Уральский федеральный университет

ИННОВАЦИОННАЯ ОБРАЗОВАТЕЛЬНАЯ ПРОГРАММА

Уральский федеральный университет

Составные типы в языке С

Массивы

Лекция 9

Иллюстративный материал к лекциям по Информатике

Автор Саблина Н.Г. 2011 г.

Уральский федеральный университет

Содержание

Составные типы в языке С

По способу организации и типу компонентов в составных типах выделяют:

- регулярные типы (массивы);
- комбинированные типы (структуры);
- файловый тип (файлы);
- объектные типы (классы).

Массивы

Массив – это совокупность данных одного типа, расположенных в памяти ЭВМ последовательно, непосредственно одно за другим.

Массивы используются для представления

- векторов,
- матриц,
- символьных строк,
- образов экрана ПЭВМ и другой однородной информации.

Особенности массива

- Все элементы массива в целом обозначаются общим групповым именем (имя массива).
- Доступ к отдельным элементам массивов организуется посредством указания имени массива и порядкового номера (индекса) элемента.
- Индекс определяет положение элемента относительно начала массива.

Описание массива

- При описании массива необходимо указать:
  - тип элементов;
  - имя массива;
  - размерность массива.
- Общая форма описания массива:

тип имя_массива [размер1][размер2]…;

Одномерные массивы

При описании одномерного массива в скобках указывается только один индекс, определяющий количество элементов в массиве.

Например:

- **int vect[10], S1[50];**
- **float A[5], B[25];**

Описаны два целочисленных массива:

- vect, содержащий 10 элементов,

- S1, содержащий 50 элементов.

Два массива действительных чисел A и B, содержащие 5 и 25 элементов соответственно.

Описание массива с инициализацией

При описании можно инициализировать элементы массива заданными значениями.

Например:

```
int   D[5]={23, 45, 32, 12, 88};
float Z[4]={0.25,   67.89,    1.1,   -34.5};
char C[3]={‘М’, ’И’, ’Р’};
```

Псевдодинамическое описание массива

Для записи количества элементов в массиве удобно использовать именованные константы.

Например:

```
const N=10, M=5;
int vest  [N];
float mas [M];
```

Индексы элементов массива

- Для обращения к отдельному элементу массива указывают имя массива и в квадратных скобках индекс (порядковый номер) этого элемента в массиве.
- Элементы в массиве нумеруются, начиная с нуля, т.е.
  - Индекс первого элемента равен 0,
  - индекс последнего элемента – на единицу меньше размера массива.

Индексы элементов массива

- В качестве индексов могут выступать
  - Числовые константы
  - переменные
  - произвольные выражения целого типа

```
int vect[20];
int i=2;
vect [5]=45;
vect [i]=45;
vect [(i+1)*2]=5-i+1;
```

Размещение массивов в памяти

- Под массив выделяется непрерывное место в оперативной памяти.
- Это позволяет рассматривать массив как структуру произвольного (прямого) доступа, т.е. можно обращаться к любому элементу массива по его индексу i, не просматривая при этом предыдущие i-1 элемент.

Размещение массивов в памяти

int A[10];

A

45	88	3	9	34	12	7	99	32	67
0	1	2	3	4	5	6	7	8	9

Индексы элементов массива

Определение адреса элемента массива в памяти

- Зная порядковый номер элемента в массиве (его индекс) и тип элементов, можно легко определить адрес i-го элемента:

Адр i = Адр начала массива + i* длина типа эл-тов;

- Объем памяти, занимаемой одномерным массивом:

Кол-во байт=<размер типа эл-тов>*<кол-во эл.>

Обработка массивов – в циклах

- Если нужно произвести какие-либо действия с каждым элементом массива, то используют циклы.
- Например:

```
const N=10;  int A [N], i;
// заполнение массива значениями  с клавиатуры
for (i=0; i<N; i++) scanf (“%d”, &A[i]);

// вывод на экран значений элементов массива
for (i=0; i<N; i++) printf (“%d ”, A[i]);

// обнуление элементов массива
for (i=0; i<N; i++) A[i]=0;
```

```
for (i=0; i<N; i++) A[i]=i;
```

Обработка массивов. Пример 1 (1)

Имеется одномерный массив, содержащий 15 случайных целых чисел.

Найти среднее значение элементов этого массива

1. **Постановка задачи**

Исходные данные:

a - массив случайных натуральных чисел, заполняется с помощью датчика случайных чисел;

n – размер массива; ***n***=15.

