

Направление: 022000.62 Экология и природопользование	Профиль:
Форма обучения: заочная - сокращенная	Курс: 1
Дисциплина: Основы программирования	Вид контроля: контрольная работа

Контрольная работа по основам программирования

ОБЩИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Многие задачи, предназначенные для решения на ЭВМ, предусматривают разработку алгоритма их реализации.

Алгоритм – это точное предписание, которое определяет процесс, ведущий от исходных данных к требуемому конечному результату.

Можно дать и другие определения алгоритма:

- 1) это описание последовательности действий для решения задачи или достижения поставленной цели;
- 2) это правила выполнения основных операций обработки данных;
- 3) это описание вычислений по математическим формулам.
- 4) это точная последовательность действий при решении какой-либо задачи, которая приводит к результату.

Алгоритмами, например, являются правила сложения, умножения, решения алгебраических уравнений, умножения матриц и т. п. Слово «алгоритм» происходит от *algoritmi*, являющегося латинской транслитерацией арабского имени хорезмийского математика IX века аль-Хорезми. Благодаря латинскому переводу трактата аль-Хорезми европейцы в XII веке познакомились с позиционной системой счисления, и в средневековой Европе алгоритмом стали называть десятичную позиционную систему счисления и правила счета в ней.

Применительно к ЭВМ алгоритм определяет вычислительный процесс, начинающийся с обработки некоторой совокупности возможных исходных данных и направленный на получение определенных этими исходными данными результатов. Термин «вычислительный процесс» распространяется и на обработку других видов информации, например, символьной, графической или звуковой.

Одним из свойств алгоритма является дискретность – возможность расчленения процесса вычислений, предписанных алгоритмом, на отдельные этапы, возможность выделения участков программы с определенной структурой. Можно выделить и наглядно представить графически три простейшие структуры: последовательность двух или более операций, выбор направления, повторение.

Любой вычислительный процесс может быть представлен как комбинация этих элементарных алгоритмических структур. Вычислительные процессы, выполняемые на ЭВМ по заданной программе, можно разделить на три основных вида: линейные, разветвляющиеся, циклические.

ЗАДАНИЕ 1. ЛИНЕЙНЫЕ АЛГОРИТМИЧЕСКИЕ ПРОЦЕССЫ

Линейным принято называть вычислительный процесс, в котором операции выполняются последовательно, в порядке их записи. Каждая операция является самостоятельной, независимой от каких-либо условий. На схеме блоки, отображающие эти операции, располагаются в линейной последовательности. Линейные вычислительные процессы имеют место, например, при вычислении арифметических выражений, когда имеются конкретные числовые данные и над ними выполняются соответствующие условию задачи действия.

На рис. 1 показан пример линейного алгоритма, определяющего процесс вычисления арифметического выражения $y = (b^2 - ac):(a + c)$ для различных значений переменных a , b и c .

СОДЕРЖАНИЕ ЗАДАНИЯ

1. Составить блок-схему алгоритма вычисления функции. Функцию предварительно разбить простейшие функции, вводя новые переменные. Один элемент блок-схемы не должен содержать более двух арифметических операций, например $k = x^2 + 4$. Результат представить в виде рисунка MS Word.

2. По блок-схеме п.1 составить код программы на языке VBA. Одному элементу блок-схемы соответствовать одна строчка программы. Данные должны вводиться с помощью оператора «Input()». Результат представить в виде текста MS Word.

3. Вычислить функцию для десяти произвольных значений аргумента. Результат представить в виде таблицы MS Word.

4. Построить график функции. Результат представить в виде вставленной в MS Word точечная сглаженной диаграммы MS Excel со следующими параметрами: название – над диаграммой, легенда – отсутствует, подписи осей – есть, сетка – с основными и промежуточными горизонтальными и вертикальными линиями, толщина линии функции – не менее 2 пт, фон – прозрачный.

ИСХОДНЫЕ ДАННЫЕ

Таблица 1 – Исходные данные к заданию 1 («№» соответствует номеру в списке группы. Номерам 31, 51, 71 и т.д. соответствует №1)

№	Функция	Данные	№	Функция	Данные
1	2	3	4	5	6
1		$a=2,22;$ $b=3;$ $c=-14,2$	11	<hr/> <hr/>	$a=2,22;$ $b=3;$ $c=-14,2$

2	—	$a=2,22;$ $b=3;$ $c=-14,2$	12	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$
3	_____	$a=2,2712;$ $b=33,18;$ $c=-14,2$	13	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$

Продолжение таблицы 1

1	2	3	4	5	6
4	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$	14	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$
5	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$	15	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$
6	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$	16	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$
7	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$	17	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$
8	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$	18	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$
9	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$	19	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$
10	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$	20	<u> </u> <u> </u>	$a=2,22;$ $b=3;$ $c=-14,2$

ЗАДАНИЕ 2. РАЗВЕТВЛЯЮЩИЕСЯ АЛГОРИТМИЧЕСКИЕ ПРОЦЕССЫ

Вычислительный процесс называется разветвляющимся, если для его реализации предусмотрено несколько направлений (ветвей). Каждое конкретное направление процесса обработки данных является отдельной ветвью вычислений. Ветвление в программе – это выбор одной из нескольких последовательностей команд при выполнении программы. Выбор направления зависит от заранее определенного признака, который может относиться к исходным данным, к промежуточным или конечным результатам. Признак характеризует свойство данных и имеет два или более значений.

Разветвляющийся процесс, включающий в себя две ветви, называется простым, более двух ветвей – сложным. Сложный разветвляющийся процесс можно представить с помощью комбинации простых разветвляющихся процессов. Направление ветвления выбирается логической проверкой, в результате которой возможны два ответа: «да» («+») – условие выполнено и «нет» («–») – условие не выполнено.

Следует иметь в виду, что, хотя на схеме алгоритма должны быть показаны все возможные направления вычислений в зависимости от выполнения определенного условия (или условий), при однократном прохождении программы процесс реализуется только по одной ветви, а остальные исключаются. Любая ветвь, по которой осуществляются вычисления, должна приводить к завершению вычислительного процесса.

На рис. 2 показан пример алгоритма с разветвлением для вычисления следующего выражения:

$$Y = \begin{cases} (a + b), & \text{если } X \leq 0 \\ \frac{c}{b}, & \text{если } X > 0. \end{cases}$$

В зависимости от истинности или ложности условия, указанного в символе решения, значение функции вычисляется по одному из двух альтернативных выражений.

На рис. 3 показан еще один пример алгоритма с разветвлением для вычисления значения ступенчатой функции:

$$z = \begin{cases} 0,5x^2 + 1, & \text{при } x \leq a \\ \cos x, & \text{при } a < x < b \\ \operatorname{tg}^2 x, & \text{при } x \geq b. \end{cases}$$

В алгоритме после ввода любых значений x , a и b и проверки условия в блоке 3 значение функции z может вычисляться по одной из трех альтернативных ветвей: в блоке 4, 5 или 6. При этом, так как количество альтернативных ветвей

больше двух, около соответствующего вывода блока 3 указывается условие перехода на соответствующую ветвь, а внутри самого блока указывается лишь имя переменной x .

Результаты вычисления значения функции по любой из альтернативных ветвей выводятся в блоке ввод-вывод 7.

