическая основа. Уровень логического мышления позволяет также относительно легко изучать другие прикладные программы, являющиеся предметом изучения дисциплины «Информатика».

Лабораторная работа №1

Программирование алгоритмов линейной структуры

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

Краткая теория

Линейный вычислительный процесс – это структура, в которой действия алгоритма выполняются последовательно друг за другом. Для программной реализации этой структуры требуются три оператора: оператор ввода (точнее, процедура ввода), оператор присваивания, оператор вывода (точнее, процедура вывода).
Оператор ввода необходимо сопровождать пояснительным текстом:

write('введите переменную');
readln(идентификатор переменной);

Оператор присваивания имеет следующую структуру:

A:=B;

где ***A*** – идентификатор переменной, ***B*** – арифметическое выражение, либо константа, либо идентификатор переменной (***A***:=sin(x)-sqr(x); ***A***:=5.7; ***A***:=Y).
Процедура вывода оформляется с помощью ключевого (зарезервированного) слова ***write*** или ***writeln.*** Символы ***ln (line)*** означают перевод курсора на следующую строку.
Примеры записи оператора вывода:

Writeln('x=',x:6:2); writeln('k=',k);

Первый оператор выводит значение переменной ***x,*** причем эта переменная вещественного типа. Для вывода результата в обычном виде необходимо указать поле вывода, т. е. количество позиций (в данном примере ***6*** с дробной частью ***2***). Второй оператор выводит значение переменной целого типа ***k***, для которой общее поле вывода назначать не обязательно, но, если оно при-

сутствует, то нельзя указывать количество позиций для дробной части.

Таблица математических функций на алгоритмическом языке PASCAL

Таблица 1

Запись функций на алгоритмическом языке Pascal

Математическая запись	Запись на алгоритмическом языке PASCAL
$\|X\|$	ABS(X)
X^2	SQR(X)
e^x	EXP(X)
Ln x	Ln(x)
$\sqrt{x}$	Sqrt(x)
x^p	Exp(p*ln(x))
Sin x	Sin(x)
Cos x	Cos(x)
Tg x	Sin(x)/cos(x)
Arctg x	Arctan(x)
π	Pi
[x] - целая часть числа	Int(x)
Дробная часть числа	Frac(x)

Правила записи арифметических выражений:

1) Выражение записывается в строку, порядок действий устанавливается скобками
2) Соблюдение баланса скобок
3) Аргумент функции заключается в скобки (смотри 2-ой столбец).

($\sin^2 x \rightarrow$sqr(sin(x)) – верно, sin^2*x – не верно)

4) Знаки математических операций пропускать нельзя
5) Операция возведения в степень записывается через определение выражения по его логарифму
6) Функция tg x выражается через sin x и cos x (смотри таблицу)

Приоритеты действий при вычислении выражений:

1)Вычисления в круглых скобках;
2) Вычисления значений функций
3) Унарные операции (NOT, унарный + , унарный −)
4) Операции типа умножения (∗, /, div, mod, and)
5) Операции типа сложения (+, −, or)
6) Операции отношения (=, < >,<, >, <=,>=)

Пояснения: div – целая часть от деления (5 div 2→2)
Mod – остаток от деления (5 mod 2→1).

Тип результата вычислений функций ABS(X) и SQR(X) совпадает с типом аргумента. Тип результата остальных вышеперечисленных функций всегда вещественный.

Контроль по теме

1. Правила записи арифметических выражений.
2. Оператор присваивания. Правила его оформления.
3. Организация простейшего ввода-вывода данных.
4. Правила записи стандартных функций.
5. Порядок выполнения действий. Приоритет операций.
6. Идентификатор. Правила обозначения.
7. Формы представления чисел. Вывод данных в нормализованном виде.
8. Основные типы данных. Диапазон типов.
9. Определение констант, переменных. Описание их в программе.

Задание 1

Вычислить значения переменных указанных в таблице заданий по заданным расчетным формулам и наборам исходных данных.

Выполнение задания должно включать все этапы подготовки и решения задач на ЭВМ.

1. Постановка задачи

2. Разработка алгоритма
3. Составление программы
4. Тестирование программы, т.е. проверка машинного результата с помощью подбора тестового варианта исходных данных для устного вычисления или с помощью микрокалькулятора

Таблица 2

Задания по вариантам

Вариант задания	Расчетные формулы
1	$a=\dfrac{2\cos(x-\pi/6)}{1/2+\sin^2 y};\ b=1+\dfrac{z^2}{3+z^2/5}$
2	$y=\left\lvert x^{y/x}-\sqrt[3]{y/x}\right\rvert;\ \varphi=(y-x)\dfrac{y-z/(y-x)}{1+(y-x)^2}$
3	$s=1+x+\dfrac{x^2}{2!}+\dfrac{x^3}{3!}+\dfrac{x^4}{4!};\ \varphi=x(\sin x^3+\cos^2 y)$
4	$y=e^{-bt}\sin(at+b)-\sqrt{\lvert bt+a\rvert};$ $s=b\sin(at^2\cos 2t)-1$
5	$\omega=\sqrt{x^2+b}-b^2\sin^3(x+a)/x;$ $y=\cos^2 x^3-x/\sqrt{a^2+b^2}$
6	$s=x^3 tg^2(x+b)^2+a/\sqrt{x+b};\ Q=\dfrac{bx^2-a}{e^{ax}-1}$
7	$R=x^2(x+1)/b-\sin^2(x+a);$ $S=\sqrt{xb/a}+\cos^2(x+b)^3$

Окончание табл.2

8	$y = \sin^3(x^2 + a)^2 - \sqrt{x/b}\,;\ z = \dfrac{x^2}{a} + \cos(x+b)^3$
9	$f = \sqrt{m \cdot tgt + \lvert c \cdot \sin t \rvert}\,;\ z = m\cos(bt\sin t) + c$
10	$y = btg^2 x - \dfrac{a}{\sin^2(x/a)}\,;\ d = ae^{-\sqrt{a}}\cos(bx/a)$
11	$f = \ln(a + x^2) + \sin^2(x/b)\,;\ z = e^{-cx}\,\dfrac{x + \sqrt{x+a}}{x - \sqrt{\lvert x-b \rvert}}$
12	$y = \dfrac{a^{2x} + b^{-x}\cos(a+b)x}{x+1}\,;$ $R = \sqrt{x^2 + b} - b^2 \sin^3(x+a)/x$
13	$z = \sqrt{ax\sin 2x + e^{-2x}(x+b)}\,;$ $\omega = \cos^2 x^3 - x/\sqrt{a^2 + b^2}$
14	$U = \dfrac{a^2 x + e^{-x}\cos bx}{bx - e^{-x}\sin bx + 1}\,;$ $f = e^{2x}\ln(a+x) - b^{3x}\ln(b-x)$
15	$z = \dfrac{\sin x}{\sqrt{1 + m^2 \sin^2 x}} - cm\ln mx\,;$ $s = e^{-ax}\sqrt{x+1} + e^{-bx}\sqrt{x+1.5}$

Пример выполнения задания 1

Задание. Вычислить значения переменных по заданным расчетным формулам и наборам исходных данных.

Постановка задачи

а) обозначение переменных

a,b,c,x,m – исходные данные;

s,z – результат;

б) классификация переменных

a,b,c,x,m,s,z – простые переменные вещественного типа.

в) расчетные формулы

$$z = \frac{\sin x}{\sqrt{1 + m^2 \sin^2 x}} - cm \ln mx;\ s = e^{-ax}\sqrt{x+1} + e^{-bx}\sqrt{x+1.5}$$

Разработка алгоритма

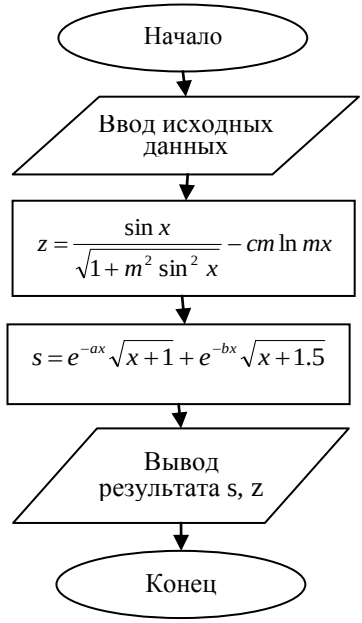


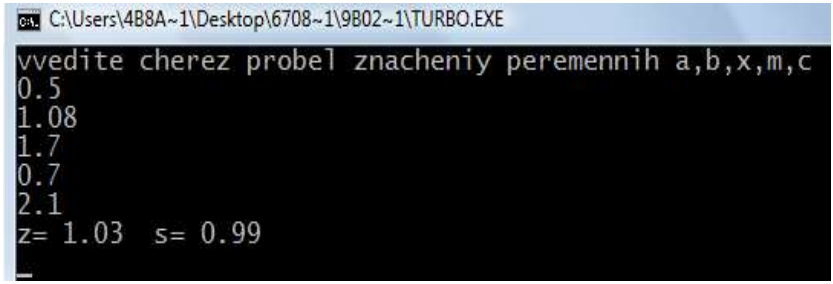
Рис.1. Блок – схема алгоритма линейной структуры примера выполнения 1 – го задания.

Составление программы

```
Program lab_1;
Uses CRT;
Var  a,b,x,m,c,z,s : real;
Begin clrscr;
Writeln (‘введите через пробел значения переменных a,b,x,m,c’);
Readln (a,b,x,m,c);
z:=sin(x)/sqrt (1+m*m*sqr(sin(x))-c*m*ln(m*x);
s:= exp(-a*x)*sqrt(x+1)+exp(-b*x)*sqrt(x+1.5);
Writeln (‘z=’,z:5:2,’’:2,’s=’,s:5:2);
Readln;
End.
```

Тестирование программы.

После получения результата работы программы провести тестирование программы с помощью микрокалькулятора или путем подбора исходных данных для устного просчета.



Задание 2

Решить смысловую задачу по вариантам.

1. Написать программу вычисления сопротивления электрической цепи, состоящей из двух последовательно соединенных сопротивлений. (Величина первого сопротивления 15 Ом, второго сопротивления 27,3 Ом)

2. Написать программу вычисления площади трапеции. (Длины оснований равны 4 и 9см, высота трапеции 5см)
3. Написать программу вычисления площади треугольника, если известна длина основания (8,5см) и высота (10см).
4. Написать программу вычисления площади треугольника, если известны длины двух его сторон и величина угла между этими сторонами. (Длина первой стороны 25см, второй стороны 17см, величина угла 20^0)
5. Написать программу вычисления площади поверхности и объёма цилиндра, с радиусом основания 7см и высотой 23см.
6. Написать программу вычисления площади круга с радиусом 5см.
7. Написать программу вычисления стоимости покупки, состоящей из нескольких тетрадей и карандашей. (Количество тетрадей – 5шт., стоимость – 2,75руб., количество карандашей – 8шт., стоимость – 0,85руб.)
8. Написать программу вычисления объёма параллелепипеда. (Длина, ширина и высота параллелепипеда – 7,5см, 2,5см и 3см соответственно)
9. Написать программу вычисления площади поверхности и объёма шара, с радиусом 3,5см.
10. Написать программу вычисления объёма полого цилиндра, если радиус цилиндра равен 5см, радиус отверстия – 2см, высота цилиндра – 9см.
11. Написать программу вычисления объёма конуса, радиус основания которого равен 2см, высота – 5см.
12. Написать программу вычисления объёма куба, если длина ребра составляет 9,5см.
13. Написать программу вычисления сопротивления электрической цепи, состоящей из двух параллельно соединенных сопротивлений. (Величина первого сопротивления 15 Ом, второго сопротивления 20 Ом)
14. Написать программу вычисления площади прямоугольника. (Длина первой стороны – 3см, второй – 7см)

15. Написать программу вычисления силы тока в электрической цепи, если напряжение равно 36 вольт, сопротивление – 1500 Ом.
16. Написать программу вычисления величины дохода по вкладу. Процентная ставка (20% годовых), время хранения (30 дней) и величина вклада (2500руб.) задаются во время работы программы.

Пример выполнения задания 2

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

Постановка задачи

а) обозначение переменных

r – радиус основания цилиндра;
h – высота цилиндра;
S – площадь основания цилиндра;
V – объем цилиндра.

б) классификация переменных

r,h,S,V – простые переменные вещественного типа.

в) расчетные формулы

вычисление объема по формуле

$$V=S*h;$$

вычисление площади по формуле

$$S=\pi*r^2 .$$

Разработка алгоритма

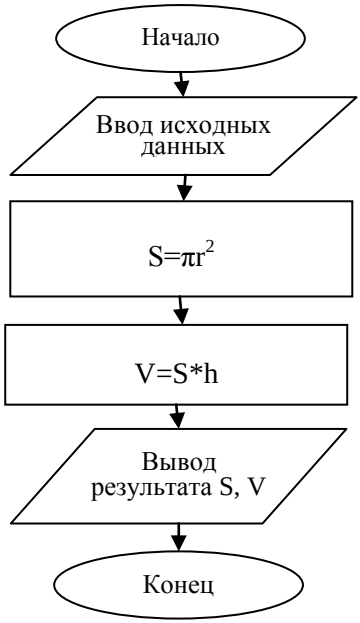


Рис. 2. Блок – схема алгоритма линейной структуры примера выполнения 2- го задания.