Выходные данные

Sredn – среднее арифметическое эл-тов массива, действительное число; выводим на экран

Обработка массивов. Пример 1 (2)

2. Метод решения

Выделим подзадачи:

a) Заполнить массив

b) Вывести его на экран

c) Вычислить среднее

$$Sredn = \frac{\sum_{i=1}^{n} a_i}{n}$$

Датчики случайных чисел

- В Си имеются два датчика случайных чисел:
  - с параметром.
  - без параметра
- Датчик случайных чисел с параметром – это функция random.
  - возвращает целые случайные числа в интервале от 0 до параметр-1.

$$0 \le random(n) \le n-1$$

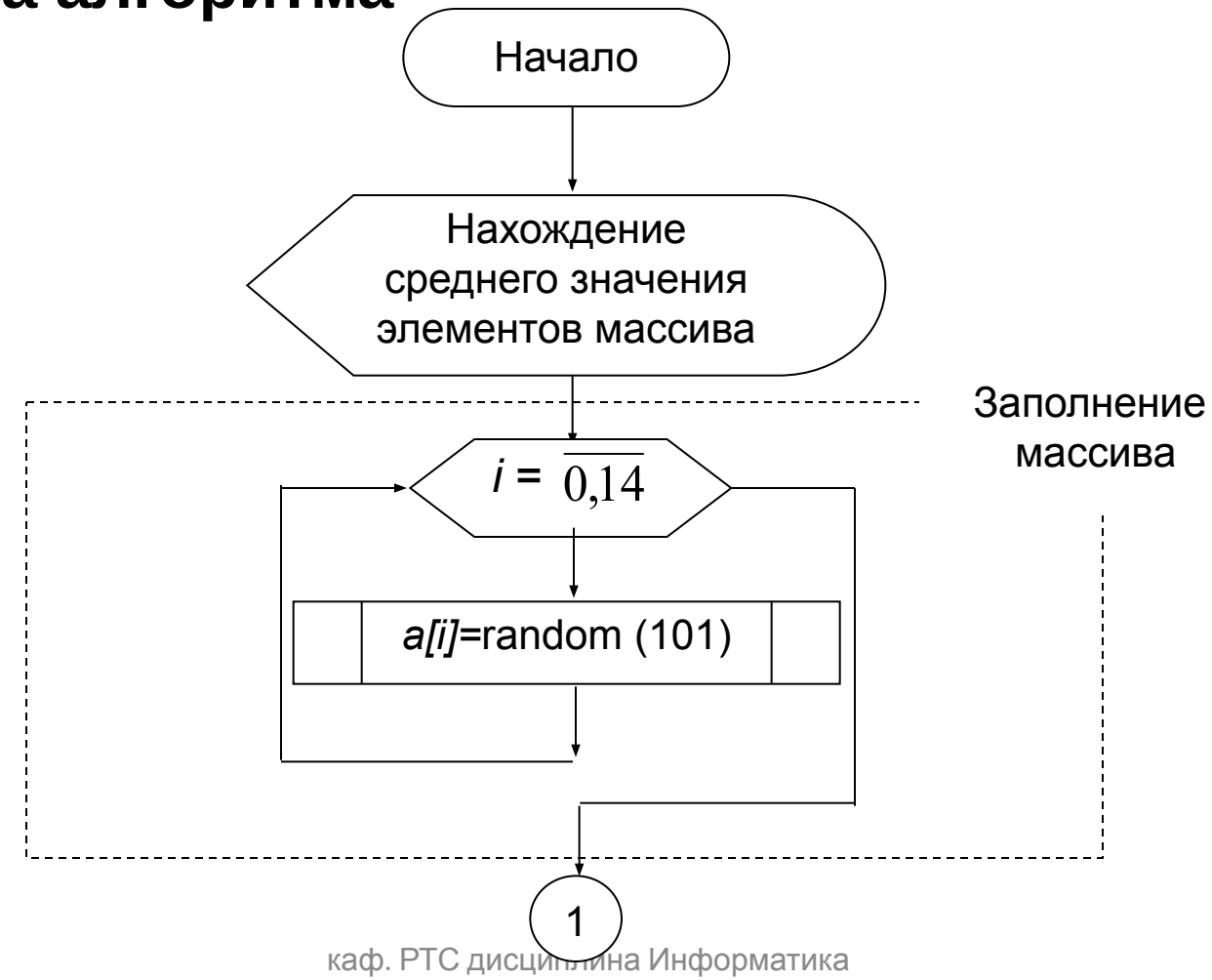
- Датчик случайных чисел без параметра – функция rand(),
  - возвращет целые положительные числа в интервале от 0 до RAND_MAX