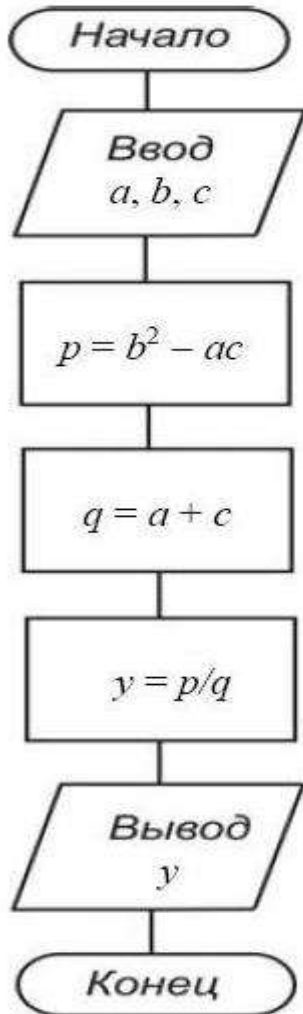


Рисунок 1 – Линейный алгоритм

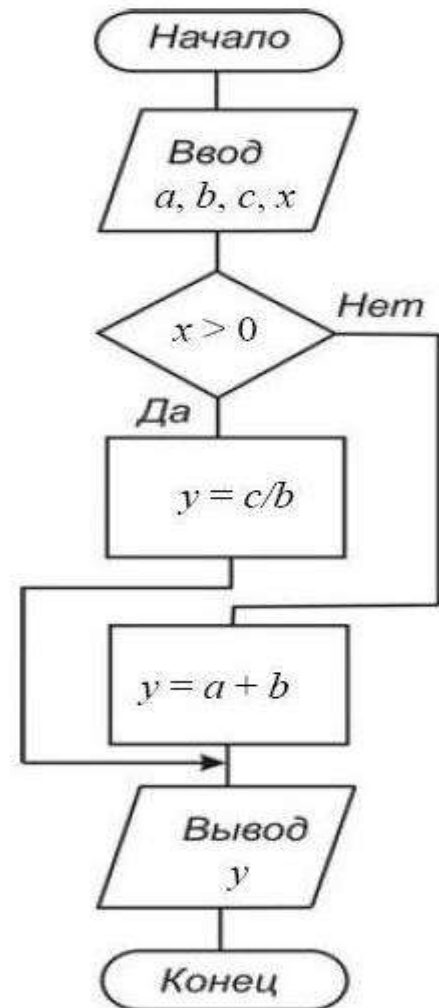


Рисунок 2 – Разветвляющийся алгоритм

СОДЕРЖАНИЕ ЗАДАНИЯ

1. Составить блок-схему алгоритма вычисления максимума и минимума функции, по правилам п.1 задания 1. Сравниваемые величины предварительно вычисляются. Результат представить в виде рисунка MS Word.
2. По блок-схеме п.1 составить код программы на языке VBA. Результат представить в виде текста MS Word.
3. Вычислить функцию для заданных значений аргумента. Результат представить в виде таблицы MS Word.

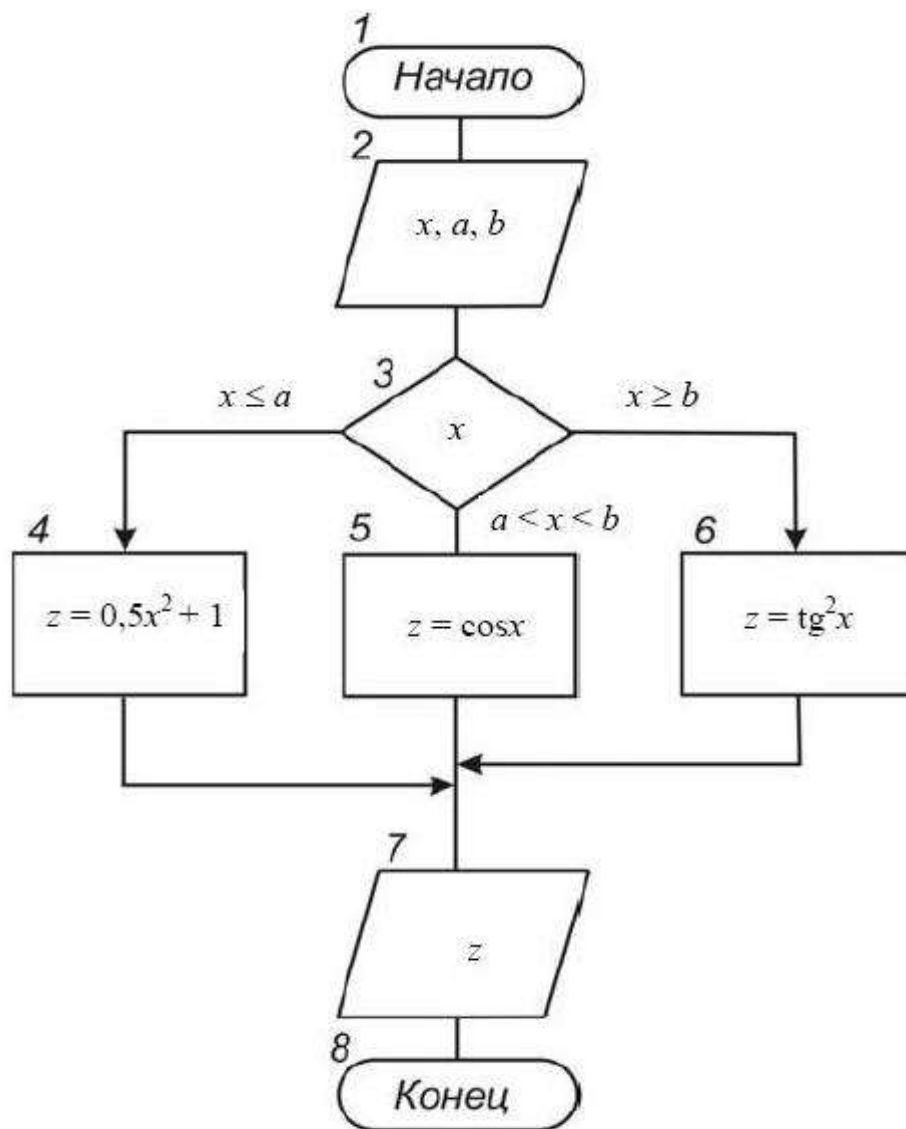





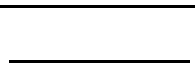








Рисунок 3 – Разветвляющийся алгоритм ступенчатой функции

ИСХОДНЫЕ ДАННЫЕ

Таблица 2 – Исходные данные к заданию 2 («№» соответствует номеру в списке группы)

№	Функция	Составляющие	Аргументы
1	2	3	4
1		- ;	$x=1,24;$ $y=2,35$
2	_____		$x=-8,2;$ $y=2,25$

Продолжение таблицы 2

1	2	3	4
3			$x=3,2;$ $y=1,32$
4			$x=1,25;$ $y=2,25$
5			$x=2,32;$ $y=-3,43$
6			$x=-2,32;$ $y=3,3$
7			$x=-4,28$
8			
9			$x=1,09$
10			$x=1,78$
11			$x=4,0$
12			x - любое
13			$x=-2,2$
14			$x=-3,8$
15			$x=4,35$
16			$x=1,24;$ $y=2,35$
17			$x=8,2;$ $y=2,25$
18			$x=3,2;$ $y=1,32$
19			$x=-1,25;$ $y=2,25$
20			$x=2,32;$ $y=3,43$

ЗАДАНИЕ 3. ЦИКЛИЧЕСКИЕ АЛГОРИТМИЧЕСКИЕ ПРОЦЕССЫ

Циклическими называются алгоритмы, содержащие циклы. Цикл – это многократно повторяемый участок алгоритма.

В организации цикла можно выделить следующие этапы (рис. 4):

- подготовка (инициализация) цикла (И);
- выполнение вычислений цикла (тело цикла) (Т);
- модификация параметров (М);
- проверка условия окончания цикла (У);
- продолжение (П).

На этапе инициализации присваиваются начальные значения переменной цикла X , тем самым обеспечивается выполнение тела цикла на первом цикле.

Тело цикла в общем случае может содержать любое количество операций (в частности, может вообще отсутствовать).