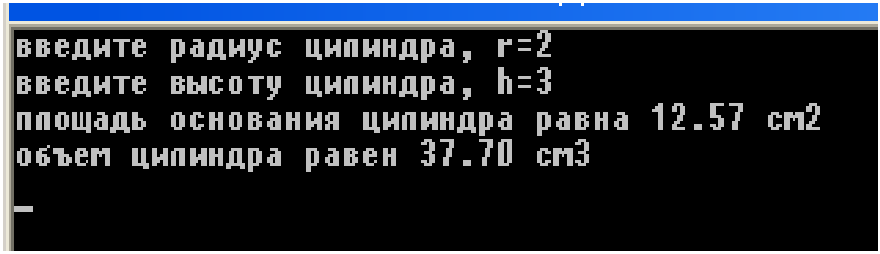
Составление программы

```
PROGRAM CYLINDR;
        {Подключение стандартного модуля}
Uses CRT;
        {Раздел описания переменных}
Var r,h,S,V: real;
Begin clrscr;
        {Ввод исходных данных с пояснением}
Write ('введите радиус цилиндра, r=');
Readln (r);
Write ('введите высоту цилиндра, h=');
```

```
Readln (h);
        {вычисление площади основания цилиндра, S}
S:=pi*sqr(r);
        {вычисление объема цилиндра, S}
V:=S*h;
        {вывод результатов}
Writeln ('площадь основания цилиндра равна', S:6:2,' см2');
Writeln ('объем цилиндра равен', V:6:2,' см3');
        {задержка завершения программы для просмотра полученных результатов}
Readln;
End.
```

Тестирование программы.

После получения результата работы программы провести тестирование программы с помощью микрокалькулятора или путем подбора исходных данных для устного просчета.



Вычисления на калькуляторе: $S=\pi*r^2 = 3,14*2^2 = 12,56$ $см^2$

$V=S*h = 12,56*3 = 37,68$ $см^3$

Лабораторная работа №2

Программирование алгоритмов разветвляющейся структуры

Цель работы – овладение практическими навыками разработки алгоритмов разветвляющейся структуры, развитие навыков по отладке и тестированию программ.

Краткая теория

Разветвляющейся алгоритм – это структура, в которой последовательный порядок действий нарушается условием, при проверке которого управление передается одному из двух или нескольких направлений (множественный выбор). В первом случае структура реализуется оператором условного перехода:

IF <условие> THEN <оператор 1> ELSE <оператор 2>;

Оператор работает следующим образом: если **условие** истинно, то выполняется **оператор 1** (или группа операторов), если условие ложно, то выполняется **оператор 2** (или группа операторов)

В блок-схемах этому оператору соответствует блок выбора в форме ромба с одним входом и двумя выходами (**да, нет**). В примерах выполнения заданий приведено его графическое изображение. Оператор условного перехода относится к группе сложных операторов, т.е. он может содержать внутри себя другие операторы, в частности, другой условный оператор. В этом случае структура называется **вложенной развилкой**. На одном из выходов (**да, нет**) или на обоих сразу может быть несколько операторов, которые необходимо заключать в операторные скобки **BEGIN…END.** Группа операторов, заключенная в операторные скобки, называется **составным оператором.** Примеры записи оператора условного перехода, включающего в себя составной оператор, приведены в таблице 3

Таблица3

Формы записи условного оператора

Количество операторов		Условная запись оператора
на выходе **да**	на выходе **нет**	
Несколько	один	IF <условие> THEN BEGIN <оператор 1>; <оператор 2>, …….. END ELSE <оператор>;

Один	несколько	IF <условие> THEN
		<оператор 1>
		ELSE
		BEGIN
		<оператор 2
		оператор 3,
		……
		END;

Несколько	несколько	IF <условие> THEN
		BEGIN
		<оператор 1>
		<оператор 2>,
		……
		END
		ELSE
		BEGIN
		<оператор 3>
		<оператор 4>,
		……..
		END;

Примечание: перед ключевым словом ELSE точку с запятой ставить нельзя.

Контроль по теме

1. Оператор условного перехода. Правила записи и его работы. Полная и неполная формы записи.
2. Оператор безусловного перехода. Какие изменения необходимо внести в структуру программы при его использовании ?
3. Определение структуры – вложенная развилка. Пример.
4. Составить последовательность операторов для вычисления величины Z=0, если X<-2; Z=1, если - 2≤X≤2; Z= -1, если X>2.
5. Зачем необходимо при отладке программы тестировать все ветви алгоритма?
6. В каких случаях в операторе условного перехода используют составной оператор?

Задание 1

Исследовать функцию на неопределенность.
Пояснение к заданию: известно, что функция может быть непрерывна в интервале [-∞,+∞], например, **y=sin(x)**, или иметь точки разрыва (неопределенности), например, **y=tg(x)** при значениях **x=π/2±n*π** (n=0,1,2…). Функция **y=ln(x)** – не определена при значениях аргумента x<=0. Если это не учитывать в программе, то её выполнение будет прерываться ошибкой:

Invalid floating point operation

В случае неопределенности функции необходимо исключить возможность её вычисления и печатать на экране примерно такой текст – «Функция не определена», естественно, печатая значения аргумента, при котором возникает неопределенность. Для облегчения выполнения задания в таблице 4 приведены точки разрыва (область неопределенности) для некоторых функций:

Таблица 4

Точки разрыва (область неопределенности) функций

Функция в математической записи	Точки разрыва (область неопределенности)
$\sqrt{x}$	x<0
Ln x	x<=0
Arcsin x	-1<x или x>1 ($\|x\|>1$)
Arccos x	-1<x или x>1 ($\|x\|>1$)
Tg x	$x=\frac{\pi}{2}\pm\pi n$
$\frac{1}{x}$	x=0

Таблица 5

Варианты заданий

Номер варианта	Исследуемая функция
1	$\ln^3 x+x^2)/\sqrt{x+t}$
2	$a/x+\sqrt{x^2+1}$
3	$\sqrt{x+t}+1/x$
4	$\ln(x+7\sqrt{x})$
5	Arccos(x-b)
6	$x^3\sqrt{x-a}$

Окончание табл.5

Номер варианта	Исследуемая функция
7	$3tgx$
8	$bx - \lg bx$
9	$at^2 \ln t$
10	$\pi x^2 - 7 / x^2$
11	$(\ln^3 x + x^2) / \sqrt{x+t}$
12	Arcsin(x+b)
13	$\frac{a+b}{e^x + \cos x}$
14	$a \lg x + \sqrt[3]{\lvert x \rvert}$
15	$\sqrt{at^2 + b\cos t + 1}$

Пример выполнения задания

Исследовать функцию y=ln(x-c-b) на неопределенность, т.е. разработать программу вычисления функции при любых значениях аргумента. Если вычисление невозможно (точки разрыва или область неопределенности), выдать сообщение «Функция не определена».

Постановка задачи

а) обозначение переменных и констант

x – исходная переменная (аргумент функции);

y – конечная переменная (значение функции);

c, b – постоянные величины.

б) классификация переменных

x,y – простые переменные вещественного типа

в) расчетные формулы

y=ln(x-c-b), если (x-b-c)>0, c=2,5, b=1,5

функция не определена, если(x-b-c)<=0

Разработка алгоритма

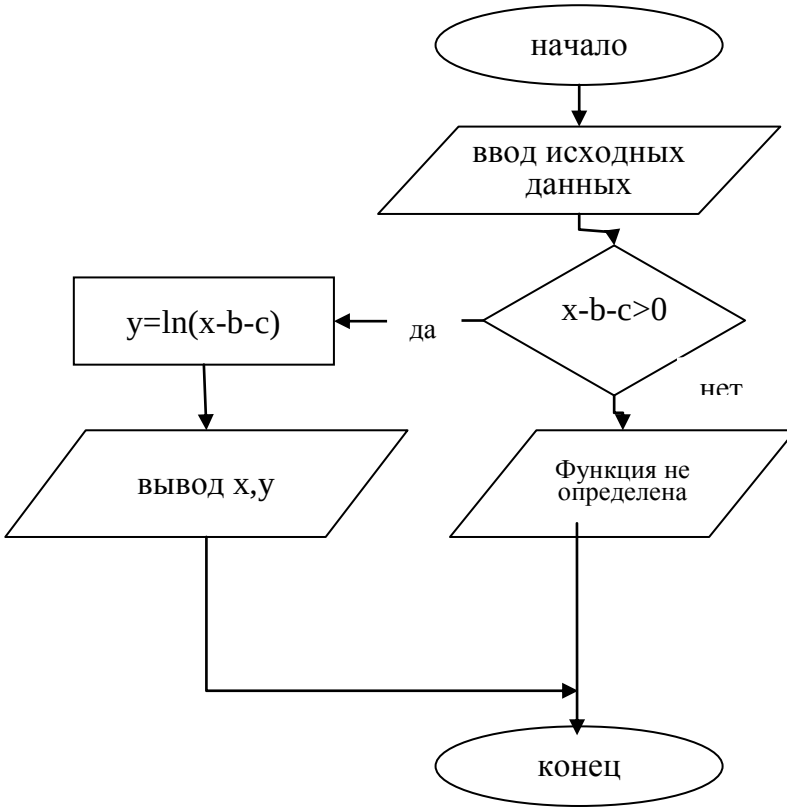


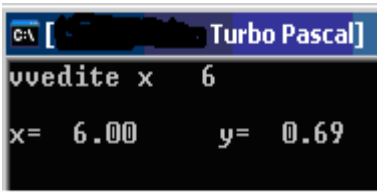
Рис.3. Блок – схема исследования функции на неопределенность

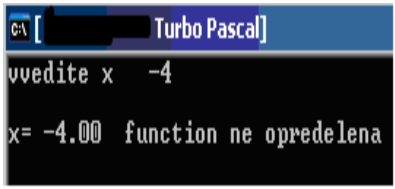
Составление программы

PROGRAM neopred;
Uses CRT;
Const c=2,5;b=1,5;

```
Var  x,y: real;
Begin clrscr;
Write ('введите значение переменной  x');
Readln (x);
If  x-b-c>0  then begin
          y:=ln(x-b-c);
          writeln (x:6:2,’’5, y:6:2); end
          else
Writeln (‘при x=’,x:6:2, '-функция не определяется');
Readln;
End.
```

Тестирование программы





Задание 2

1. Вычислить значение функции, заданной в таблице. В разработанном алгоритме предусмотреть возможные неопределенности.
2. Выполнить программу и протестировать все ветви алгоритма.
3. Для облегчения тестирования вывод результата сопроводить текстом «вычислено по формуле……».

Таблица 6

Варианты заданий

Номер варианта	Функция	Условие	Исходные Данные
1	$y=\begin{cases} at^2\ln t \\ 1 \\ e^{at}\cos bt \end{cases}$	1≤t≤2 t<1 t>2	a=-0.5 b=2
2	$y=\begin{cases} \pi x^2-7/x^2 \\ ax^3+7\sqrt{x} \\ \ln(x+7\sqrt{x}) \end{cases}$	x<1.3 x=1.3 x>1.3	a=1.5
3	$\omega=\begin{cases} ax^2+bx+c \\ a/x+\sqrt{x^2+1} \\ (a+bx)/\sqrt{x^2+1} \end{cases}$	x<1.2 x=1.2 x>1.2	a=2.8 b=-0.3 c=4
4	$y=\begin{cases} \pi x^2-7/x^2 \\ ax^3+7\sqrt{x} \\ \ln(x+7\sqrt{\lvert x+a\rvert}) \end{cases}$	x<1.4 x=1.4 x>1.4	a=1.65
5	$y=\begin{cases} 1.5\cos^2 x \\ 1.8ax \\ (x-2)^2+6 \\ 3tgx \end{cases}$	x<1 x=1 1<x<2 x>2	a=2.3

Продолжение табл.6

Номер варианта	Функция	Условие	Исходные данные
6	$\omega = \begin{cases} x^3\sqrt{x-a} \\ x\sin ax \\ e^{-ax}\cos ax \end{cases}$	x>a x=a x<a	a=2.5
7	$Q = \begin{cases} bx - \lg bx \\ 1 \\ bx + \lg bx \end{cases}$	bx<1 bx=1 bx>1	b=1.5
8	$y = \begin{cases} \sin x \lg x \\ \cos^2 x \end{cases}$	x>3.5 x≤3.5	-
9	$f = \begin{cases} \lg(x+1) \\ \sin^2\sqrt{\lvert ax\rvert} \end{cases}$	x>1 x≤1	a=20.3
10	$z = \begin{cases} (\ln^3 x + x^2)/\sqrt{x+t} \\ \sqrt{x+t} + 1/x \\ \cos x + t\sin^2 x \end{cases}$	x<0.5 x=0.5 x>0.5	t=2.2
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≤x<6 x≥6	a=2.6 b=-0.39
12	$y = \begin{cases} a\lg x + \sqrt[3]{\lvert x\rvert} \\ 2a\cos x + 3x^2 \end{cases}$	x>1 x≤1	a=0.9

Окончание табл.6

Номер варианта	Функция	Условие	Исходные данные
13	$\omega = \begin{cases} a/i + bi^2 + c \\ i \\ ai + bi^3 \end{cases}$	i<4 4≤i≤6 i≥6	a=2.1 b=1.8 c=-20.5
14	$z = \begin{cases} a\sin x(\frac{i^2+1}{n}) \\ \cos(i+1/n) \end{cases}$	$\sin\frac{i^2+1}{n} > 0$ $\sin\frac{i^2+1}{n} < 0$	a=0.3 n=10
15	$\omega = \begin{cases} \sqrt{at^2 + b\sin t + 1} \\ at + b \\ \sqrt{at^2 + b\cos t + 1} \end{cases}$	t<0.1 t=0.1 t>0.1	a=2.5 b=0.4

Пример выполнения задания 2

Вычислить y=a*sin(x). если x≥1, a=2,5
y=b*cos(x), если x<1, b=1,5

Постановка задачи

а) обозначение переменных
x – переменная величина, исходные данные;
y – переменная величина, результат;
a,b – постоянные величины.
б) классификация переменных
x,y – простые переменные вещественного типа
в) расчетные формулы
y=a*sin(x). если x≥1, a=2,5
y=b*cos(x), если x<1, b=1,5

Разработка алгоритма

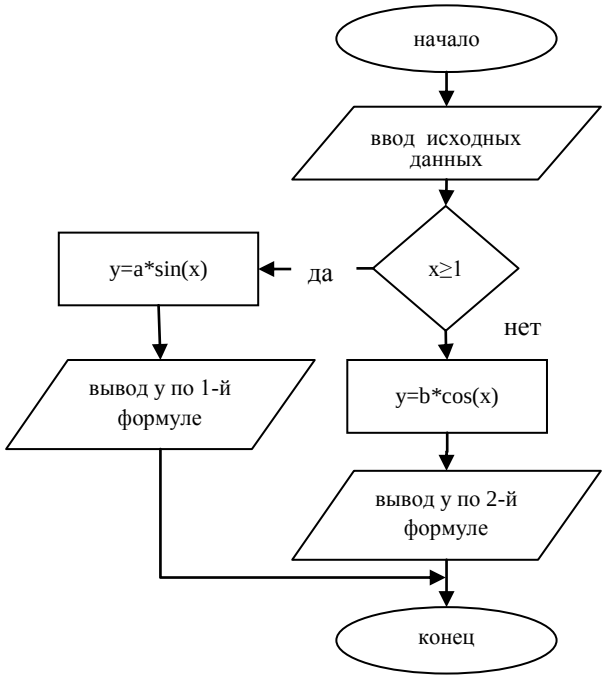


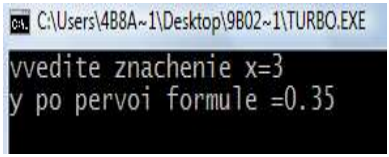
Рис.4. Алгоритм вычисления функции (задание 2).