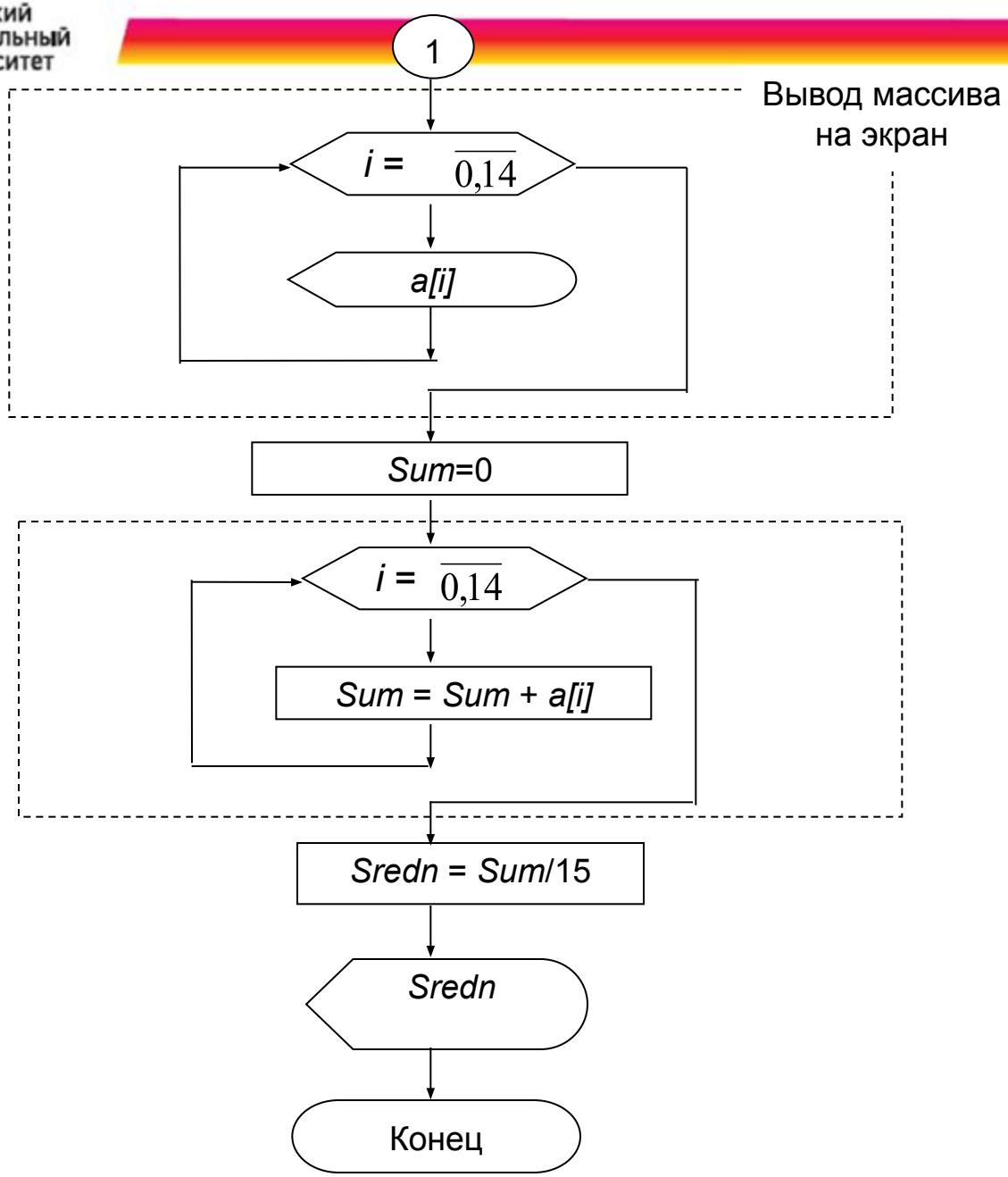
$$0 \le rand() \le RAND\_MAX$$

  - Значение константы RAND_MAX определяет максимально возможное целое число (типа int). Если тип int имеет длину 2 байта, то RAND_MAX=32767

Обработка массивов. Пример 1 (3)