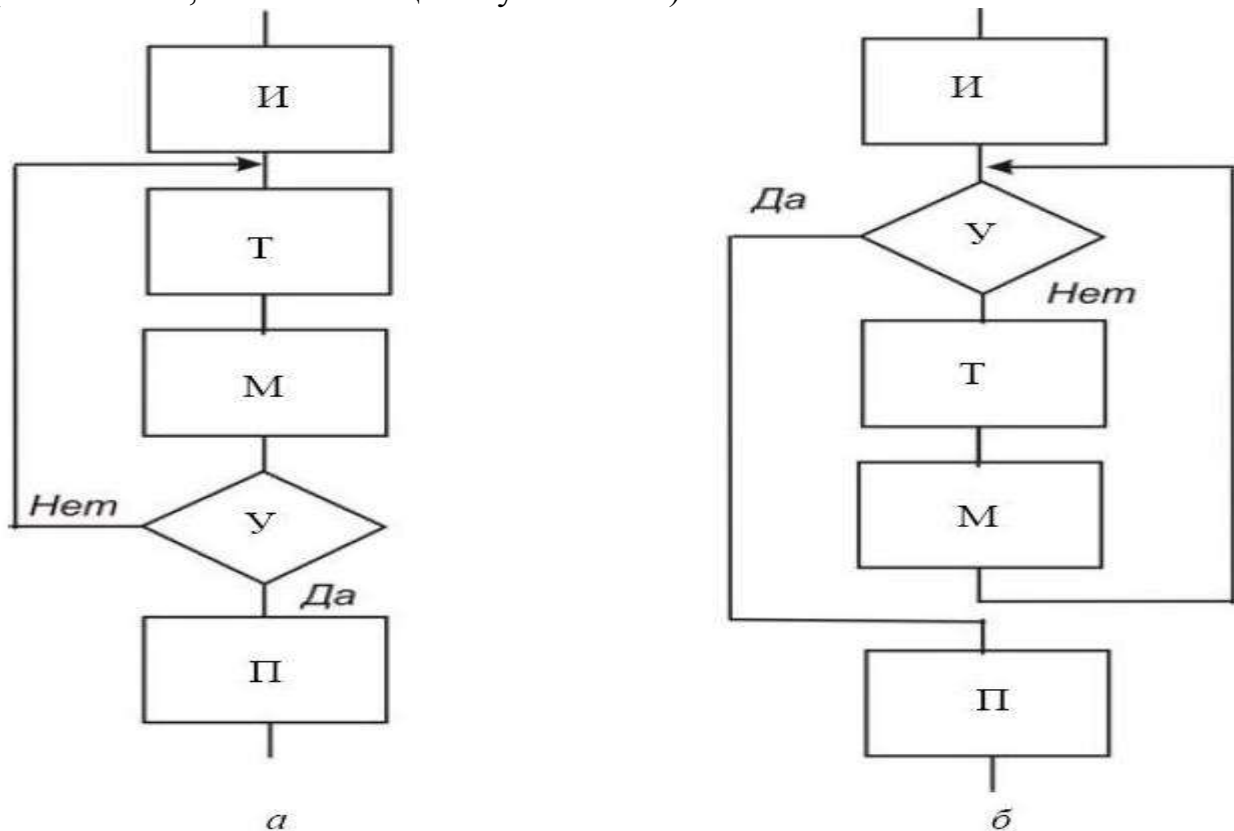


Рисунок 4 – Последовательность этапов цикла: *а* – с нижним окончанием, *б* – с верхним окончанием.

На этапе модификации параметров в случае, если существует закономерность изменения значений переменной цикла, а также если эти значения используются строго последовательно, то к текущему значению переменной цикла прибавляется шаг ΔX с учетом знака, т.е. $X = X + \Delta X$.

Если не существует никакой закономерности изменения значений переменной цикла, то эти значения могут храниться в одномерном массиве в виде $X(i)$. В этом случае в качестве переменной цикла будет выступать i ,

модификация которой будет также производиться путем изменения ее значения на единицу, т.е. $i = i + 1$.

Проверка условия окончания цикла чаще всего предполагает сравнение текущего значения переменной цикла с ее конечным значением. В общем случае условие окончания цикла может быть явно не связано ни со значением переменной цикла, ни с ее конечным значением.

Порядок выполнения этапов, например, T и M , может изменяться. В зависимости от расположения проверки условия окончания цикла различают циклы с нижним и верхним окончаниями. Для цикла с нижним окончанием (рис. 4,а) тело цикла выполняется как минимум один раз, так как сначала производятся вычисления, а затем проверяется условие выхода из цикла. В случае цикла с верхним окончанием (рис. 4,б) тело цикла может не выполниться ни разу в случае, если сразу соблюдается условие выхода.

После выхода из цикла алгоритм переходит на этап продолжения программы (этап P на рис. 4).

Особенности алгоритмов и программ с накоплением

Вычисление суммы (рис. 5,а) и произведения нескольких чисел, а также подсчет количества (рис. 5,б) в большинстве языков программирования выполняется постепенно, путем накопления.

1. Сумма вычисляется по формуле:

$$\text{сумма} = \text{сумма} + \text{слагаемое}.$$

Перед вычислениями начальному значению суммы нужно присвоить значение «ноль».

2. Произведение вычисляется по формуле

$$\text{произведение} = \text{произведение} \times \text{сомножитель}.$$

Перед вычислениями начальному значению произведения нужно присвоить значение единицы.

3. Подсчет количества выполняется по формуле, подобной формуле для вычисления суммы:

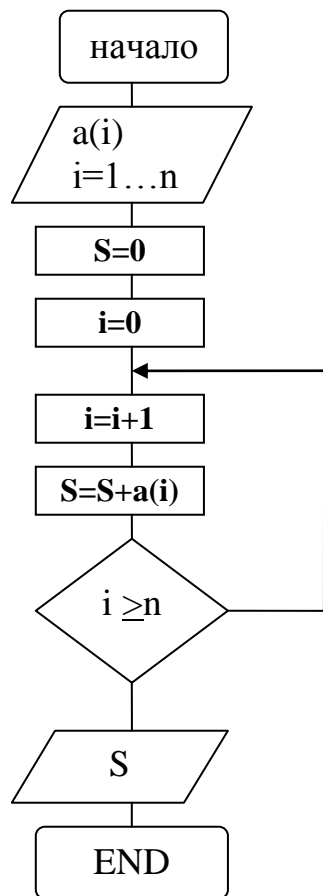
$$\text{количество} = \text{количество} + 1.$$

Начальному значению количества присваивается значение «ноль».

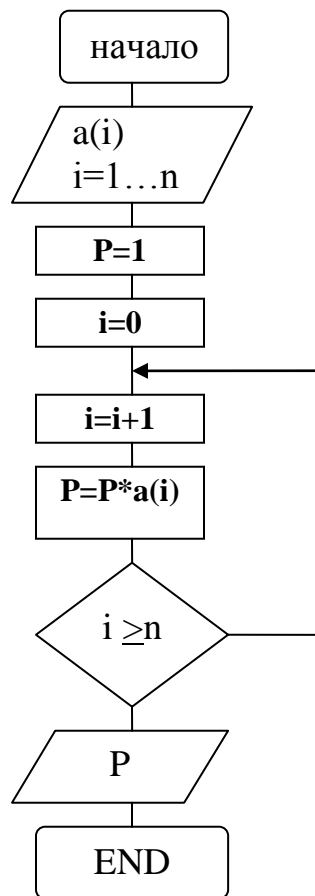
СОДЕРЖАНИЕ ЗАДАНИЯ

1. Составить блок-схему алгоритма вычисления суммы и произведения полученных значений функции. Результат – рисунок MS Word.
2. По блок-схеме составить код программы на языке VBA. Результат – текст MS Word.
3. Вычислить сумму и произведение функции для заданных значений аргумента.

Исходные данные в таблице 3.