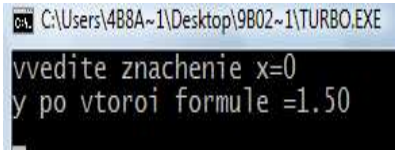
Составление программы

```
PROGRAM razvetvl;
Uses CRT;
Const  a=2,5;b=1,5;
Var  x,y: real;
Begin clrscr;
Write ('введите значение переменной  x');
Readln (x);
If  x>=1  then begin
```

```
        y:=a*sin(x);
        writeln (‘y по 1-й формуле =’, y:6:2); end
        else begin
        y:=b*cos(x);
Writeln ('y по 2-й формуле = ', y:6:2); end;
Readln;
End.
```

Тестирование программы





Лабораторная работа №3

Вычислительный процесс циклической структуры

Цель работы – овладение практическими навыками разработки, программирования вычислительного процесса циклической структуры, развитие дальнейших навыков по отладке и тестированию программы.

Краткая теория

Циклический вычислительный процесс – это структура, в которой действия повторяются многократно. Повторяющиеся действия внутри цикла называют **телом цикла**. Для правильной организации работы цикла необходимо внутри тела цикла иметь переменную, управляющую циклом. Эта переменная называется **параметром цикла**. Во избежание зацикливания, т.е. бесконечной работы цикла, параметр цикла должен изменяться по определенному закону. В Паскале возможна организация трех структур циклов: с предусловием (**while**), с постусловием (**repeat**) и с параметром или со счетчиком – (**for**).

Цикл с предусловием (while) – это структура, в которой условие проверки цикла на окончание стоит перед телом цикла (рис. 5).

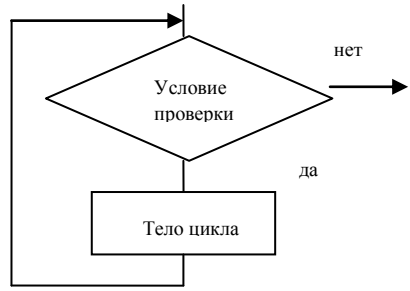


Рис.5. Условная схема цикла с предусловием.

Как видно из рисунка 5, цикл с предусловием выполняется до тех пор, пока условие проверки цикла на окончание **истинно.** Цикл закончит свою работу, когда условие проверки станет **ложным.**

Структура оператора цикла с предусловием:

While <условие проверки цикла на окончание> do
Begin
< тело цикла>
End;

Операторные скобки **Begin…End** ставятся в том случае, если тело цикла содержит более одного оператора.

Цикл с постусловием (repeat) – это структура, в которой условие проверки цикла на окончание стоит после тела цикла (рис.6).

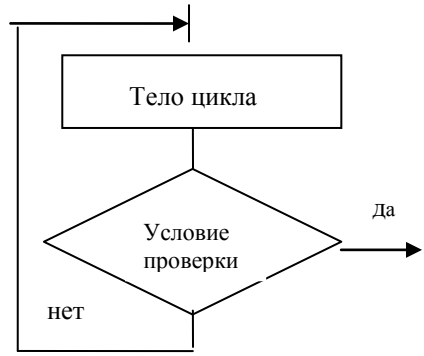


Рис.6. Структура цикла с постусловием.

Как видно из рис.6, цикл с постусловием работает в том случае, если условие проверки цикла на окончание **ложно,** когда оно станет **истинным,** цикл закончит свою работу.

Структура цикла с постусловием:

Repeat
<тело цикла>
Until <условие проверки цикла на окончание>:

Цикл с параметром (со счетчиком) – это структура, в которой циклом управляет переменная целого типа, имеющая начальное,

конечное значения и шаг изменения: плюс один(+1) или минус один(-1). В отличие от предыдущих циклов, которые относятся к группе циклов с неизвестным числом повторений, цикл с параметром является структурой с известным числом повторений цикла. В блок-схемах эта структура изображается следующим образом (рис.7):

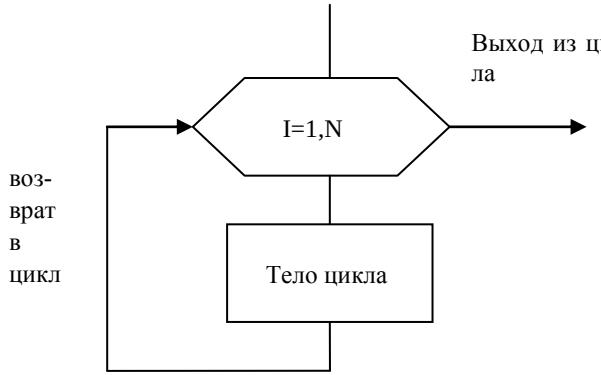


Рис.7. Структура цикла со счетчиком.

I – параметр цикла (любой идентификатор **целого типа**);
N – конечное значение параметра цикла (любой идентификатор **целого типа**). Конечное значение может быть выражением с результатом **целого типа** или константой **целого типа.**
Работает цикл со счетчиком по следующим правилам:

1. Параметру цикла присваивается начальное значение
2. Выполняется условие проверки цикла на окончание
3. Если условие проверки истинно, выполняется тело цикла
4. Параметр цикла изменяется по определенному закону
5. Выполняется условие проверки цикла на окончание
6. Если условие проверки истинно, выполняется тело цикла
7. Если условие проверки ложно, цикл заканчивает работу

Таким образом, цикл со счетчиком работает по таким же правилам, что и цикл с предусловием.

Структура цикла со счетчиком (шаг +1):

```
For I=1 to N do
Begin
<тело цикла>
End;
```

Структура цикла со счетчиком (шаг -1):

```
For I=N downto 1 do
Begin
<тело цикла>
End;
```

Контроль по теме

1. Правила организации циклов: WHILE, REPEAT, FOR.
2. Объясните основные преимущества и недостатки каждого из них.
3. Что такое параметр цикла, тело цикла?
4. Что такое «зацикливание»? В каких случаях возникает эта ошибка?
5. Какой из трех циклов всегда сработает хотя бы один раз?
6. При каком условии заканчиваются циклы: WHILE, REPEAT, FOR.?

Задание 1

Вычислить значение функции, заданной в таблице заданий (в соответствии с вариантом задания) тремя способами (с использованием команд while, repeat, for). Осуществить вывод значений вводимых исходных данных и результат вычисления значения функции в таблицу, сопровождая вывод наименованиями переменных.

Таблица 7

Варианты заданий

Номер варианта	Функция	Условие	Исходные Данные	Диапазон и шаг изменения аргумента
1	$y=\begin{cases} mt^2 \ln t \\ 1 \\ e^{mt}\cos nt \end{cases}$	1≤t≤2 t<1 t>2	m=-0.5 n=2	tЄ[0:3] Δt=0.15
2	$y=\begin{cases} \pi x^2 - 7/x^2 \\ mx^3 + 7\sqrt{x} \\ \ln(x+7\sqrt{x}) \end{cases}$	x<1.3 x=1.3 x>1.3	m=1.5	xЄ[0:3] Δx=0.15
3	$\omega=\begin{cases} mx^2 + nx + c \\ m/x + \sqrt{x^2+1} \\ (m+nx)/\sqrt{x^2+1} \end{cases}$	x<1.2 x=1.2 x>1.2	m=2.8 n=-0.3 c=4	xЄ[0:3] Δx=0.15
4	$y=\begin{cases} \pi x^2 - 7/x^2 \\ mx^3 + 7\sqrt{x} \\ \ln(x+7\sqrt{\lvert x+m\rvert}) \end{cases}$	x<1.4 x=1.4 x>1.4	m=1.65	xЄ[0:3] Δx=0.15
5	$y=\begin{cases} 1.5\cos^2 x \\ 1.8mx \\ (x-2)^2+6 \\ 3tgx \end{cases}$	x<1 x=1 1<x<2 x>2	m=2.3	xЄ[0:3] Δx=0.15

Продолжение табл.7

Номер варианта	Функция	Условие	Исходные Данные	Диапазон и шаг изменения аргумента
6	$\omega = \begin{cases} x^3\sqrt{x-m} \\ x\sin mx \\ e^{-mx}\cos mx \end{cases}$	x>a x=a x<a	m=2.5	xЄ[0:3] Δx=0.15
7	$Q = \begin{cases} nx - \lg nx \\ 1 \\ nx + \lg nx \end{cases}$	nx<1 nx=1 nx>1	n=1.5	xЄ[0:3] Δx=0.15
8	$y = \begin{cases} \sin x \lg x \\ \cos^2 x \end{cases}$	x>3.5 x≤3.5	-	xЄ[0:3] Δx=0.15
9	$f = \begin{cases} \lg(x+1) \\ \sin^2\sqrt{\lvert mx\rvert} \end{cases}$	x>1 x≤1	m=20.3	xЄ[0:3] Δx=0.15
10	$z = \begin{cases} (\ln^3 x + x^2)/\sqrt{x+t} \\ \sqrt{x+t}+1/x \\ \cos x + t\sin^2 x \end{cases}$	x<0.5 x=0.5 x>0.5	t=2.2	xЄ[0:3] Δx=0.15
11	$s = \begin{cases} \dfrac{m+n}{e^x+\cos x} \\ (m+n)/(x+1) \\ e^x \sin x \end{cases}$	x<2.8 2.8≤x<6 x≥6	m=2.6 n=-0.39	xЄ[0:3] Δx=0.15
12	$y = \begin{cases} m\lg x + \sqrt[3]{\lvert x\rvert} \\ 2m\cos x + 3x^2 \end{cases}$	x>1 x≤1	m=0.9	xЄ[0:3] Δx=0.15

Окончание табл.7

Номер варианта	Функция	Условие	Исходные Данные	Диапазон и шаг изменения аргумента
13	$\omega = \begin{cases} m/i + ni^2 + c \\ i \\ mi + ni^3 \end{cases}$	i<4 4≤i≤6 i≥6	m=2.1 n=1.8 c=-20.5	iЄ[0:3] Δi=0.15
14	$z = \begin{cases} m\sin x(\frac{i^2+1}{n}) \\ \cos(i+1/n) \end{cases}$	$\sin\frac{i^2+1}{n} > 0$ $\sin\frac{i^2+1}{n} < 0$	m=0.3 n=10	iЄ[0:3] Δi=0.15
15	$\omega = \begin{cases} \sqrt{mt^2 + n\sin t + 1} \\ mt + n \\ \sqrt{mt^2 + n\cos t + 1} \end{cases}$	t<0.1 t=0.1 t>0.1	m=2.5 n=0.4	tЄ[0:3] Δt=0.15

Пример выполнения задания

Задание. Вычислить значения функции y=sin(x) на отрезке [a,b] с шагом h тремя способами (с использованием команд while, repeat, for). Для примера: a=1, b=3, h=0,5.

Постановка задачи

а) обозначение переменных

x – переменная величина, аргумент функции;

y – переменная величина, результат;

a,b,h – постоянные величины.

б) классификация переменных

x,y – простые переменные вещественного типа.

в) расчетные формулы записать самостоятельно

Разработка алгоритма

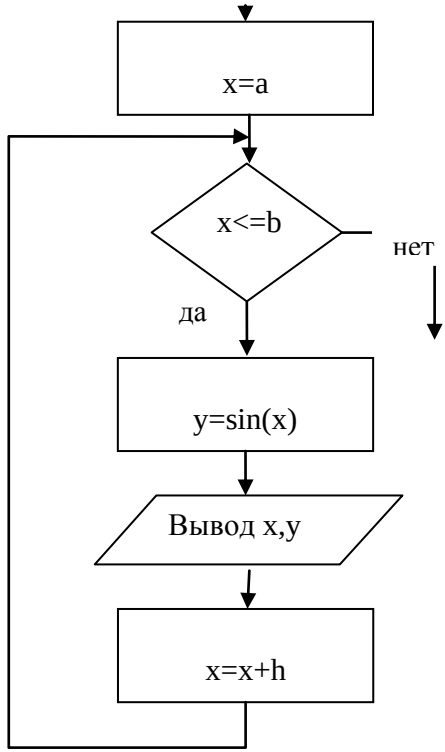


Рис.8. Алгоритм табулирования функции в цикле While.

б) использование команды repeat…until

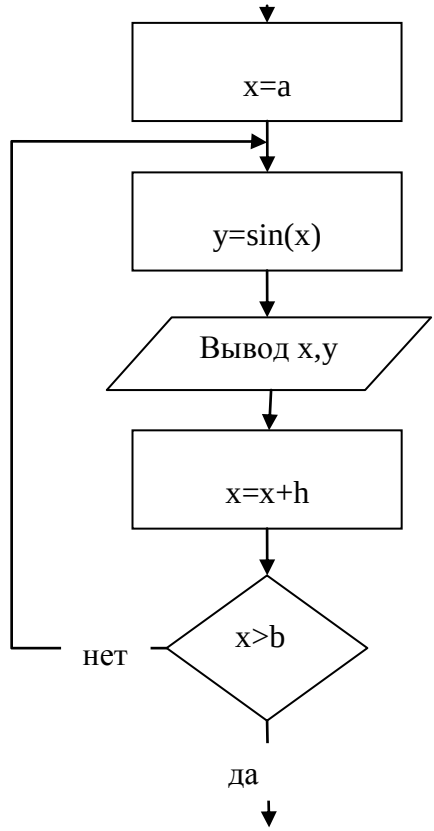


Рис.9. Алгоритм табулирования функции в цикле Repeat.