3. Схема алгоритма

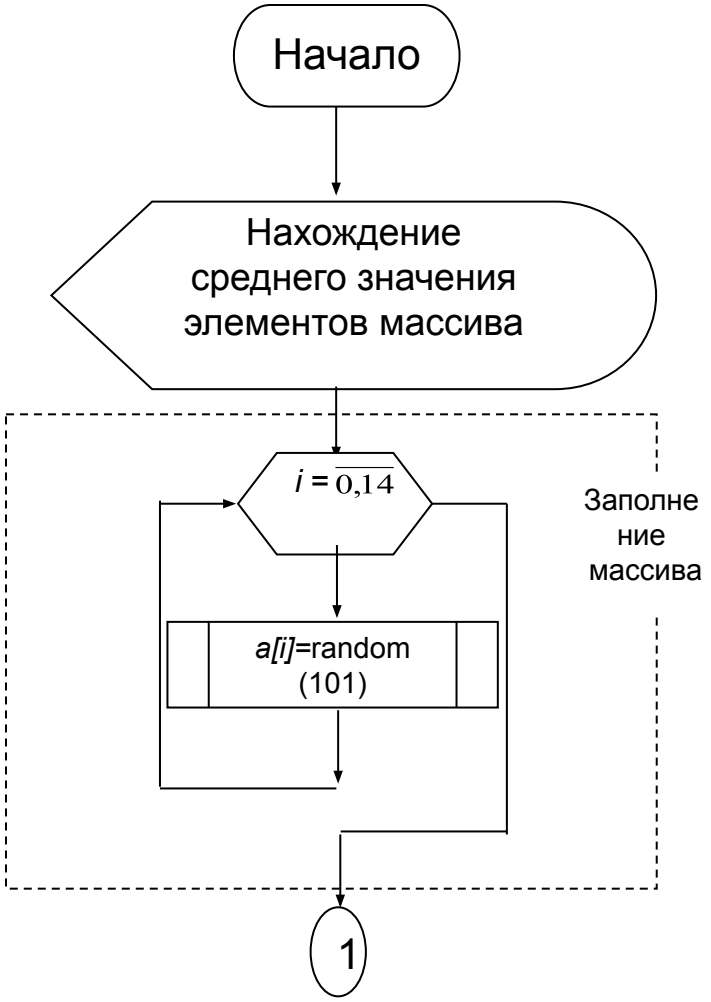



1
Вывод массива на экран
i = 0,14
a[i]
Sum=0
i = 0,14
Sum = Sum + a[i]
Sredn = Sum/15
Sredn
Конец

Обработка массивов. Пример 1 (4)

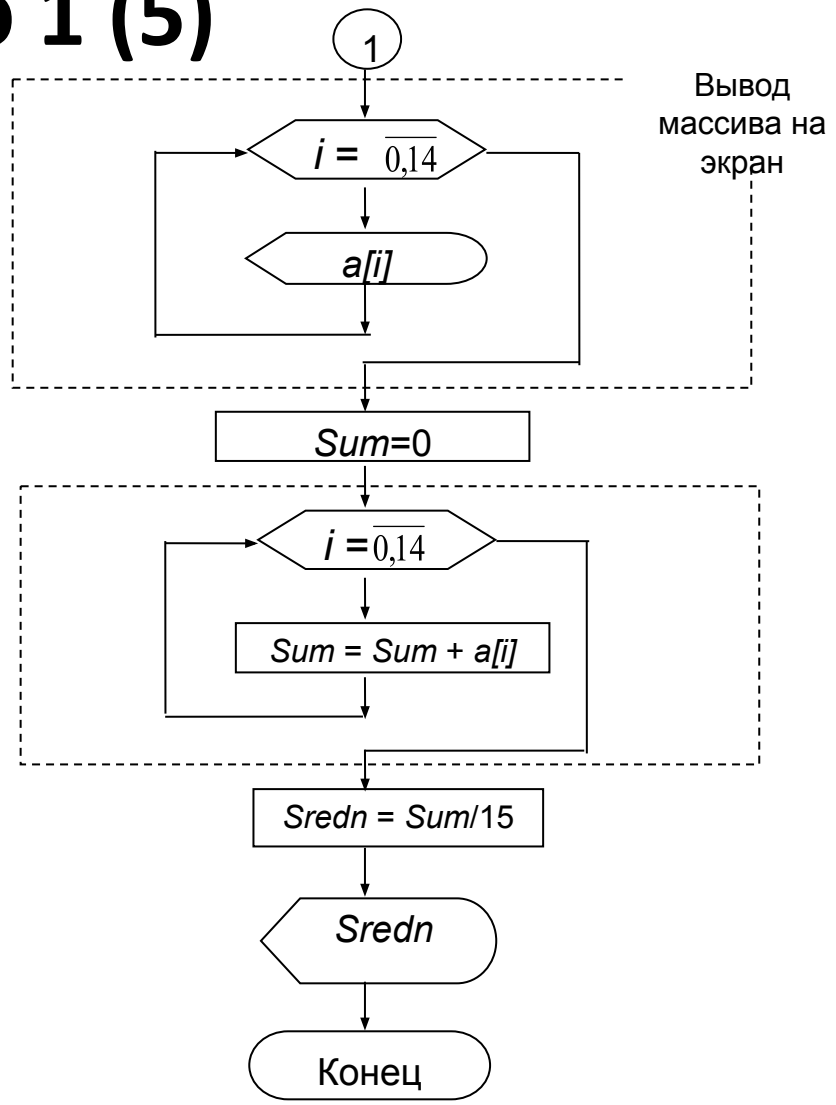
4. Текст программы

```
#include <stdio.h>
#include <stdlib.h>
 main()
{
int  Sum=0,  a[15];
float Sredn;

//заставка
printf  ("\nПрограмма вычисления среднего
      элементов
одномерного массива\n");



//заполнение массива
  for ( int i=0; i<15; i++)   a[i]=random(101);
```



Обработка массивов. Пример 1 (5)

```
//вывод массива на экран
printf ("\nИсходный массив случайных
       чисел\n");
 for (i=0; i<15; i++)   printf ("%d  ", a[i] );


  //вычисление суммы элементов массива
  for (i=0; i<15;   i++)   Sum+=a[i];

//вычисление среднего элементов массива
Sredn=Sum/15.0;

//вывод результата
printf ("\nСреднее значение = %f\n", Sredn);
return 0;
 }
```



Двумерные массивы

- Язык С допускает многомерные массивы

простейшая форма - двумерный массив (матрица).