а)



б)

Рисунок 5 – Циклический алгоритм: а – для подсчета суммы, б – для подсчета конечного значения

Аналогично п.1-3, вычислить сумму членов ряда с точностью $E=10^{-2}$ (ряд продолжается, пока выполняется условие $x > E$), при заданном $x = 0,75$. Исходные данные в таблице 4.

ИСХОДНЫЕ ДАННЫЕ

Таблица 3 – Исходные данные к п. 1-3 задания 3 («№» – номер в списке группы)

№	Выражение	Данные	№	Выражение	Данные
1	2	3	4	5	6
1			11	_____	
2			12		

Продолжение таблицы 3

1	2	3	4	5	6
3	<hr/> -		13		
4			14	<hr/> <hr/>	
5	<hr/> <hr/>		15	<hr/> <hr/>	
6			16		
7			17		
8	<hr/> <hr/>		18	<hr/> <hr/>	
9			19		
10			20	<hr/> <hr/>	

Таблица 4 – Исходные данные к п. 4 задания 3 («№» – номер в списке группы)

№	Выражение	№	Выражение
1	2	3	4
1	— —	11	—, E=10 ³
2	— —	12	— —, E=10 ²
3	— —	13	— —
4	— —	14	— —
5	— —	15	— —
6	— —	16	— —
7	— = — =, E=10 ²	17	— —
8	— —	18	— —
9	— —	19	— = — =
10	— —	20	— — = — =

ЗАДАНИЕ 4. ЗАДАЧИ ПО ОБРАБОТКЕ ОДНОМЕРНЫХ МАССИВОВ

Массив – это совокупность элементов, имеющих одинаковое имя и отличающихся номером (индексом) их расположения в последовательности.

Одномерный массив – последовательность чисел $A(1), A(2), A(3), \dots, A(n)$. Размерность массива равна количеству элементов в нем.

Пример 1: Дан одномерный массив $A(i)$ длиной 20, упорядоченный по убыванию. Переупорядочить массив $A(i)$ по возрастанию его элементов,

используя первоначальную упорядоченность массива, и сохранить его значения в массиве $B(i)$. Вывести полученный массив $B(i)$ на дисплей.

Словесный алгоритм решения задачи:

1. В цикле последовательно перебрать все элементы массива $B(i)$ (начиная с первого по двадцатый).
2. Присвоить им значения соответствующих элементов массива $A(21-i)$ (начиная с конца).
3. Вывести полученное значение на дисплей
4. Повторять тело цикла 20 раз.

Блок-схема алгоритма приведена на рис. 6.

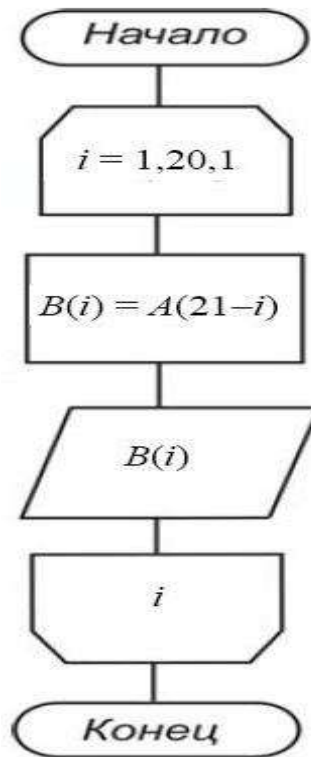


Рисунок 6 – Блок-схема примера 1

Пример 2: Дан одномерный массив. Сформировать массив по убыванию. Попарное сравнение чисел в массиве и постановка их по убыванию.

Схема алгоритма и код программы для VBA приведены на рис. 7 и 9.

Пример 3: Сформировать одномерный массив по убыванию методом пузырька.

Блок-схема алгоритма и код программы для VBA приведены на рис. 8 и 10.

Пример 4: Найти максимальный элемент одномерного массива.

Описание алгоритма:

1. Задается число элементов массива n .
2. Начальному значению переменной i присваивается 0.
3. Дается приращение 1 значению переменной i .
4. Первому $a(i)$ числу присваивается максимум.