в) использование команды for…do

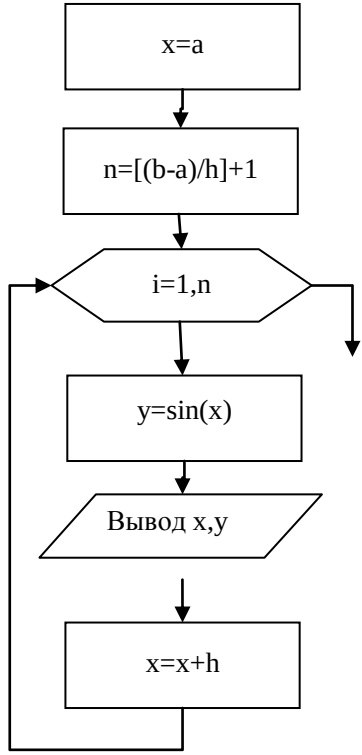


Рис. 10. Алгоритм табулирования функции в цикле For.

Составление программы

а) Цикл while… do

```
PROGRAM zikl2;
Uses CRT;
Const  a=1; b=3; h=0,5;
```

```
Var  x,y: real;
Begin clrscr;
writeln ('_____');
writeln ('|  x          |        y       |');
writeln ('_____');
writeln;
x:=a;
While x<=b do begin
          y:=sin(x);
Writeln ('|',x:6:2,'  |',y:6:2,' |');
x:=x+h;
End;
writeln ('|_____|');
Readln;
End.
```

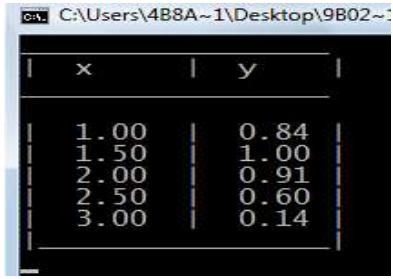
б) Цикл repeat … until

```
PROGRAM zikl;
Uses CRT;
Const  a=1; b=3; h=0,5;
Var  x,y: real;
Begin clrscr;
writeln ('_____');
writeln ('|  x          |        y       |');
writeln ('_____');
writeln;
x:=a;
repeat
          y:=sin(x);
Writeln ('|',x:6:2,'  |',y:6:2,' |');
x:=x+h;
until x>b;
writeln ('|_____|');
Readln;
End.
```

в) цикл for… do

```
PROGRAM zikl;
Uses CRT;
Const  a=1; b=3; h=0,5;
Var  x,y: real;
       i,n : integer;
Begin clrscr;
writeln ('_____');
writeln ('|  x        |       y      |');
writeln ('_____');
writeln;
n:=trunc ((b-a)/h)+1; {trunc – функция преобразования вещественного типа в целый}
x:=a;
for  i:=1 to n do begin
          y:=sin(x);
Writeln ('|',x:6:2,'  |',y:6:2,' |');
x:=x+h;
end;
writeln ('|_____|');
Readln;
End.
```

Тестирование программы



Лабораторная работа №4

Программирование задач прикладного характера

Цель работы – овладение навыками алгоритмизации и программирования вычислительных структур при решении задач прикладного характера.
В данной работе рассматриваются две прикладные задачи: вычисление определенного интеграла двумя методами и вычисление суммы бесконечного ряда.

Краткая теория вычисления определенного интеграла методами трапеций и Симпсона.

Значение определенного интеграла $z = \int_a^b f(x)dx$

представляет собой площадь криволинейной трапеции (рис.11), т.е. площадь, ограниченную сверху графиком подынтегральной функцией f(x), снизу - длиной отрезка интегрирования (b-a), слева и справа – значениями функции в начале и конце отрезка – f(a) и f(b).

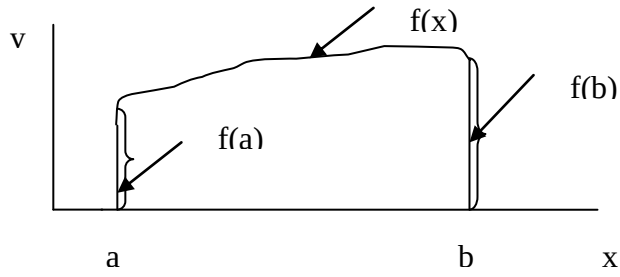


Рис.11. Графическая интерпретация определенного интеграла.

При вычислении определенного интеграла методом трапеций подынтегральная функция f(x) заменяется хордой и пло-

щадь трапеции, ограниченная сверху хордой, является приближенным значением определенного интеграла. Для получения более точного результата, отрезок интегрирования разбивают на несколько частей (n) и на каждом шаге применяют формулу вычисления площади трапеции. При вычислении определенного интеграла методом Симпсона подынтегральная функция заменяется параболой, проходящей через три точки с координатами: [a,f(a); a+h, f(a+h);b,f(b)]. Для получения более точного значения определенного интеграла интервал [a,b] также разбивают на несколько частей, причем количество шагов должно быть четным.

Контроль по теме

1. Геометрический смысл определенного интеграла.
2. Алгоритм метода трапеций. Вывод расчетной формулы.
3. Алгоритм метода Симпсона. Достоинства метода. Вывод расчетной формулы.
4. Сравнительная характеристика методов. Достоинства. Недостатки.
5. Почему в алгоритме метода Симпсона «n» должно быть четным?
6. Объяснить, почему начальная сумма в методе Симпсона S=f(a)-f(b)?
7. Можно ли при программировании метода Симпсона использовать цикл FOR?
8. Объяснить конечное значение и закон изменения параметра цикла в методе Симпсона.

Задание 1

Вычислить значение определенного интеграла $z = \int_a^b f(x)dx$.. В таблице 8 заданы: подынтегральная функция $f(x)$, границы отрезка интегрирования [a, b] Вычисление провести двумя методами: методом трапеций и Симпсона. Сравнить

результаты вычислений. Точность вычислений задать равной 0,001. Начальное количество разбиений отрезка N=2.

Таблица 8

Варианты заданий

Номер варианта	f(x)	a	b
1	$2+x-x^2$	0	1
2	$(1-x)^4$	0.2	1.5
3	cosx+chx	-0.8	0.4
4	$x^{1/3}(1-x)^{2/3}$	0.1	0.6
5	x^3-6x^2+9x+4	0.2	1.5
6	x^3-6x^2+9x+4	2	4
7	$2x^2-x^4$	-2	0.8
8	x^2-3x+2	1	2
9	$x(x-1)^{1/3}$	0.1	1.2
10	xe^{-x}	0.1	1.5
11	$\ln^2 x/x$	6	8
12	$x+1/x$	0.1	1.5
13	arctg $x-1/2*\ln(x^2+1)$	0.15	1.5
14	$\|x\|e^{-\|x-1\|}$	-2	-0.5
15	$\ln^2 x/x$	0.1	1.9

Пример выполнения задания

Задание 1. Вычисление определенного интеграла методом трапеций.

Постановка задачи

а) обозначение переменных:
Исходные данные -a – начало отрезка интегрирования;
b – конец отрезка интегрирования;
n – количество разбиений отрезка;

e – заданная точность;

Промежуточные переменные –

h – шаг интегрирования;
s1 – значение интеграла вычисленное на предыдущем шаге;
i – счетчик шага в цикле;

Конечная переменная (результат) - s – значение определенного интеграла, вычисленное с заданной точностью.

б) классификация переменных

a, b, n, e, h, s1, s – простые переменные вещественного типа;
i, n – простые переменные целого типа

в) расчетные формулы

s1=0;
s=(f(a)+f(b))/2;h=(b-a)/n;{начальные присваивания}
i:=1,n…. s=s+f(a+i*h);{суммирование значений функций на i – том шаге внутри цикла}
s=s*h {конечное значение интеграла}
если $|s1-s|<=e$, то вывод результата s
иначе s1=s;n=2*n и вычисление продолжается, т.е. переход на начальные условия.

В данном примере рассматривается вычисление определенного интеграла:

$$\int_0^1 e^x \cdot dx$$

Разработка алгоритма

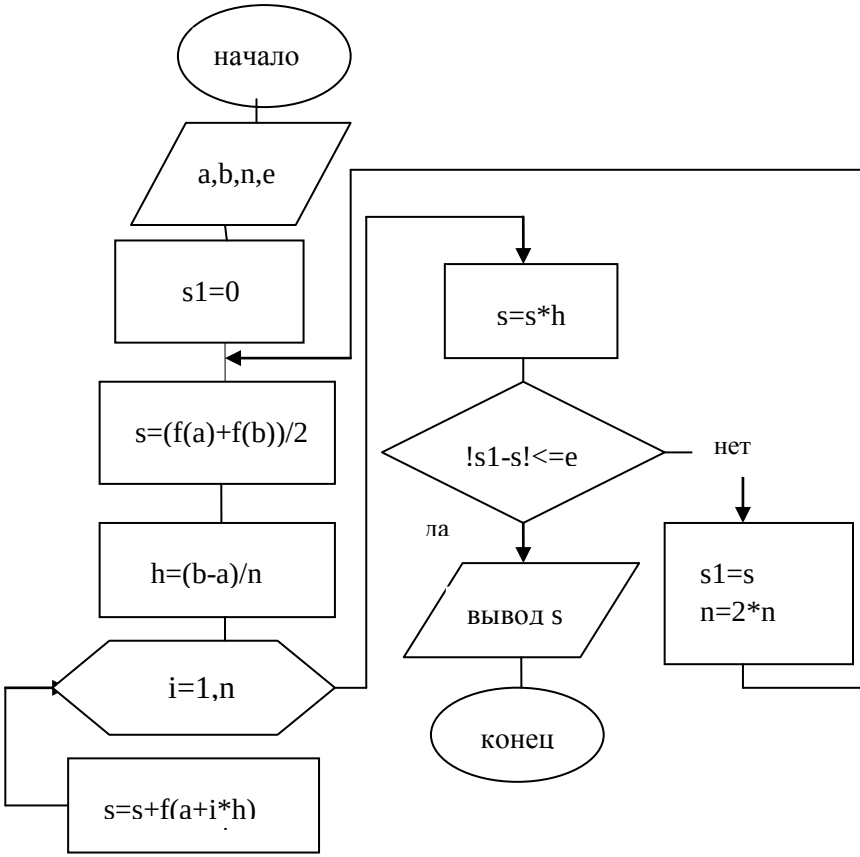
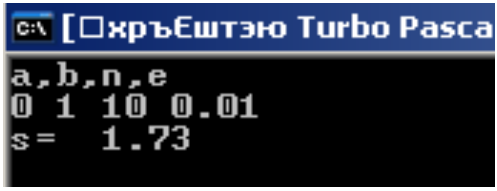


Рис.12. Метод трапеций

Составление программы

```
program int_trap;
var  a,b,h,e,s,s1:real;
i,n:integer;
label 20;
begin
writeln('a,b,n,e');readln(a,b,n,e);
s1:=0;
20: s:=(exp(a)+exp(b))/2;
h:=(b-a)/n;
for i:=1 to n do s:=s+exp(a+i*h);
s:=s*h;
if abs(s1-s)<=e then writeln('s=',s:6:2)
else begin s1:=s;n:=2*n; goto 20 end;
readln end.
```

Тестирование программы



Задание 2.Вычисление определенного интеграла методом Симпсона.

Постановка задачи аналогична постановке по методу трапеций.

Расчетные формулы

s1=0;
s=f(a)-f(b);h=(b-a)/n;{начальные присваивания}
i:=1,[n/2]……s=s+4*f(a+(2*i+1)*h)+2*(f(a+2*i*h));{суммировани
е значений функций внутри цикла}
s=s*h/3 {конечное значение интеграла}
если |s1-s|<=e, то вывод результата s

иначе s1=s;n=2*n и вычисление продолжается, т.е. переход на начальные присваивания.