- При описании двумерного массива
  - первый размер определяет количество строк,
  - второй - количество столбцов.

Можно сказать, что двумерный массив - это массив одномерных массивов.

Представление двумерного массива

Двумерный массив int a[3][4] можно представить в виде таблицы

a[0][0]	a[0][1]	a[0][2]	a[0][3]
a[1][0]	a[1][1]	a[1][2]	a[1][3]
a[2][0]	a[2][1]	a[2][2]	a[2][3]

Выделение памяти под матрицу

- Колич.байт =размер типа данных *колич. строк *колич. столбцов
- В памяти компьютера массив располагается непрерывно по строкам, т. е.

a[0][0], a[0][1], a[0][2], a[0][3], a[1][0], a[1][1], a[1][2], a[1][3], a[2][0], …, a[2][3].

Пример описания двумерных массивов

```
const N=4, M=5;
int A[N][M], B[N][N];
float X[10][5], Y[M][N];
```

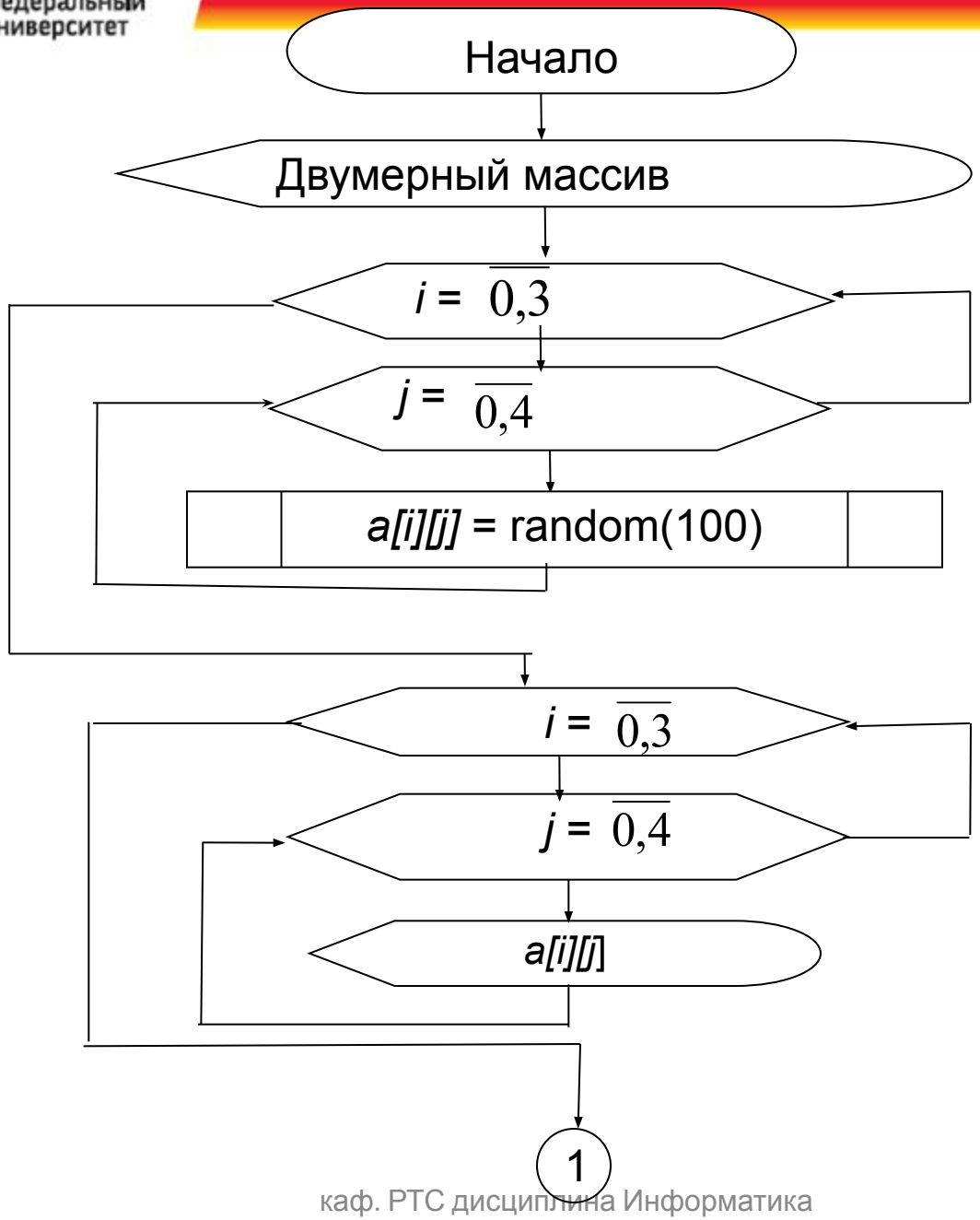
Обращение к элементам матрицы:

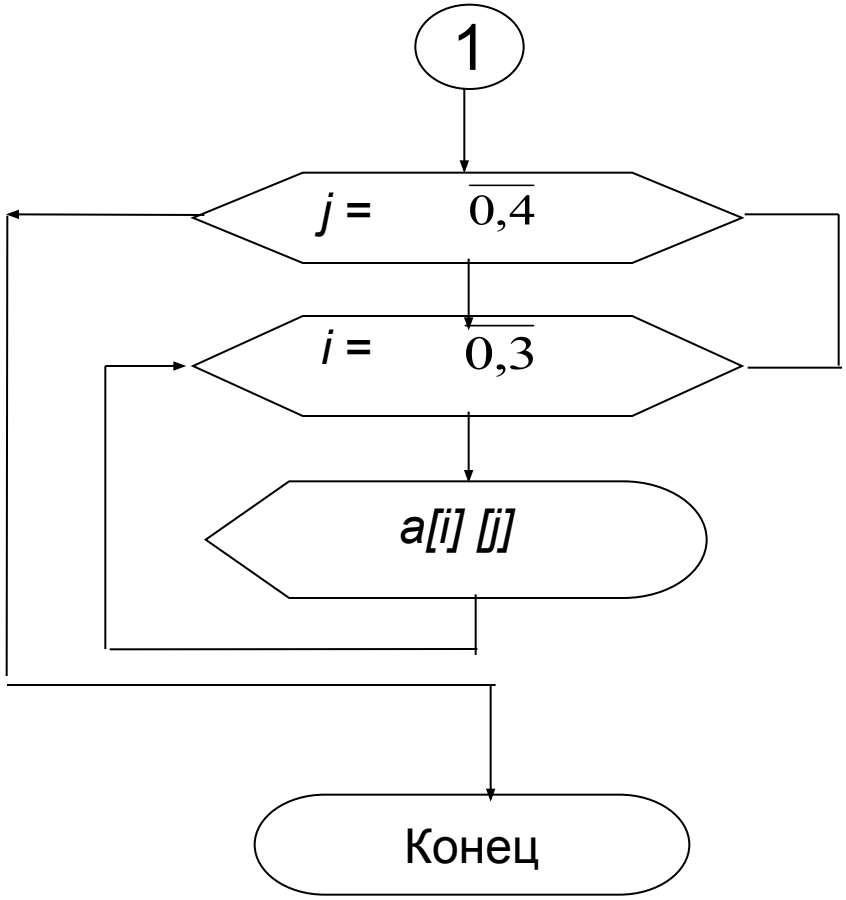
```
A[0][3]=8;   X[1][2]=0.678;
B[2][0]= A[0][3]*2;
Y[2][1]=Y[2][2]=5.5;
```

Примеры работы с матрицами (1)

Для однородной обработки каждого элемента матрицы используют вложенные циклы. Например,

```
//заполнение матрицы с помощью датчика случайных чисел
for (int i=0; i<N; i++)            //цикл по строкам
      for ( int j=0; j<M ; j++)    //цикл внутри строки  по столбцам
            A[i][j]=random (50);
//вывод  матрицы на экран
for ( i=0; i<M; i++)           //цикл по строкам
      { for ( j=0; j<N ; j++)            //цикл внутри строки  по столбцам
            printf  (“%5.1f “,Y[i][j] );
            printf  (“\n “ );          // переход на новую строку
      }
```


Начало
Двумерный массив
i = 0,3
j = 0,4
a[i][j] = random(100)
i = 0,3
j = 0,4
a[i][j]
1


1
j = 0,4
i = 0,3
a[i] [j]
Конец

Примеры работы с матрицами (2)

```
const M=6, N=5;
float Y[M][N];
int B[N][N];

//обнуление матрицы
for ( i=0; i<M; i++)            //цикл по строкам
      for (  j=0; j<N ; j++)          //цикл внутри строки  по столбцам
            Y[i][j]=0.0;

// заполнение единичной матрицы:
// элементы главной диагонали равны 1,
// все остальные элементы - 0
for ( i=0; i<N; i++)            //цикл по строкам
      for ( j=0; j<N ; j++)           //цикл внутри строки  по столбцам
            if (i==j) B[i][j]=1; else B[i][j]=0;
```

Пример 2. Поиск максимального. Постановка задачи.

Имеется одномерный массив, содержащий 20 натуральных случайных чисел. Найти элемент массива, содержащий максимальное число.

Постановка задачи

- *Исходными данными* для этой задачи является массив натуральных случайных чисел (формируется в ходе выполнения программы).
- *Выходными данными* является номер (*k*) максимального элемента в массиве и само значение этого элемента (все выводится на экран монитора).