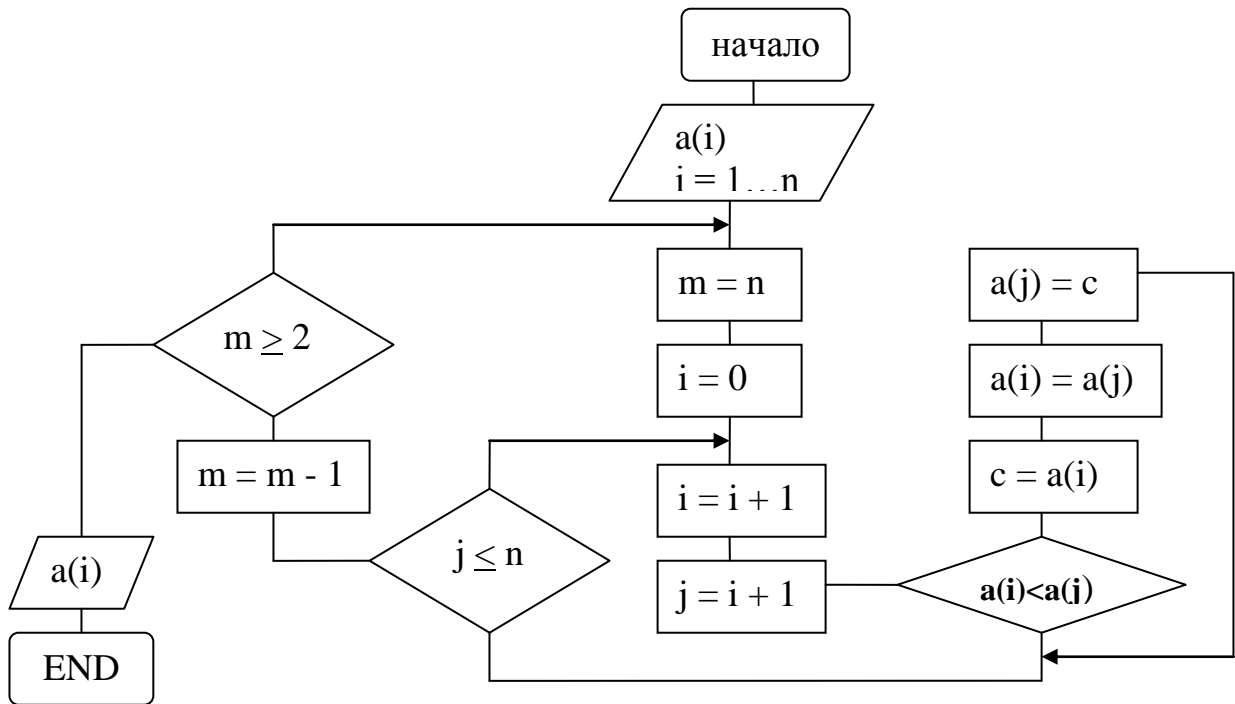


Рисунок 7 – Блок-схема примера 2

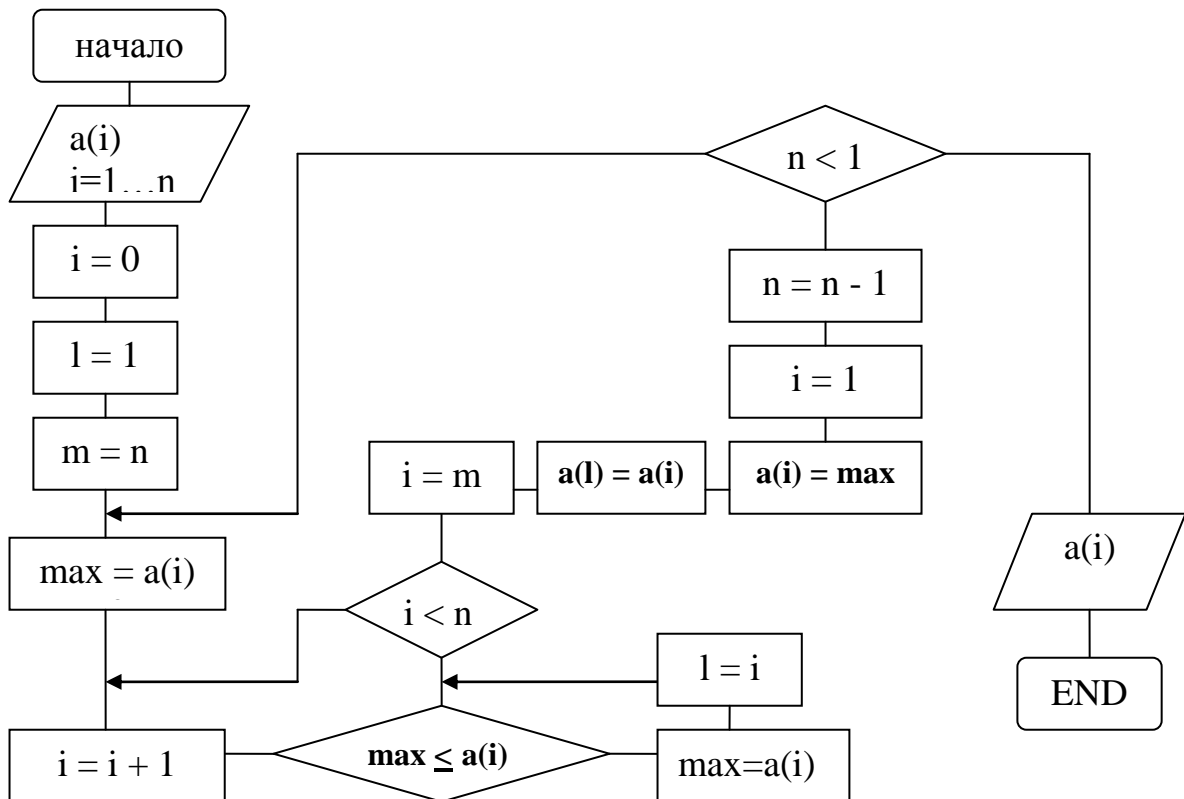


Рисунок 8 – Блок-схема примера 3

```

Sub poparnoe_sravnenie()
Dim i, j, m as Integer
For m = 5 to 2 Step -1
For i = 1 to (m - 1)
j = i + 1
a(i) = cells(1,i)
a(j) = cells(1,j)
If a(i) < a(j) then
Cells(1,j) = a(i)
Cells(1,i) = a(j)
Next i
Next m
End Sub

```

Рисунок 9 –
Код программы примера 2

```

Sub puzir()
Dim i, j as Integer
Dim k, max as Variant
For j = 5 to 1 Step -1
i = 1
l = 1
k = Cells(1, j)
max = Cells(1, i)
For i = 1 to j
If Cells(1, i) < max then
max = Cells(1, i)
l = i
End If
Next i
Cells(1, j) = max
Cells(1, l) = k
Next k
End Sub

```

Рисунок 10 –
Код программы примера 3

5. Делается сравнение, переменной $a(i)$ с максимумом. Если условие выполняется, переменной $a(i)$ присваивается максимум, если нет— делается еще одна проверка в п.6

6. Проверяется, не закончились ли элементы, которые нужно перебирать (i). Если значение переменной i не удовлетворяет условию, происходит приращение i и ввод новой переменной $a(i)$ – переход к п.3; иначе – завершается выполнение программы.

Блок-схема алгоритма и код программы приведены на рис. 11 и 12.

Пример 5: Найти число положительных, отрицательных чисел и нулей в одномерном массиве

Описание алгоритма:

1. Задаются начальные значения переменных i, j, g, m .
2. Организуется цикл пока значение i больше или равно n . (n – последнее число в массиве).
3. В теле цикла делаются 3 проверки чисел массива на знак числа или равенство нулю. Если значение переменной $a(i)$ удовлетворяет условию, то счетчик увеличивает на единицу соответственно j, g или m (каждый счетчик соответствует своему условию: j – для $a(i) > 0$; g – для $a(i) < 0$; m – для $a(i) = 0$).

Блок-схема алгоритма и код программы приведены на рис. 13 и 14.

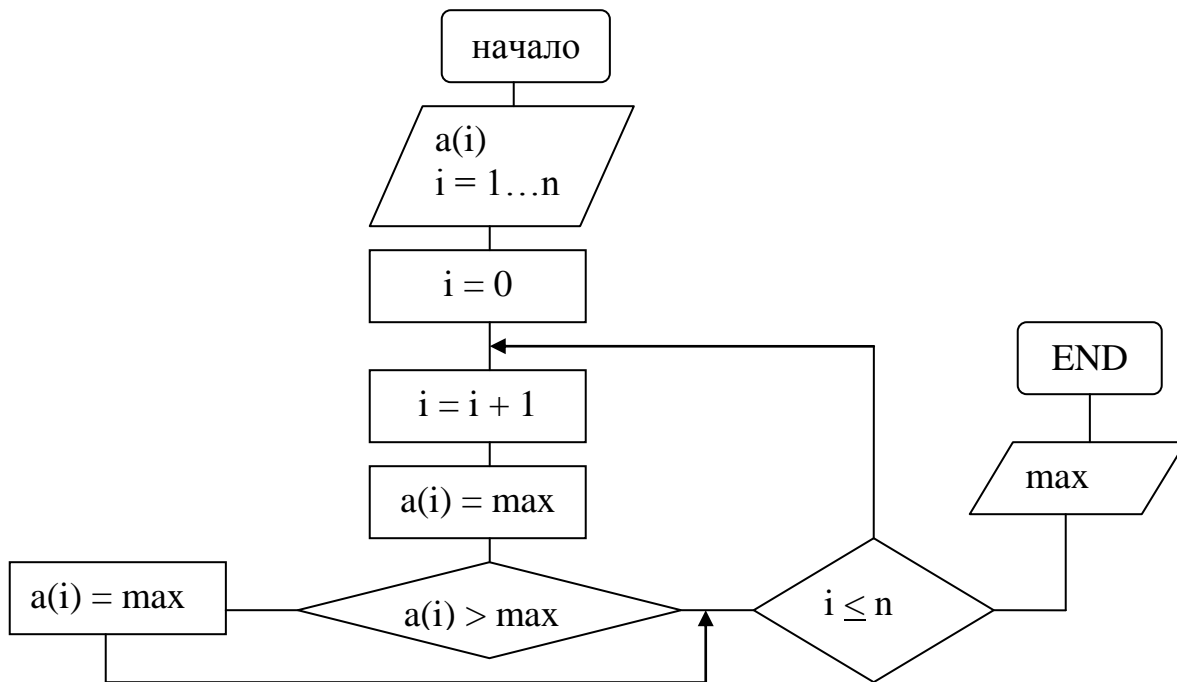


Рисунок 11 – Блок-схема примера 4

```

Function maximum(a)
Dim i, n as Integer
n = a.Rows.Count * a.Columns.Count
m = a(1)
For i = 2 to n
If a(i) > m Then
m = a(i)
End If
Next i
maximum = m
End Function
  
```

Рисунок 12 – Код программы примера 4

Пример 6: Найти сумму диагональных элементов 3-х мерного массива. Блок-схема алгоритма приведена на рис. 15.

Пример 7 (Задача с использованием алгоритма интерполирования): Имеются данные измерения температуры воздуха в продолжение дня: 8 часов – 7°C; 14 часов – 15°C; 10 часов – 10°C; 17 часов – 9°C и т.д.

Необходимо определить, какой была температура в 9 часов, в 12 часов 30 минут, в 15 часов 15 минут.