Разработка алгоритма

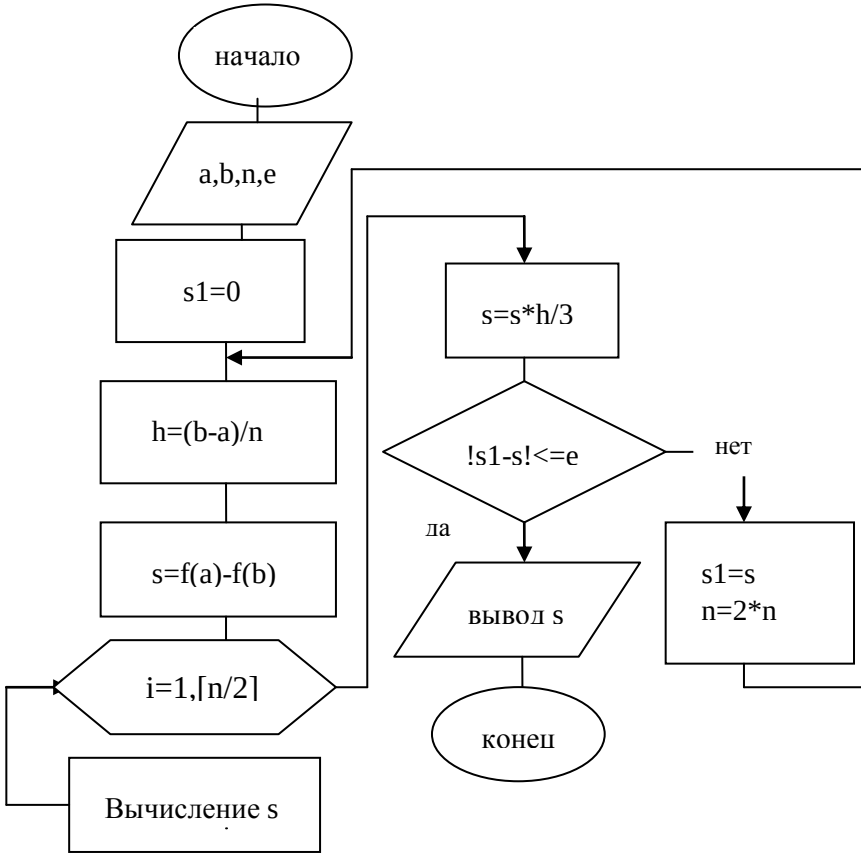
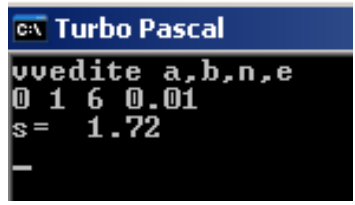


Рис.13. Метод Симпсона

Составление программы

```
program CIMPSON;
        VAR a,b,e,h,s,s1:real;
         i,n:integer;
         label 10;
BEGIN
     writeln('vvedite a,b,n,e');
     readln(a,b,n,e);
     s1:=0;
     10: h:=(b-a)/n;
     s:=exp(a)-exp(b);
for i:=1 to trunc(n/2) do
     s:=s+4*exp(a+(2*i-1)*h)+2*exp(a+2*i*h);
     s:=s*h/3;
     if abs(s-s1)<=0.01 then writeln('s=',s:6:2) else begin
     s1:=s;n:=2*n; goto 10;end;
     readln end.
```

Тестирование программы



Краткая теория по вычислению суммы бесконечного ряда

Вычисление суммы бесконечного ряда осуществляется с заданной точностью, т.е. суммирование членов ряда происходит до тех пор, пока очередной член ряда станет меньше заданной точности. Алгоритм вычисления требует организации цикла с неизвестным числом повторений (While или Repeat) так как заранее неизвестно, при каком члене ряда будет достигнута требуемая точность. Вычислять текущий (очередной) член ряда по

общей формуле с точки зрения затрат времени нецелесообразно, поэтому его вычисляют по рекуррентной формуле:

$$A_i = Q * A_{i-1},$$

где A_i - вычисляемый текущий член ряда; Q – приращение (множитель); A_{i-1} - предыдущий член ряда. Другими словами, каждый последующий член ряда получается из предыдущего домножением на приращение.

Контроль по теме

1. Вывод формулы приращения члена ряда.
2. Организация алгоритма суммирования членов ряда в структуре цикла While
3. Организация алгоритма суммирования членов ряда в структуре цикла Repeat
4. Возможно ли вычисление суммы ряда в цикле For?
5. Практическая значимость задачи.

Задание 2. Вычислить значение суммы членов бесконечного ряда с заданной точностью (e=0,001). На печать вывести значение суммы и число членов ряда, вошедших в сумму.

Таблица 9

Варианты заданий

Вариант задания	Сумма ряда	Значение X
I	$s = -\frac{(2x)^2}{2} + \frac{(2x)^4}{24} + .. + (-1)^n \frac{(2x)^{2n}}{(2n)!} + ..$	0,20

Продолжение табл. 9

Вариант задания	Сумма ряда	Значение X
2	$s = x - \frac{x^3}{3} + .. + (-1)^n \frac{x^{2n-1}}{2n-1} + ..$	0,10
3	$s = x^3/5 - x^5/17 + .. + (-1)^{n+1} \frac{x^{2n+1}}{4n^2+1} + ..$	0,15
4	$s = 1 + \cos\frac{\pi}{4} \cdot \frac{x}{1!} + .. + \frac{\cos n\frac{\pi}{4}}{n!} x^n + ..$	0,12
5	$chx \approx s = 1 + \frac{x^2}{2!} + .. + \frac{x^{2n}}{(2n)!} + ..$	0,70
6	$\pi \approx s = 4\left(1 - 1/3 + .. + (-1)^n \frac{1}{2n-1} + ..\right)$	-
7	$arctgx \approx s = 1/x - 1/(3x^3) + .. + (-1)^n 1/((2n+1)x^2$	1,5
8	$\cos(\pi/6) \approx s = 1 - \frac{\left(\frac{\pi}{6}\right)^2}{2!} + \frac{\left(\frac{\pi}{6}\right)^4}{4!} - .. + (-1)^n \frac{\left(\frac{\pi}{6}\right)^2}{(2n)}$	-
9	$ch \approx x + \frac{x^3}{3!} + .. + \frac{x^{2n+1}}{(2n+1)!} + ..$	1,7
10	$\sin\frac{\pi}{3} \approx s = \frac{\pi}{3} - \frac{\left(\frac{\pi}{3}\right)^3}{3!} + .. + (-1)^n \frac{\left(\frac{\pi}{3}\right)^{2n+1}}{(2n+1)!}$	-

Окончание табл.9

Вариант задания	Сумма ряда	Значение X
11	$s = 1 + \frac{x^2}{2!} + .. + (-1)^n \frac{2n-1}{(2n)!} x^{2n} + ..$	0,75
12	$s = \frac{2}{3}\sin 2x - \frac{3}{8}\sin 3x + .. + (-1)^n \frac{n}{n^2-1}\sin nx$	0,62
13	$s = 1 + \frac{\cos x}{1!} + .. + \frac{\cos nx}{n!} + ..$	0,20
14	$s = \frac{x^3}{3} - \frac{x^5}{15} + .. + (-1)^n \frac{x^{2n+1}}{4n^2-1} + ..$	0,30
15	$s = \frac{x\cos\frac{\pi}{3}}{1} + \frac{x^2\cos 2\frac{\pi}{3}}{2} + .. + \frac{x^n \cos n\frac{\pi}{3}}{n}$	0,25

Пример выполнения задания

Вычислить значение суммы членов бесконечного ряда $s = 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^n}{n!} + \cdots$ с точностью (e=0,001). На печать вывести значение суммы и число членов ряда, вошедших в сумму.

Постановка задачи

а) обозначение переменных:
Исходные данные -
x – значение аргумента;
e – заданная точность;
Промежуточные переменные – A – текущий член ряда;
n – номер члена ряда;

Конечная переменная (результат)-

s – значение суммы ряда, вычисленное с заданной точностью.

б) классификация переменных

x, e, h, A, s – простые переменные вещественного типа;

n – простая переменная целого типа

в) расчетные формулы

Предлагаются расчетные формулы для цикла Repeat

S=1+x; n=2; A=x{начальные присваивания}

A=A*x/n; s=s+A; n=n+1

Если A<=e →да →вывод s,n, конец

↓

нет

Разработка алгоритма

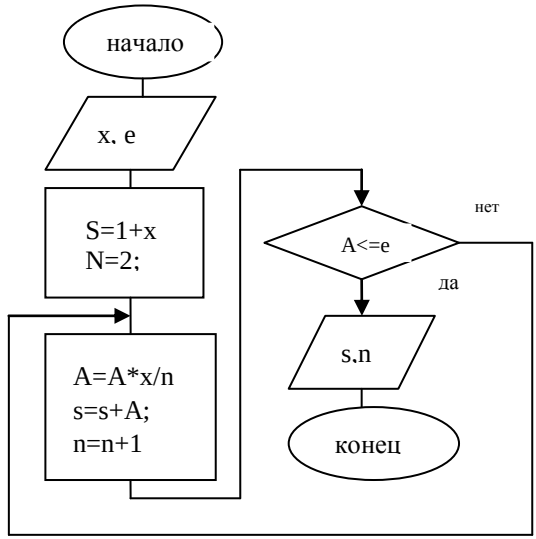
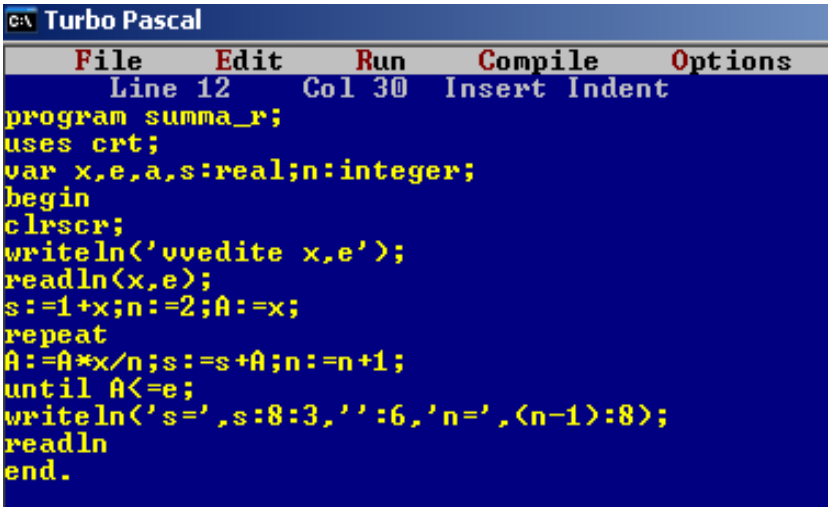
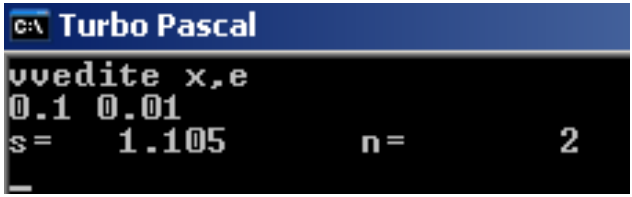


Рис.14. Вычисление суммы бесконечного ряда

Составление программы



Тестирование программы



Лабораторная работа №5

Обработка одномерных массивов

Цель работы – овладение практическими навыками работы с массивами, особенностями их ввода и вывода, приобретение дальнейших навыков по организации программ циклической структуры с использованием приемов программирования.

Краткая теория по обработке одномерных массивов

Одномерный массив – это последовательность величин (числовых, строковых, символьных), обозначенная одним идентификатором. Элементы массивов обозначаются индексными переменными. В одномерных массивах индекс – это порядковый номер величины в массиве. Для решения задач прикладного характера необходимо знать базовые алгоритмы обработки одномерных массивов. К основным структурам относят следующие:

Определение максимального (минимального) элемента;
Суммирование элементов массива по условию;
Определение количества по условию;
Формирование массива из исходного;
Выход из цикла до его окончания;
Циклический сдвиг в массиве и т.д.

Для решения задач на базе этих структур можно использовать любой из трех изученных ранее циклов.

Контроль по теме

1. Ввод – вывод одномерных массивов, описание их в программе.
2. Типовые алгоритмы обработки одномерных массивов.
3. Записать любой алгоритм обработки массива в виде расчетных формул, блок – схемы и программы
4. Представить любой алгоритм в любом из трех циклов.

Задание к работе

1. Обработать массив в соответствии с вариантом задания, указанного в таблице 10.
2. Проверить правильность выполнения программы с помощью тестового варианта.

Таблица 10

Варианты заданий

Вариант задания	Массив	Условие задачи	Ограничения
1	X(100)	Вычислить сумму и количество элементов массива X, находящихся в интервале [0,1]	$0 \leqslant x_i \leqslant 1$
2	A (80)	Вычислить среднее арифметическое значение элемента массива A	$a_i > 0$
3	X(70)	Переписать элементы массива X, находящиеся в интервале [-1,0], в массив Y	$-1 \leqslant x_i \leqslant 1$
4	B (50)	Определить максимальный элемент массива B и его порядковый номер	$в_i > 0$
5	C (40)	Вычислить минимальный элемент массива C и его номер	$c_i < 0$
6	D(80)	Найти максимальный и минимальный элементы массива D и поменять их местами	—
7	Y(20)	Вычислить среднее геометрическое элемента массива Y	$y_i > 0$

Окончание табл.10

Вариант задания	Массив	Условие задачи	Ограничения
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 дробных элементов массива X	$x_i > 0$, $N \leqslant 30$
11	A(N)	Найти сумму, произведение и количество нечетных элементов массива A	$a_i > 0$, $N \leqslant 50$
12	X(N)	Переписать в массив Y под- ряд положительные элементы массива X	$x_i > 0$, $N \leqslant 40$
13	X(N)	Переписать подряд в массив Y положительные и в массив Z отрицательные элементы массива X	$N \leqslant 40$
14	B(K)	Определить первый элемент массива B, кратный 5 и его порядковый номер	$в_i < 0$, $K \leqslant 40$
15	C(K)	Определить произведение элементов массива C до первого четного элемента	c_1 - нечетный $K \leqslant 20$

Пример выполнения задания.

Найти первый отрицательный элемент в массиве чисел A [1..n] и определить его номер, если отрицательных элементов нет, выдать сообщение об этом..