Пример 2. Поиск максимального Метод решения.

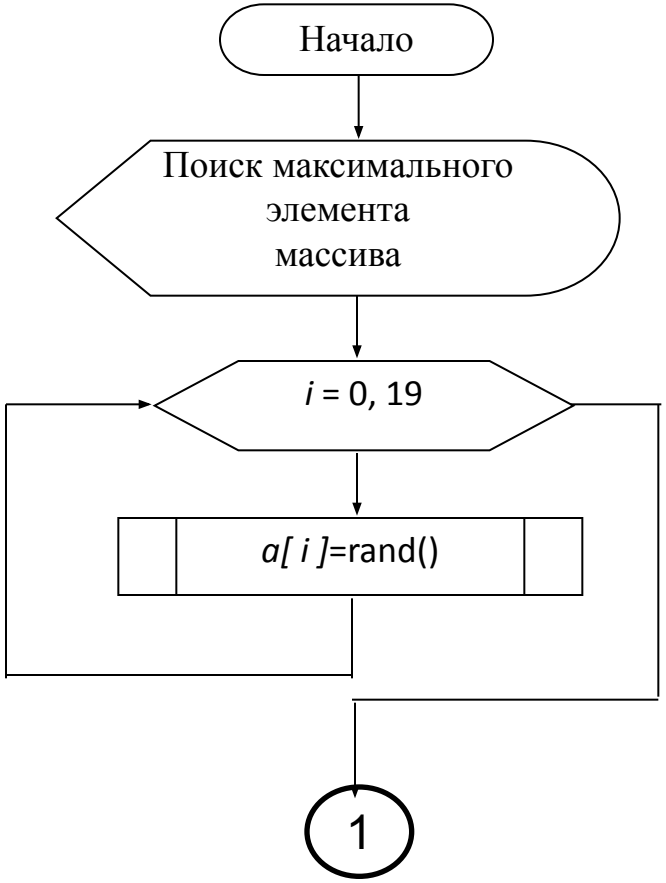
Метод решения

- В первую очередь необходимо заполнить массив натуральными числами с помощью датчика случайных чисел.
- После заполнения массива случайными числами будем производить попарное сравнение элементов массива, каждый раз запоминая номер большего

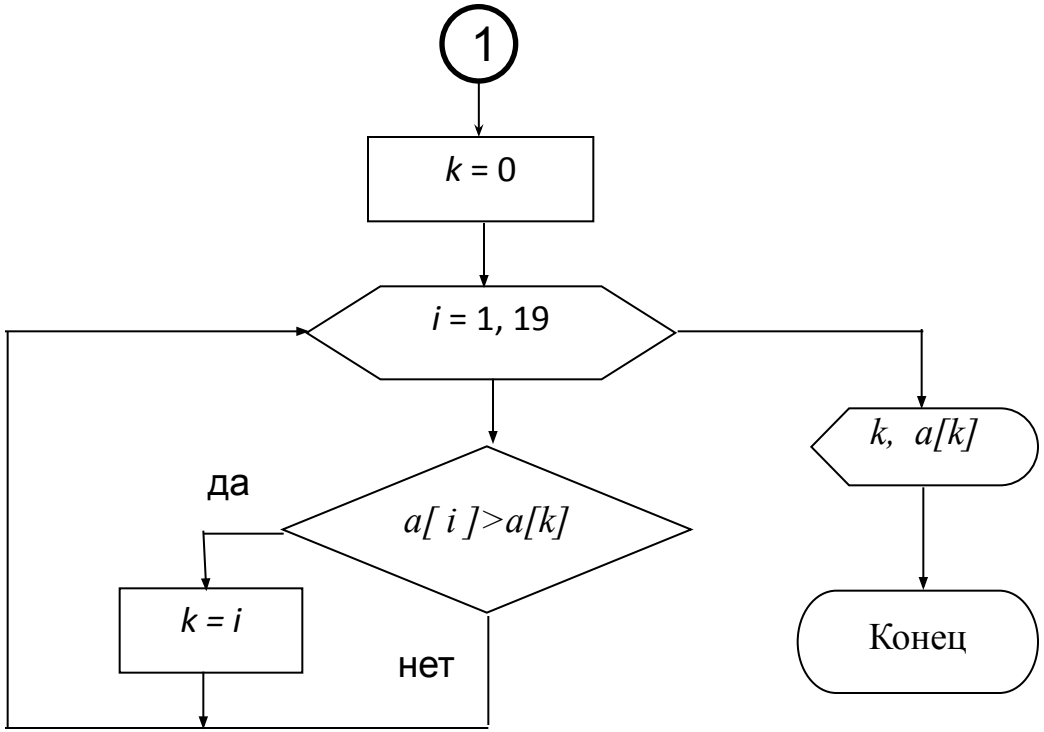
Пример 2. Поиск максимального Метод решения.