Описание алгоритма:

1. Задается начальное значение переменных i , T , t .
2. Организуется цикл пока значение i больше или равно n . (n – последнее число в массиве).

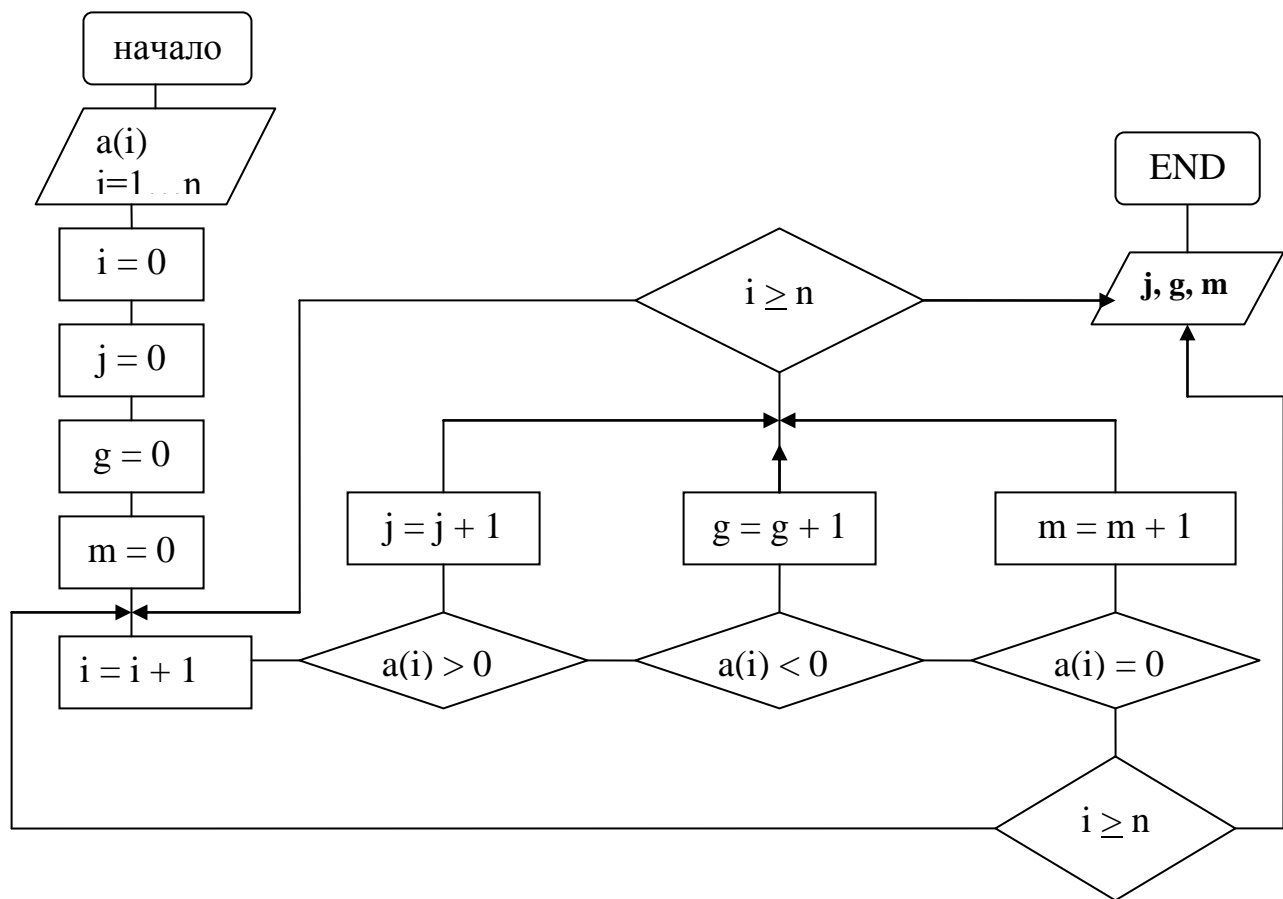


Рисунок 13 – Блок-схема примера 5

```

Function zxc(a)
Dim i, j, m, n, g As Integer
n = a.Rows.Count * a.Columns.Count
j = 0
g = 0
m = 0
For i = 1 to n
If a(i) > 0 Then
j = j + 1
End If
If a(i) < 0 Then
g = g + 1
End If
If a(i) = 0 Then
m = m + 1
End If
Next i
MsgBox("положительных" + Str(j) + "_" + "отрицательных" + Str(g) + "_" + "нули" + Str(m))
End Function

```

Рисунок 14 – Код программы примера 5

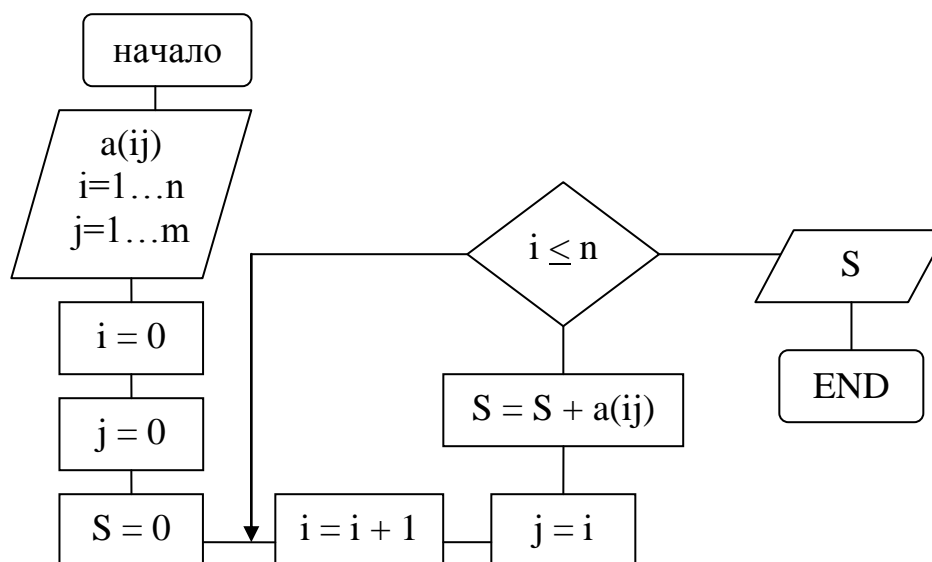


Рисунок 15 – Блок-схема примера 5

3. Проверка условия, входит ли переменная в уменьшенный промежуток, ограниченный известными значениями. Если значение переменной T не удовлетворяет условию, то проверяется следующий промежуток; иначе – задача решается по формуле $t = \frac{T - T_i}{T_{i+1} - T_i} (t_{i+1} - t_i) + t_i$.

Блок-схема алгоритма приведена на рис. 16.

Пример 8 (Решение квадратного уравнения):

Описание алгоритма:

1. Задаются начальные значения переменных a, b, c .
2. Вычисляется дискриминант.
3. Выполняется проверка, дискриминанта
4. Если дискриминант меньше нуля – появляется сообщение, что решений нет; иначе — находятся корни уравнения.

Блок-схема алгоритма приведена на рис. 17.

СОДЕРЖАНИЕ ЗАДАНИЯ

1. Составить блок-схему алгоритма операций с массивом (рисунок MS Word).
 2. По блок-схеме составить код программы (текст MS Word).
 3. Вычислить значение функции.
- Исходные данные в таблице 5.