Постановка задачи

Обозначение переменных:

A [1..n] – исходный массив, вещественного типа
i – номер элемента в массиве, целого типа
n – количество элементов в исходном массиве, целого типа
nom – номер первого отрицательного элемента в массиве, целого типа
A[nom] – первый отрицательный элемент массива A, вещественного типа

Расчетные формулы

nom = 0
i = 1
    если A[i]<0 то nom=i, выход из цикла
    иначе i = i+1
i <=n

Разработка алоритма

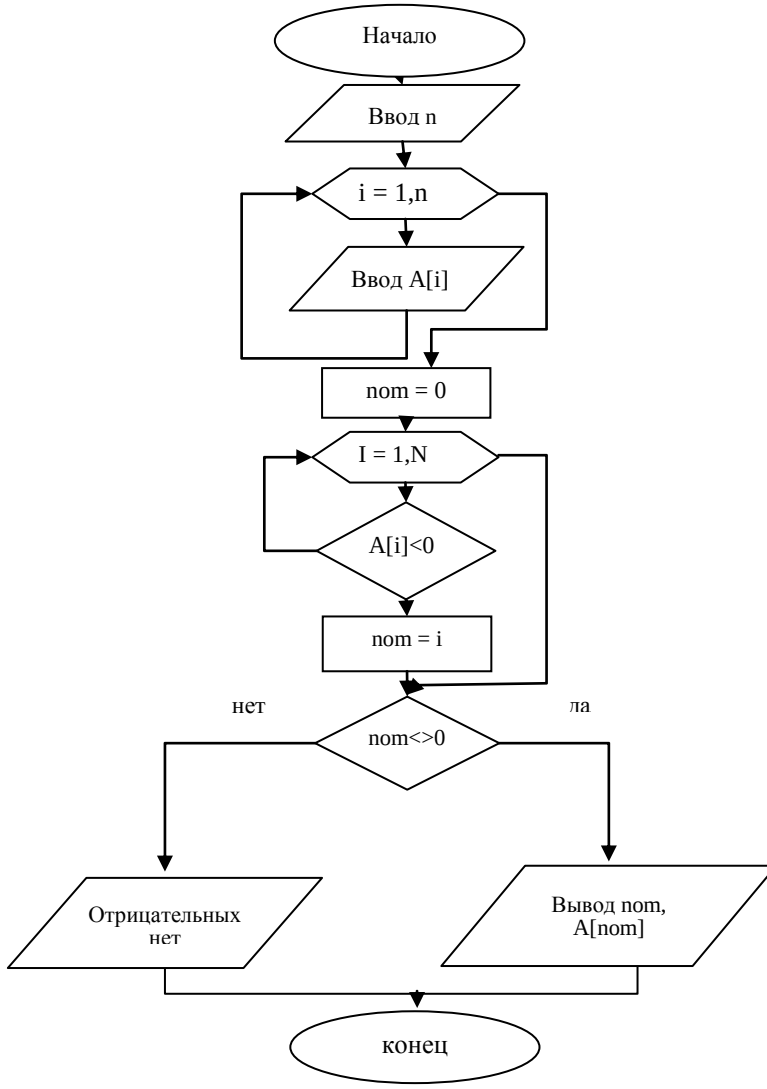
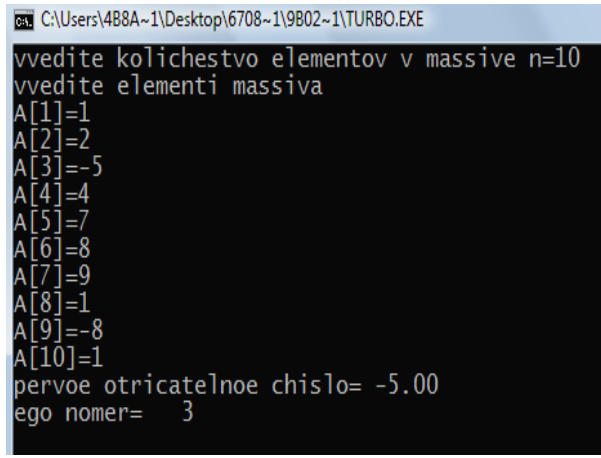


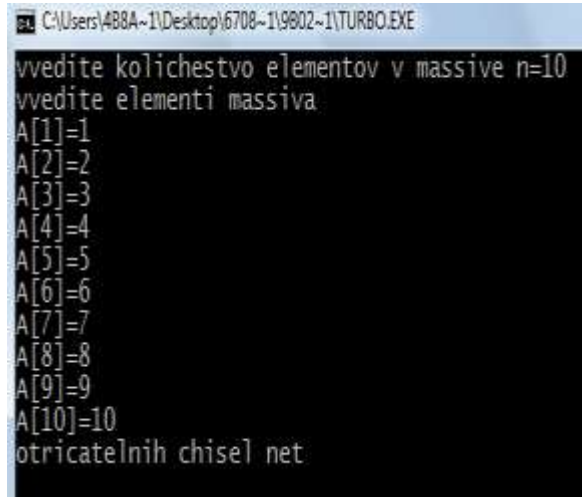
Рис. 15. Алгоритм обработки одномерного массива.

Составление программы

```
PROGRAM  massiv;
USES CRT;
LABEL 1;
VAR  A: array [1..100] of real;
        i, n, nom : integer;
BEGIN ClrScr;
WRITE (‘ введите количество элементов в массиве, n’);
READLN (n);
WRITELN (‘введите элементы массива’);
FOR i:=1 TO n DO
        BEGIN
                WRITE (‘A[‘,i,’]’);
                READLN (A[i]);
        END;
nom:=0;
FOR i:=1 TO n DO
        IF A[i]<0 THEN BEGIN
                nom:=i;
                goto 1;
        END;
1: IF nom<>0 THEN BEGIN
        WRITELN (‘Первое отрицательное число=’, A[nom]:6:2);
        WRITELN (‘Его номер=’, nom:4);
END
        ELSE
                WRITELN (‘Отрицательных чисел нет’);
READLN;
END.
```

Тестирование программы.





Лабораторная работа №6

Вложенные циклы. Обработка матриц.

Цель работы – овладение приемами обработки вложенных циклов на примерах поиска элементов матрицы.

Краткая теория по вложенным циклам

Вложенные циклы (цикл в цикле)– это структура, в которой один цикл (внутренний) является телом (областью действия) другого цикла (внешнего). Структуру **Цикл в цикле** схематично можно представить слебдующим образом (рис.16.):

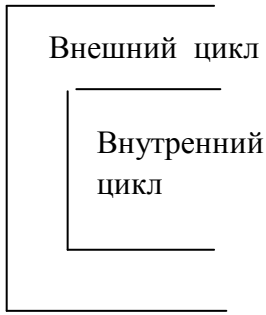


Рис. 16. Схематичное изображение структуры Цикл в цикле.

Следует помнить правила организации циклов:

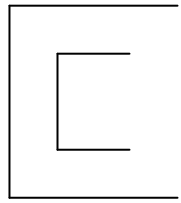
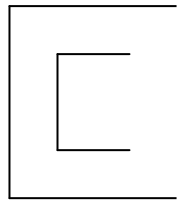
1) И внешний, и внутренний цикл работают по правилам выбранных простых циклов
2) Не допускается пересечение циклов, т.е. внутренний цикл не должен выходить за пределы внешнего, другими словами, параметр внешнего цикла изменяется после окончания внутреннего цикла

3) Параметры циклов (переменные, управляющие их работой) должны быть обозначены разными идентификаторами.

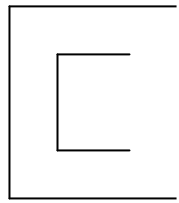
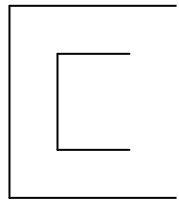
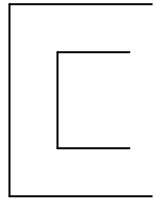
Так как в алгоритмическом языке Паскаль существует 3 структуры цикла (FOR,Repeat,While), то возможно девять их сочетаний для реализации структуры **Вложенный цикл.** В таблице 11 приведены семь вариантов.

Таблица 11

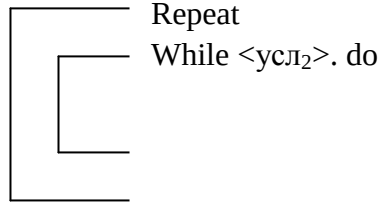
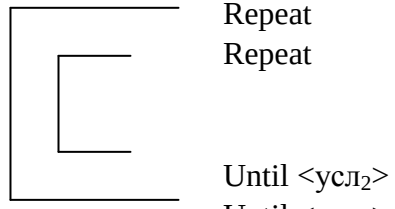
Структуры вложенных циклов

Структура цикла		Схематичное изображение
внешнего	внутреннего	
For	For	 For ПЦ$_1$…. For ПЦ$_{2...}$
For	While	 For ПЦ$_1$…. While <усл$_2$> do

Продолжение табл.11

Структура цикла		Схематичное изображение
внешнего	внутреннего	
While	For	 While <усл$_1$> For ПЦ$_2$
While	While	 While <усл$_1$>do While <усл$_2$>do
While	Repeat	 While <усл$_1$> do Repeat Until <усл$_2$>

Окончание табл.11

Структура цикла		Схематичное изображение
внешнего	внутреннего	
Repeat	While	
Repeat	Repeat	
Примечание: индекс “1” в схематичном изображении относится к внешнему циклу, индекс “2” – к внутреннему		

Структура Вложенный цикл используется для решения многих задач, в частности, для всевозможных матричных операций, причем вложенность циклов неограниченна.

Для обработки матриц удобно использовать структуру, в которой и внешний, и внутренний циклы организованы со счетчиками (параметрами), т.е. цикл FOR. В некоторых случаях удобнее использовать другие комбинации циклов. Понятно, что выбор структуры зависит от желания пользователя, но в курсе обучения студенты должны демонстрировать свободное владение материалом по данной теме, т. е. по просьбе преподавателя уметь представлять задачу в различных вариантах циклов. Множество

задач по обработке матриц условно можно разделить на три группы:

1) поиск элементов в матрице, в котором просмотр элементов производится либо по столбцам, либо по строкам
2) поиск элементов по строкам
3) поиск элементов по столбцам

Рассмотрим 3 варианта задач в качестве примера выполнения индивидуальных заданий

Пример выполнения задания

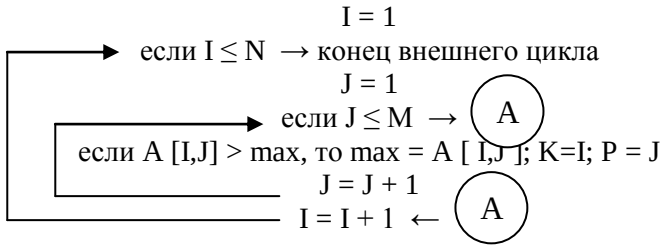
Задача 1. Определите максимальный элемент матрицы и его местонахождение.

Это задание относится к первой группе обработки матриц, т.е. поиск максимального элемента можно производить либо по строкам, либо по столбцам.

Постановка задачи

а) обозначение переменных
N – количество сток в матрице;
M – количество столбцов в матрице;
I – номер строки;
J – номер столбца;
A [I,J] – элемент матрицы стоящий в I – той строке и J – столбце;
MAX – максимальный элемент матрицы;
K – номер строки, в которой находится максимальный элемент;
P – номер столбца, в котором находится максимальный элемент
б) классификация переменных по типам
N,M,J,K,P – простые переменные целого типа;
MAX – переменная вещественного (или целого)типа;
A [I,J] – индексная переменная вещественного (или целого)типа
в) расчетные формулы
В расчетных формулах схематично показывается выполнение структуры вложенный цикл по действиям. Выберем обработку матрицы по строкам, следовательно, параметром внешнего цик-

ла будет переменная «I», а внутреннего цикла – переменная «J».
Max = A[1,1], K = 1, P = 1



Разработка алгоритма

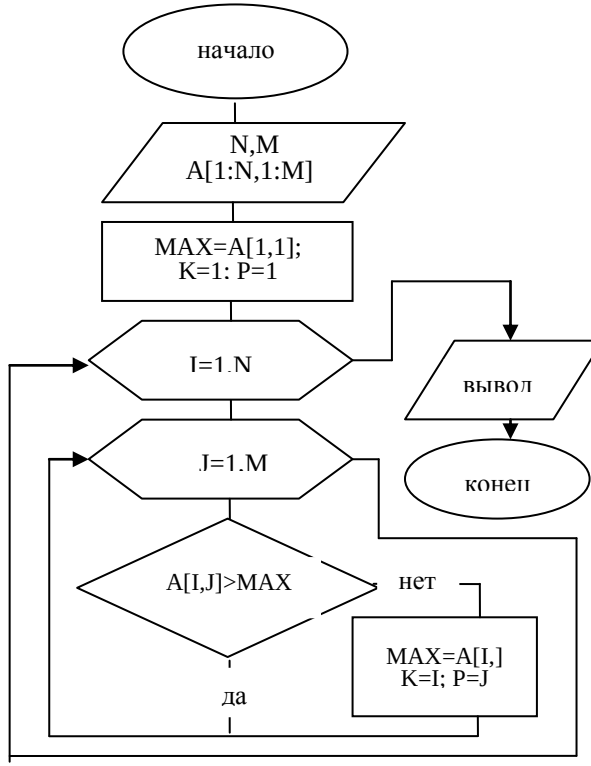
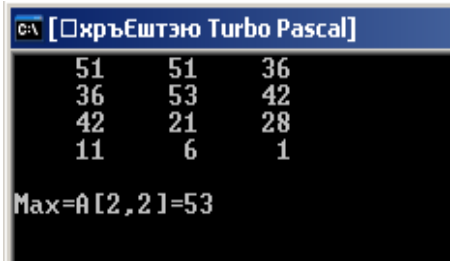


Рис. 17. Поиск максимального элемента в матрице.

Сотавление программы

```
PROGRAM  maximum;
Var I,J,N,M,K,P: integer;
Max: Real;
A: Array [1..50,1..50]of Real;
BEGIN
WRIELN (`введите кол- во строк`);
READLN (N);
WRITELN (`введите кол- во столбцов`);
READLN (M);
{Ввод матрицы}
FOR I: = 1 TO N do
FOR J: =1 TO M do
BEGIN
WRITLEN (‘A[‘,I,’,’,J,’J = ‘);
READLN (A[I,J]);
END;
Max: = A[1,1]; K: = 1; P: = 1;
FOR I: = 1 TO N do
FOR J: = 1 TO M do
IF A [I,J] > MAX THEN
BEGIN
Max: = A [I,J]; K: = I; J: = P;
END;
{вывод матрицы}
FOR I: = 1 TO N do  BEGIN
FOR J: =1 TO M do
WRITE (A [I,J]:6:2);
WRITELN;   END;
{вывод результата}
WriteLn  (`max = `, A[K,P]:6:2);
READLN; END.
```

Тестирование программы



Задание 2 Определить максимальный элемент матрицы в каждой строке

При рассмотрении этой задачи, учитывая подробное описание задачи 1, представим фрагменты блок – схемы и программы

Фрагмент блок – схемы

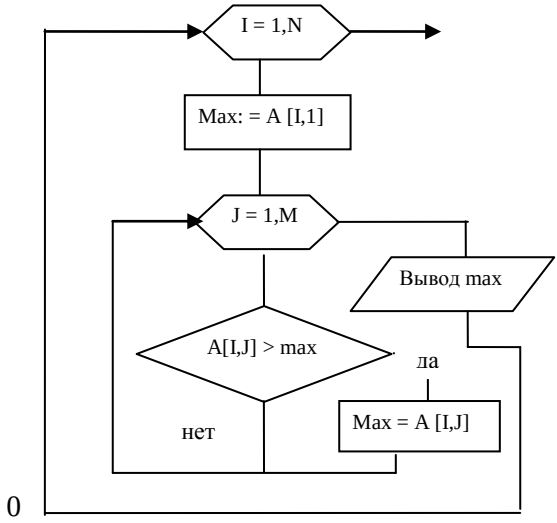


Рис. 18. Поиск максимального элемента в каждой строке матрицы.

Фрагмент программы

```
                    ……
 FOR I: = 1 TO N do
    BEGIN
     Max: = A [I,1];
      FOR J: = 1 TO M do
      IF A [I,J] > max
      THEN max: = A[I,J]:
WRITELN (‘max[‘,I,’]=’, max:6);
END;
                    ……
```

Примечание: в данной задаче матрица описана типом INTEGER

Тестирование

Результаты обработки матрицы по строкам представлены в окне вывода

Задача 3. Определить максимальный элемент в каждом столбце матрицы

Эта задача также рассматривается фрагментарно.

Фрагмент блок - схемы

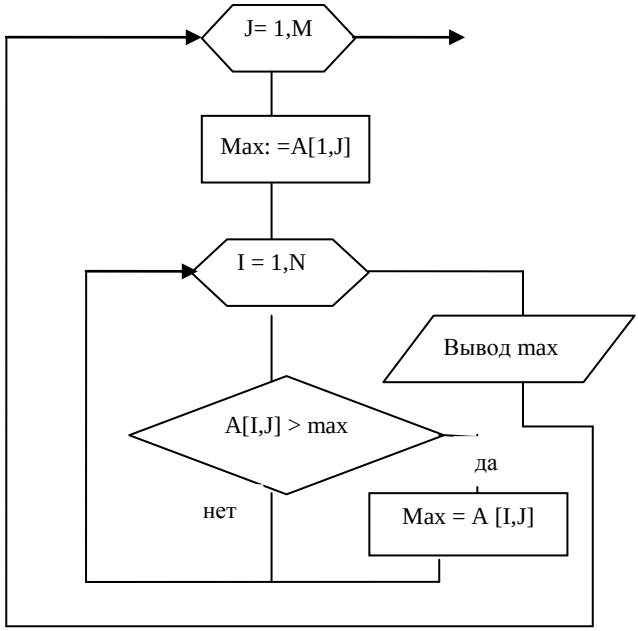


Рис. 19. Поиск максимального элемента в каждом столбце матрицы.

Фрагмент программы

```
                ……..
FOR J: = 1 TO M DO
   BEGIN
      Max: = A[1,J];
      FOR I: =1 TO N DO
IF A[I,J] > max THEN
      Max: = A [I,J];GOTOXY(12*J,I+2);
WRITE (‘max[‘,j,’]=’,max,’’:6);
END;
```

Примечание: в данной задаче матрица описана типом INTEGER

Тестирование программы

Результаты выполнения программы представлены в окне вывода

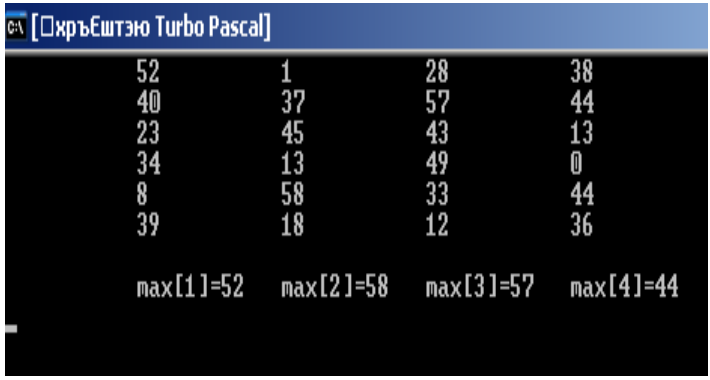


Таблица 12

Варианты заданий

№ варианта	Имя матрицы и размеры	Действия	Условия и ограничения
1	A (10, 15)	Вычислить сумму и число положительных элементов каждого столбца матрицы. Результаты напечатать в виде двух строк	$a_{ij} > 0$
2	A (N, M)	Вычислить сумму и число отрицательных элементов каждой строки матрицы. Результаты напечатать в виде двух столбцов	N≥20 M≤15

Продолжение табл.12

№ варианта	Имя матрицы и размеры	Действия	Условия и ограничения
3	B (N, N)	Вычислить сумму и число элементов матрицы, находящихся под главной диагональю и на ней	N<12
4	C (N, N)	Вычислить сумму и число положительных элементов матрицы, находящихся над главной диагональю	C_{ij}> 0 N≤12
5	D (K, K)	Записать на место отрицательных элементов матрицы нули и вывести ее на печать в общепринятом виде	K≤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≤20 M≤10
8	F(10,10)	Транспонировать матрицу и вывести на печать элементы главной диагонали и диагонали, расположенной под главной. Результаты разместить в двух строках	

Продолжение табл.12

№ варианта	Имя матрицы и размеры	Действия	Условия и ограничения
9	N(10, 10)	Для целочисленной матрицы найти для каждой строки число элементов, кратных пяти, и наибольший из полученных результатов	$n_{ij}/5*5=n_{ij}$
10	V(15, 10)	Упорядочить по возрастанию элементы каждого столбца матрицы. Отпечатать полученную матрицу 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)	Ввести исходные данные в первые 24 строки и первые 7 столбцов. Вычислить 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	

Окончание табл. 12

№ варианта	Имя матрицы и размеры	Действия	Условия и ограничения
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	

Лабораторная работа №7

Программирование с использованием подпрограмм. Процедуры и функции.

Краткая теория

При разработке сложных программ в процессе нисходящего проектирования фрагменты (модули) задачи представляются в виде подпрограмм. Использование подпрограмм позволяет создать четкую, последовательную, читабельную структуру программ.

На алгоритмическом языке Pascal возможно создание двух видов подпрограмм – процедур и функций. Программа, содержащая подпрограмму, состоит из двух частей – основной и подпрограммы. Подпрограмма оформляется в разделе описаний основной программы, причем структура её подобна основной программе.

Структура подпрограммы- процедуры.

PROCEDURE <имя> (список формальных параметров);
<Описание локальных переменных>
Begin
<**Тело процедуры**>
End;

Структура подпрограммы – функции.

FUNCTION <имя> (список формальных параметров) : <тип >;
<описание локальных переменных>
Begin
<тело функции>
End;

В процедурах и функциях допускается вложенность структур. Кроме того, подпрограммы могут быть рекурсивными, то есть обращаться к самим себе.

Обращение к подпрограмме осуществляется из основной программы. При обращении формальные параметры “заменяются” фактическими. Обращение к процедуре осуществляется оператором процедуры:

<имя процедуры> (список фактических параметров).

Обращение к подпрограмме – функции осуществляется из любого оператора основной программы.

Целесообразность использования подпрограмм заключается в возможности многократного обращения к ней при различных значениях фактических параметров.

Отличие процедур от подпрограмм – функций заключается в том, что последние могут выдавать только один результат, а процедура – несколько.

Важным моментом в понимании работы программы является связь между формальными и фактическими параметрами. При обращении к подпрограмме фактические параметры должны соответствовать формальным по количеству, типу и порядку следования.

Следует четко различать входные и выходные данные (результаты) при обращении к подпрограммам. Входные параметры могут быть константами, переменными, выражениями. Перед входными параметрами – переменными необходимо ставить слово **VAR**. Выходные параметры могут быть только переменными, поэтому в списке формальных параметров (в заголовке подпрограммы) перед ними обязательно указывается ключевое слово **VAR.**

Задание 1.

Разработать программу с использованием подпрограммы - функции

Таблица 13

Варианты заданий

Вариант задания	Условия задачи	Примечания
1	Вычислить большие корни квадратных уравнений: x^2-ax+b=0 cy^2-dx-f=0	Все корни действительные
2	Подсчитать число точек, находящихся внутри круга радиусом R с центром в начале координат; координаты точек заданы массивами X(10), Y(10)	Расстояние точки от начала координат вычислять в подпрограмме
3	Определить периметры треугольников, заданных координатами их вершин XA(5), XB(5), XC(5) YA(5), YB(5), YC(5)	Длину стороны треугольников вычислять в подпрограмме
4	Подсчитать число точек, находящихся внутри круга радиусом R с центром в точке с координатами (1,1); координаты точек заданы массивами X(8), Y(8)	Расстояние точки от центра круга определять в подпрограмме
5	Вычислить $z = \frac{\nu_1 + \nu_2 + \nu_3}{3}$, где ν_1, ν_2, ν_3 – объемы шаров с радиусами R_1, R_2, R_3 соответственно	ν_i вычислить в подпрограмме
6	Вычислить суммы положительных элементов массивов X(N), Y(M), Z(K)	**N<=20** M<=30 K<=10
7	Вычислить среднее арифметическое положительных элементов для массивов A(N1), B(N2), C(N3)	Размерность массивов различна

Окончание табл. 13

Вариант задания	Условия задачи	Примечания
8	Подсчитать количество элементов матриц X(10,5) и Y(4,8), удовлетворяющих условиям $0<=X_{ij}<=1$ и $0<=Y_{ij}<=1$	количество элементов определять в подпрограмме
9	Вычислить суммы положительных элементов каждой строки для матриц A(10,12) и B(5,7)	суммы положительных элементов определять в подпрограмме
10	Вычислить $z = \frac{x_{m1} + x_{m2}}{2}$, где x_{m1} и x_{m2} – наименьшие элементы массивов X1(7), X2(12)	наименьшие элементы определять в подпрограмме
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(12); s_2 – сумма отрицательных элементов массива Y(10)	Обе суммы вычислять в одной подпрограмме
13	Подсчитать число нулевых элементов в матрицах A(N,M) и B(M,N)	M<=20 N<=20
14	Вычислить суммы элементов нижних треугольных матриц для матриц A(15,15) и B(10,10)	
15	Определить число положительных элементов до первого отрицательного в массивах X(20), Y(15), Z(10)	

Задание 2.

Разработать программу с использованием подпрограммы - процедуры

Таблица 14

Варианты заданий

Вариант задания	Условия задачи	Примечания
1	Вычислить $z=\frac{s_1+s_2}{k_1k_2}$, где s_1 и k_1 – сумма и количество положительных элементов массива X(N); s_2 и k_2 – сумма и количество положительных элементов массива Y(M)	M<=100 N<=100
2	Вычислить $z=\frac{e^{s1}+e^{s2}}{k1\cdot k2}$, где s1 и k1 – сумма и количество положительных элементов массива X(20); s2 и k2 – сумма и количество отрицательных элементов массива Y(10)	Обе сумм вычислять в одной подпрограмме
3	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
4	Найти наибольшие элементы и их порядковые номера в массивах X(N) и Y(M)	N<=20 M<=15
5	Переписать положительные элементы массива X(20) и Y(15) в массив Z подряд	Запись в массив Z осуществлять в подпрограмме

Окончание табл.14

Вариант задания	Условия задачи	Примечания
6	Найти наименьшие элементы, номера строк и столбцов, в которых они расположены, для матриц A(10,15) и B(15,12)	
7	Вычислить $z = \frac{\sum_{i=1}^{10} \sin x_i + \sum_{i=1}^{15} \cos y_i}{\sum_{i=1}^{20} \lvert x_i \rvert}$, где x_i и y_i заданы массивами	Все суммы вычислять в одной подпрограмме
8	Вычислить $z = \frac{x_{max} - y_{min}}{2}$, где x_{max} – максимальный элемент массива X(20) y_{min} – минимальный элемент массива Y(20)	x_{max} и y_{min} вычислять в одной подпрограмме
9	Найти средние значения и стандартные отклонения для элементов массивов X(N) и Y(M)	**N<=**40 M<=60
10	Вычислить суммы и количества элементов, находящихся в интервале от A до B для матриц X(10,8) и Y(10,12)	

Пример выполнения задания 1.

Программирование с использованием подпрограммы – функции.

Задача. Вычислите значение выражения $P = \frac{A! + B!}{(A+B)!}$ используя подпрограмму – функцию.

Постановка задачи

а) обозначение переменных

A, B – исходные переменные.
P – результат.

б) классификация переменных.

A, B - простые переменные целого типа; **P –** простая переменная вещественного типа.

в) расчетные формулы.

Так как вычисление факториала будет производиться в подпрограмме – функции необходимо ввести формальный параметр, который при обращении к подпрограмме будет заменяться фактическими (**A, B, A+B**). Формальный параметр обозначим переменной **F**.

A, B – входные фактические параметры (переменные).
(A+B) – входной фактический параметр (выражение).
Запишем расчетные формулы вычисления факториала. Это – алгоритм произведения.

```
S = 1
I = 1
► Если I<=F, то S=S*I
└─── I=I+1
```

Разработка алгоритма

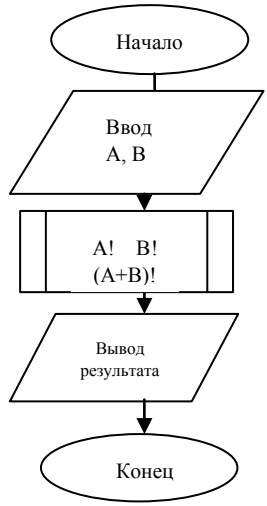


Рис.20. Вычисление факториала в подпрограмме – функции.

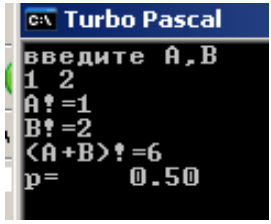
Составление программы.

```
PROGRAM FACTORIAL;
VAR A, B : INTEGER;
P : REAL;
{Описание подпрограммы – функции}
FUNCTION FACT (F : Integer) : Integer;
VAR
I, S : Integer;
BEGIN
S := 1;
FOR I := 1 to F do
S := S*I;
FACT := S;
END;
BEGIN
WRITELN (‘введите A,B’);
READLN (A,B);
```

```
P := (FACT(A) + FACT(B))/FACT(A + B);
WRITELN('A!=',FACT(A)); WRITELN('B!=',FACT(B));
WRITELN('(A+B)!=',FACT(A+B));
WRITELN ('P = ', P:6:2);
READLN;
END.
```

Обращение к подпрограмме – функции происходит из оператора присваивания три раза. Кроме того, для вывода промежуточных значений (факториалов) с целью облегчения проверки конечного результата, сделаны обращения к подпрограмме – функции из операторов вывода, а точнее из встроенных процедур вывода.
Результаты выполнения программы представлены ниже.