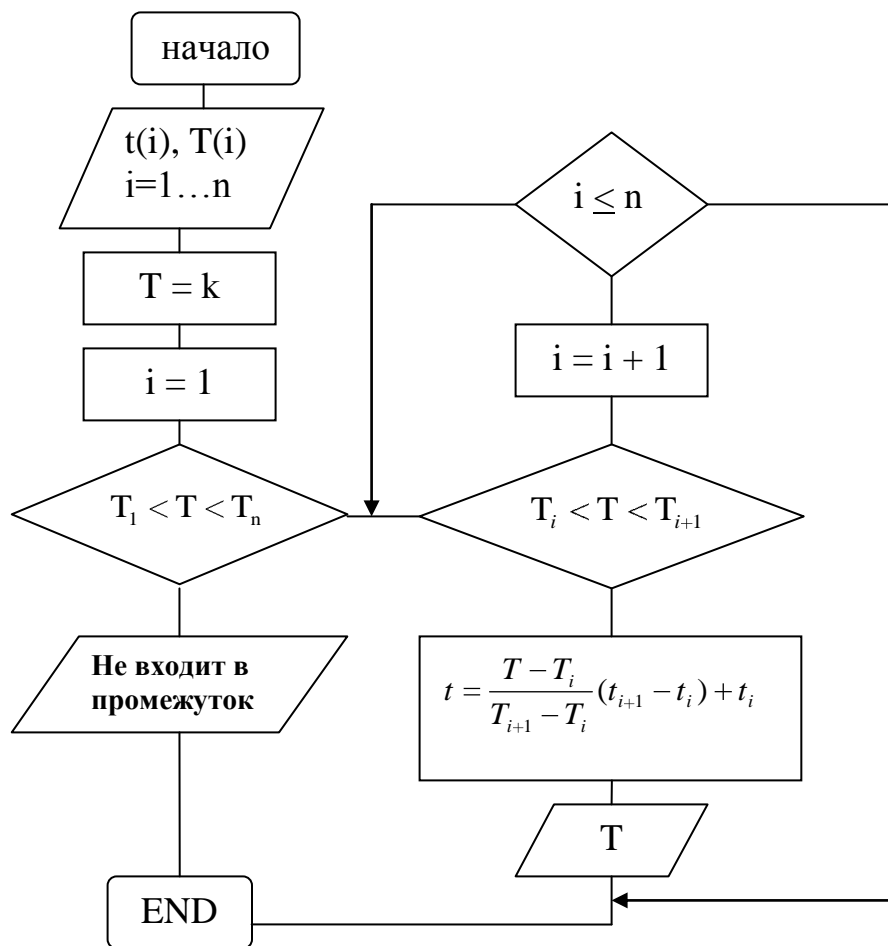


Рисунок 16 – Блок-схема интерполирования

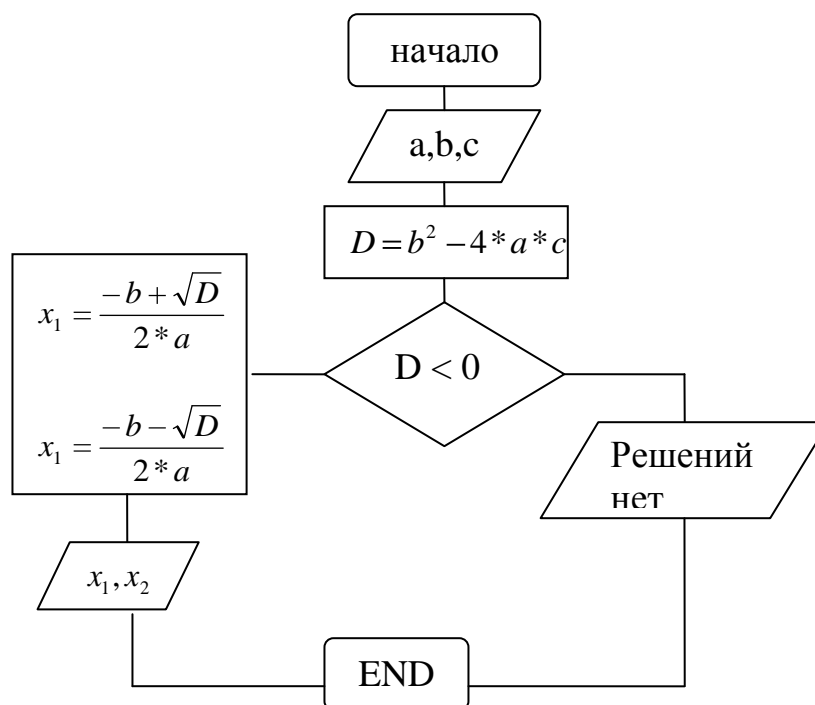


Рисунок 17 – Блок-схема решения квадратного уравнения

ИСХОДНЫЕ ДАННЫЕ

Таблица 5 – Исходные данные к заданию 4

№	Задание	Функция, массив	№	Задание	Функция, массив
1	2	3	4	5	6
1	Найти максимум и сумму положительных элементов массива <i>u</i>		11	Найти количество и произведение четных элементов массива <i>d</i>	
2	Найти минимум и сумму отрицательных элементов массива <i>A</i>		12	Найти сумму элементов над главной диагональю и максимальный из них	
3	Найти количество и сумму элементов массива <i>S</i> , меньших 2		13	Найти номер первого положительного элемента массива <i>t</i>	

Продолжение таблицы 5

1	2	3	4	5	6
4	Найти наибольшее и наименьшее значения функции		14	Найти произведение отрицательных элементов над главной диагональю	
5	Найти количество и сумму положительных элементов массива A		15	Найти номер первого отрицательного элемента массива G	
6	Найти произведение нечетных элементов массива t		16	Найти сумму положительных элементов над главной диагональю и максимальный из них	
7	Найти номер первого отрицательного элемента массива q		17	Найти количество и произведение нечетных элементов массива j	
8	Найти минимум и сумму положительных элементов массива A		18	Найти количество и сумму четных элементов массива a	
9	Найти количество и сумму положительных элементов массива Q , больших 2		19	Найти количество и произведение нечетных элементов массива k , больших 2	

Продолжение таблицы 5

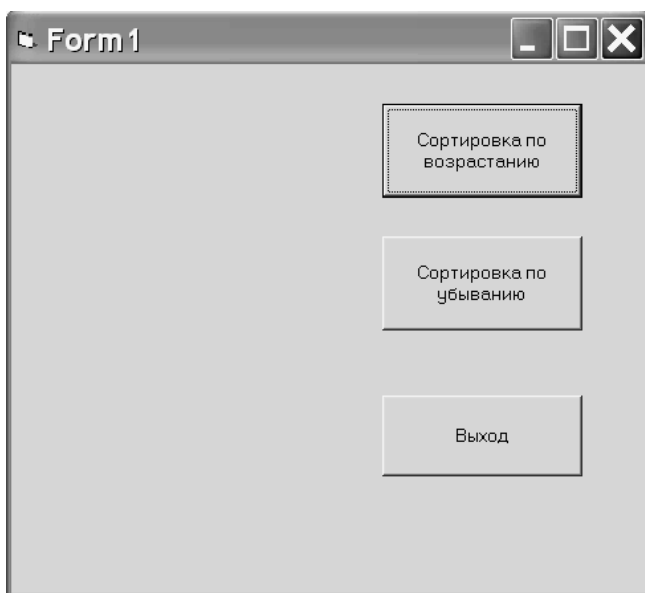
1	2	3	4	5	6
10	Найти количество и произведение элементов массива d больших 2		20	Найти произведение четных элементов массива C	

ЗАДАНИЕ 5. СОЗДАНИЕ ФОРМЫ

СОДЕРЖАНИЕ ЗАДАНИЯ

Создать форму для вычисления функции задания 4.