Тестирование программы



Пример выполнения задания 2.

Программирование с использованием подпрограммы – процедуры.

Задача. Вычислить значение выражения $Y = \dfrac{S_1 + S_2 + S_3}{K_1 + K_2 + K_3}$ используя подпрограмму процедуру.

S_1 - сумма нечетных элементов массива A[1..N]

S_2 - сумма нечетных элементов массива B[1..M]

S_3 - сумма нечетных элементов массива C[1..P]

K_1, K_2, K_3 - количество нечетных элементов в этих массивах, соответственно.

Постановка задачи.

а) обозначение переменных.
Исходные данные – три массива различной размерности.
A[1..N] – массив, состоящий из N элементов.
B[1..M] - массив, состоящий из M элементов.
C[1..P] - массив, состоящий из P элементов.
I – порядковый номер элементов массива.
A[I], B[I], C[I] – элементы массивов, стоящие на I-том месте.
Результат:
Y – конечная переменная

S_1 - сумма нечетных элементов массива A[1..N]

K_1 - количество нечетных элементов массива A[1..N]

S_2 - сумма нечетных элементов массива B[1..M]

K_2 - количество нечетных элементов массива B[1..M]

S_3 - сумма нечетных элементов массива C[1..P]

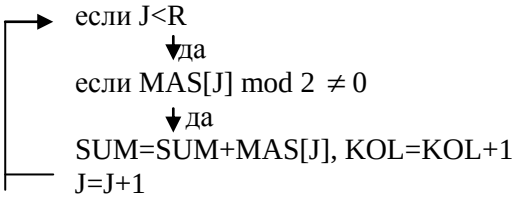
K_3 - количество нечетных элементов массива C[1..P].

б) расчетные формулы

Вычисление суммы и количества будет производиться в подпрограмме – процедуре.
Введем формальные параметры, которые при обращении к процедуре будут заменяться фактическими.
MAS[1..R] – формальный массив, состоящий из R – элементов
SUM – сумма нечетных элементов формального массива MAS[1..R]
KOL – количество нечетных элементов формального массива MAS[1..R]

Алгоритм вычисления SUM и KOL в расчетных формулах:

SUM = 0; KOL = 0;
J = 1;



Разработка алгоритма

На этом этапе нужно определиться с количеством процедур. Возможны такие варианты:

1) Ввод, обработку, вывод массивов организовать в одной процедуре. Это самый компактный вид программы, но, чтобы вывод не прерывался вводом, необходимо массивы формировать датчиком случайных чисел.
2) Ввод, вывод массивов организовать в основной программе, а обработку оформить процедурой.
3) Ввод и обработку, организовать в двух процедурах, а вывод в основной программе.
4) Ввод, обработку и вывод организовать в 3-х процедурах

Рассмотрим второй вариант. Выбор его обусловлен с целью организации единственной процедуры для того, чтобы на ней понять взаимодействие формальных и фактических параметров. С точки зрения компактности, структурности и экономии памяти целесообразно выбрать 1-й или 4-й варианты.

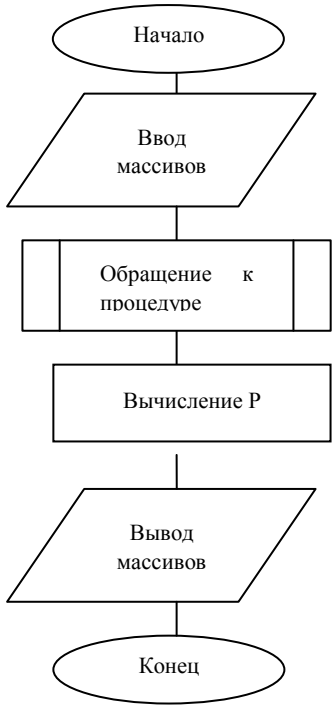


Рис. 21. Обработка одномерных массивов в подпрограмме - процедуре.

Составление программы.

Так как в качестве формального параметра используется массив MAS[1..R] необходимо в разделе TYPE обозначить его тип.

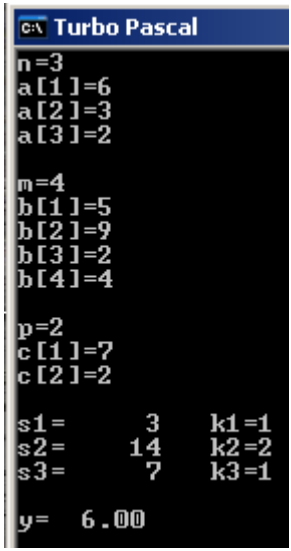
```
Program SK;
TYPE
Massiv = array[1..100] of integer;
Var
{Описание глобальных параметров}
A, B, C : massiv; I, N, M, P : integer;
```

```
S1, S2, S3,K1, K2, K3 : integer;
Y : real;
Procedure  SSS(var R : integer; var MAS : massiv; var SUM : inte-
ger; var K : integer);
Var
{Описание локальных переменных}
J : integer;
BEGIN
{тело процедуры}
SUM := 0; KOL := 0;
FOR J := 1 to R do
IF MAS[J] mod 2<>0
Then BEGIN
SUM := SUM + MAS[J];
KOL := KOL +1;
End;
End;  {конец процедуры}
BEGIN
{начало операторного блока основной программы}
Writeln(‘введите N’); readln(N);
FOR I:=1 to N do begin
Write(‘a[‘,I,’]=’); readln(A[I]);   end;
Writeln(‘введите M’);  readln(M);
FOR I := 1 to M do begin
Write(‘B[’,I,’]=’);  readln(B[I]);   end;
Writeln(‘введите P’);  readln(P);
FOR I := 1 to P do begin
Write (‘C[‘,I,’]=’);   readln(C[I]);   end;
{Первое обращение к процедуре}
SSS(N, A, S1, K1);writeln(‘S1=’,S1,’’:3,’K1=’,K1);
{Второе обращение к процедуре}
SSS(M, B, S2, K2); writeln(‘S2=’,S2,’’:3,’K2=’,K2);
{Третье обращение к процедуре}
SSS(P, C, S3, K3); writeln(‘S3=’,S3,’’:3,’K3=’,K3);
Y = (S1+S2+S3)/(K1+K2+K3);
Writeln(‘Y=’, Y:6:2);
{вывод массивов}
```

```
Writeln('массив A');
FOR I := 1 to N do write (A[I]:6);
Writeln('Массив B');
FOR I :=1 to M do write (B[I]:6);
Writeln('Массив C');
FOR I := 1 to P do write (C[I]:6);
Readln;
end.
```

Примечание: в данной программе показана организация одной процедуры. Для сокращения объма программы желательно ввод и вывод организовать в виде процедур.

Тестирование программы



Лабораторная работа №8

Разработка алгоритмов для работы с символьными данными (текстом)

Цель работы – овладение навыками алгоритмизации и программирования задач, обрабатывающих символьные данные; ввода и вывода символьных данных, их обработки; использования функций обработки символьных данных.

Краткая теория

При обработке текстовой информации (строк и символов) используются те же алгоритмы, что и при обработке числовых данных. Это - прежде всего базовые структуры (линейный, ветвление, циклы), а также их усложненные конструкции - вложенные развилки, вложенные циклы. Кроме того, необходимо знать типовые алгоритмы обработки одномерных и двумерных массивов. Всё это подробно рассмотрено и практически усвоено в предыдущих заданиях. Следует помнить, что при использовании в программах строк и символов необходимо в разделе описаний указывать соответствующий тип: **String** (строка) и **Char** (символ). Для обработки текстовых данных можно использовать четыре стандартных процедуры и столько же стандартных функций:

Delete (stroka,N,M) – процедура удаления из строки **M** символов, начиная с позиции **N;**

Insert (S1, stroka, N) – процедура вставки подстроки **S1** в строку, начиная с символа **N;**

Str (X,S) – процедура преобразования численного значения **X** в его строковое представление **S;**

Val (S,X) – процедура преобразования строкового выражения в его численное представление **X;**

Concat (S1,S2[,S3,…SN]) – функция конкатенации (соединения) последовательности строк;

Copy (stroka,N,M) – функция выделения из строки подстроки, начиная с символа **N** длиной **M** символов;

Length (stroka) – функция ,определяющая длину строки;

Pos (S1, stroka) – функция, определяющая позицию, начиная с которой подстрока **S1** входит в строку

Контроль по теме

1. Способы записи символьных и строковых констант.
2. Какие операции можно выполнять над символьными данными?
3. Способ описания символьных данных.
4. Назвать стандартные функции, используемые для обработки символьных данных.
5. Отличие строковых и символьных переменных.
6. Типизированные константы строкового типа

Задание к работе

1. Разработать программу обработки символьных данных в соответствии с вариантом, указанным в таблице 14
2. Проверить правильность выполнения программы с помощью тестового варианта.

Таблица 15

Варианты заданий

Вариант задания	Условие задачи
1	Проверить, имеется ли в заданном тексте баланс открывающих и закрывающих скобок
2	Для встречающихся в заданном тексте пар рядом расположенных символов указать, сколько раз встречается каждое из таких двухбуквенных сочетаний
3	Отредактировать предложение, удаляя из него лишние пробелы, оставляя только по одному пробелу между словами
4	В заданном предложении указать слово, в котором доля гласных (А, Е, И, О) максимальна

Окончание табл.15

Вариант задания	Условие задачи
5	Для каждого символа заданного текста указать, сколько раз он встречается в тексте. Сообщение об одном символе должно печататься не более одного раза
6	Для каждого слова заданного предложения указать долю согласных. Определить слово, в котором доля согласных максимальна
7	Найти самое длинное симметричное слово заданного предложения, например АККА
8	Отредактировать заданное предложение, заменяя многоточия точкой
9	В заданном предложении найти самое короткое и самое длинное слова
10	Из заданного текста предложения выбрать и напечатать только те символы, которые встречаются в нем только один раз (в том порядке, в котором они встречаются в тексте)
11	В заданном тексте заменить последовательность символов X (I) на A (I) и подсчитать число произведенных замен
12	В заданном тексте удалить символ «,» и подсчитать число удаленных символов
13	Из текста выбрать числа и записать в массив N. Количество чисел не более 10
14	Удалить из текста символы « O, A » и подсчитать длину сформированного текста
15	В тексте предложения заменить символ «.» на символы «,». Конечные символы удалить, не заменяя на запятые. Определить длину предложения. Если в тексте встречается несколько символов «.» подряд, то вместо них поставить одну запятую

Пример выполнения задания

Задание. Написать программу, которая удаляет начальные пробелы из введенной с клавиатуры строки.

Постановка задачи

St – строка текста, переменная строкового типа;

Разработка алгоритма

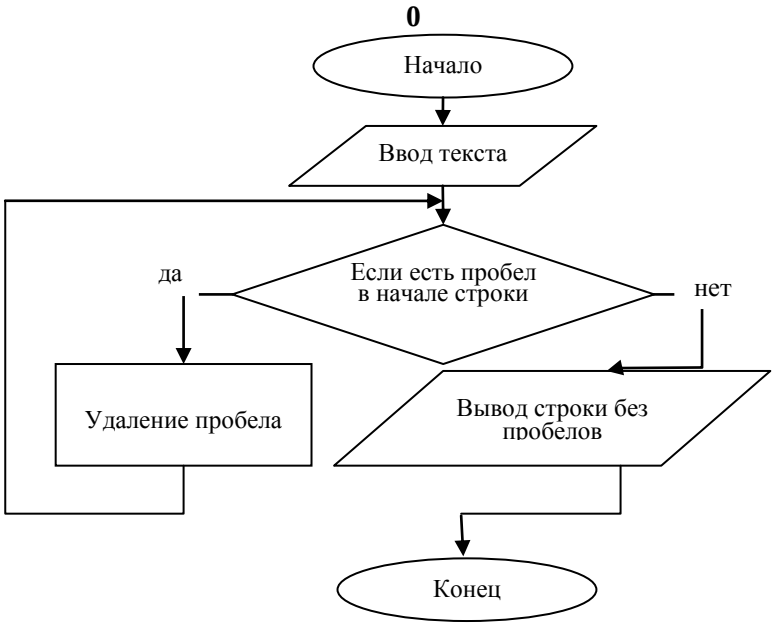


Рис. 21. Обработка текстовой информации

Составление программы

```
PROGRAM stroka;
USES CRT;
VAR  st:string[80];
BEGIN clrscr;
WRITELN ('введите строку');
READLN (st);
```

```
WHILE (pos(' ',st)=1) AND (length(st)>0) DO
delete (st,1,1);
WRITE ('строка без начальных пробелов:',st);
READLN;
END.
```

Тестирование программы

Список литературы

1. Алексеев В.Е., Валиулин А.С., Петрова Г.Б. Вычислительная техника и программирование: Практикум по программированию. – М.: Высшая школа, 1991. – 400с.
2. *Аляев Ю.А.,Гладкой В.П.* Практикум по алгоритмизации и программированию на языке Паскаль: Учебное пособие.-М.:Финансы и статистика, 2004. – 528с.
3. *Культин Н.Б.* TurboPascal в задачах и примерах.- СПб.: БХВ-Петербург, 2006.-256с.:ил.
4. *Павловская Т.А.*Паскаль. Программирование на языке высокого уровня: Учебник. - Питер, 2003. - 400с.
5. *Попов В.В* .Паскаль и Делфи: Учебное пособие. - Питер, 2005. – 576с.
6. *Фаронов В.В.* TurboPascal 7.0: Учебное пособие.-М.:«Нолидж», 1998.-616 с.

ОГЛАВЛЕНИЕ