Пример создания формы для вычисления функции по сортировке массива



-создаёт массив случайных чисел и сортирует их по возрастанию

-создаёт массив случайных чисел и сортирует их по убыванию

-ВЫХОД

Рисунок 18 – Внешний вид формы

Ход работы:

1. Создаём проект, перетаскиваем на него 3 кнопки и надпись

2. Меняем следующие свойства:

Кнопка 1	Caption	Сортировка по возрастанию
Кнопка2	Caption	Сортировка по убыванию
Кнопка3	Caption	Выход
Метка1	Caption	

3. Вводим код в окно кода:

Option Explicit

Option Base 1

Private Sub Command1_Click()

```

Dim i As Integer, b(10) As Single
Print
Print "Исходный массив"
For i = 1 To 10
    b(i) = Int(10 * Rnd + 10)
    Print b(i);
Next i
Print
Print "Отсортированный массив"
Call SortMas(b(), 10, 2)
Label1.Caption = "Сортировка по возрастанию"
For i = 1 To 10
    Print b(i);
Next i
End Sub

```

```

Private Sub Command2_Click()
    Dim i As Integer, b(10) As Single
    Print
    Print "Исходный массив"
    For i = 1 To 10
        b(i) = Int(10 * Rnd + 10)
        Print b(i);
    Next i
    Print
    Print "Отсортированный массив"
    Call SortMas(b(), 10, 1)
    Label1.Caption = "Сортировка по убыванию"
    For i = 1 To 10
        Print b(i);
    Next i
End Sub

```

```

Private Sub Command3_Click()
End
End Sub

```

4.Добавляем модуль и вводим код:

```

Sub SortMas(a() As Single, n As Integer, f As Byte)
    Dim i As Integer, j As Integer
    If f = 1 Then
        For i = 1 To n - 1
            For j = i + 1 To n
                If a(i) < a(j) Then

```

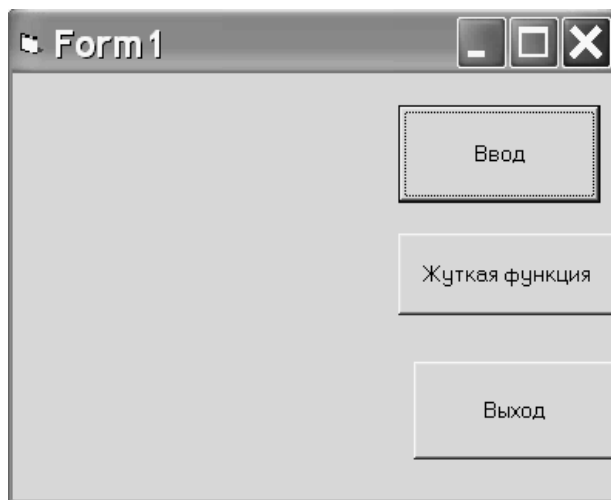
```

        a(i) = a(i) + a(j)
        a(j) = a(i) - a(j)
        a(i) = a(i) - a(j)
    End If
Next j
Next i
Else
    For i = 1 To n - 1
        For j = i + 1 To n
            If a(i) > a(j) Then
                a(i) = a(i) + a(j)
                a(j) = a(i) - a(j)
                a(i) = a(i) - a(j)
            End If
        Next j
    Next i
End If
End Sub

```

5. Запускаем приложение

Пример создания формы для вычисления двух функций



- считает функцию $y = 1 + X/1! + X^2/2! + X^3/3! + \dots$

-считает $y = -1 - X^2/3 - X^4/5 + X^6/6 + X^8/8 + X^{10}/10 - X^{12}/11 - \dots$

-Выход из программы

Рисунок 19 – Внешний вид формы

Ход работы:

1. Создаём проект, перетаскиваем на форму 3 кнопки

2. Изменяем следующие свойства:

Кнопка1	Caption	Ввод
Кнопка2	Caption	Жуткая функция
Кнопка3	Caption	Выход

3, Вводим код в окно кода:

```
Option Explicit
```

```
Dim eps As Double, Xn As Double, Xk As Double, Hy As Double
```

```
Function FN(x As Double, eps As Double) As Double
```

```
Dim U As Double
```

```
Dim N As Integer
```

```
Dim Y As Double
```

```
U = 1: Y = 1
```

```
N = 1
```

```
Do While Abs(U) > eps
```

```
N = N + 1
```

```
U = U * x / (N - 1)
```

```
Y = Y + U
```

```
Loop
```

```
FN = Y
```

```
End Function
```

```
Private Sub cmdVvod_click()
```

```
Dim x As Double, Y As Double
```

```
eps = Val(InputBox("Введите точность"))
```

```
Xn = Val(InputBox("Введите начальное значение X"))
```

```
Xk = Val(InputBox("Введите конечное значение X"))
```

```
Hy = Val(InputBox("Введите шаг"))
```

```
Print "Значение X", "Значение Y"
```

```
x = Xn
```

```
Do
```

```
Y = FN(x, eps)
```

```
Print x, Y
```

```
x = x + Hy
```

```
Loop Until x > Xk
```

```
End Sub
```

```
Function FN2(x As Double, eps As Double) As Double
```

```
Dim U As Double, Y As Double, N As Integer, K As Integer
```

```
U = -1: Y = -1: N = 0: K = 1
```

```
Do While Abs(U) > eps
```

```
N = N + 1
```

```
U = U * x * x * K
```

```
K = K + 2
```

```
If (N / 3 + 1) Mod 2 = 0 Then
```

```
    K = K - 1: U = -U
```

```
End If
```

```

U = U / K
Y = Y + U
Loop
FN2 = Y
End Function

```

```

Private Sub Command2_Click()
Dim x As Double, Y As Double
eps = Val(InputBox("Введите точность"))
10 Xn = Val(InputBox("Введите начальное значение X(от -1 до 1)"))
If Xn <=-1 Or Xn >= 1 Then GoTo 10
20 Xk = Val(InputBox("Введите конечное значение X(от -1 до 1)"))
If Xk >= 1 Or Xk <= Xn Then GoTo 20
Hy = Val(InputBox("Введите шаг"))
Print "Значение X", "Значение Y"
x = Xn
Do
Y = FN2(x, eps)
Print x, Y
x = x + Hy
Loop Until x > Xk
End Sub

```

```

Private Sub Command3_Click()
End
End Sub

```

5.Запускаем программу.

Пример создания формы для вычисления функции определения является ли вводимое выражение символом, чётным или нечётным числом, большой или маленькой латинской буквой



Рисунок 20 – Внешний вид формы

Кнопка «Запуск» запускает функцию на выполнение

Кнопка «Выход» служит для выхода из программы.

Ход работы:

1.Создаём стандартный exe

2.Переносим на форму 2 кнопки

3.Устанавливаем значение Caption первой кнопки кА Запуск, второй – Выход

4.Вводим код в окно кода:

```
Private Sub Command1_Click()  
    Dim msg, w  
    msg = "Введите символ"  
    w = InputBox(msg)  
    If Not (IsNumeric(w)) Then  
        If Len(w) = 1 Then  
            Select Case Asc(w)  
                Case 65 To 90  
                    msg = "Введена большая лат.буква"  
                Case 97 To 122  
                    msg = "Введена малая лат.буква"  
                Case Else  
                    msg = "Это не латинская буква и не число"  
            End Select  
        Else  
            msg = "Это не символ"  
        End If  
    Else  
        Select Case Val(w)  
            Case 1, 3, 5, 7, 9  
                msg = "Это нечётная цифра"  
            Case 0, 2, 4, 6, 8  
                msg = "Это чётная цифра"  
            Case Else  
                msg = "Это не цифра и не буква"  
            End Select  
        End If  
        Print msg  
    End Sub
```

```
Private Sub Command2_Click()
```

```
End
```

```
End Sub
```

Запускаем приложение.

ЛИТЕРАТУРА

1. Петрова Т.Е., Линденбаум М.Д. Вычислительная техника и программирование: Методические указания и задания к контрольной работе.; Рост.гос. ун-т путей сообщения. Ростов н/Д, 1997. 36 с.
2. Гарнаев А.Ю. Использование MS Excel и VBA в экономике и финансах. - СПб.:БХВ – Санкт-Петербург, 1999.- 336 с.,ил.