- Например: сравниваем первый и второй элементы.
- Пусть первый оказался больше второго, запомним его номер в переменной *k*. Далее сравниваем этот больший элемент с третьим элементом. Снова запоминаем номер большего элемента и т.д. до 20-го элемента. В результате по окончании процесса сравнения переменная *k* будет содержать номер максимального элемента данного массива.

Пример 2. Поиск максимального. Блок-схема.

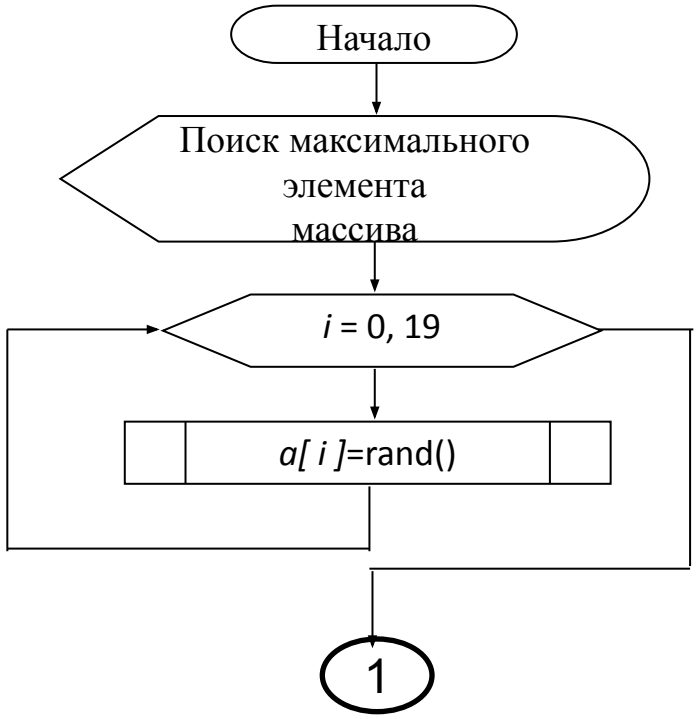


Пример 2. Поиск максимального. Блок-схема.



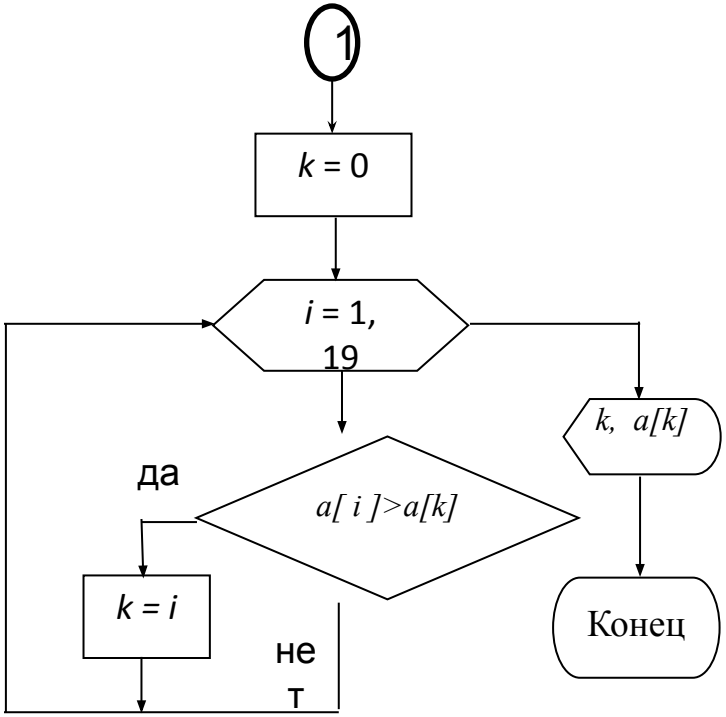
Пример 2. Поиск максимального. Текст программы.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
int  k,  a[20];

printf("\nПрограмма для поиска максимального элемента в
       массиве\n");
 //заполнение массива
  for (int i=0; i<20; i++)   a[i]=rand();

//вывод массива на экран
printf("\nИсходный массив случайных чисел\n\n");
for (i=0; i<20; i++)   printf("%d  ", a[i] );
```



Пример 2. Поиск максимального. Текст программы.

```
//поиск максимального элемента
k=0;
for (i=1; i<20; i++)   if (a[i]>a[k]) k=i;
//вывод результатов
printf("\n\n Максимальный элемент
     a[%d]=%d", k, a[k]);
return 0;
 }
```



Пример 3. Сортировка массива по убыванию. Постановка задачи.

Имеется одномерный массив, содержащий 20 натуральных случайных чисел (аналогично прим. 4.2). Расположить элементы массива в порядке убывания их значений.

Постановка задачи

- *Исходными данными* для этой задачи является массив натуральных случайных чисел (формируется в ходе выполнения программы).
- *Выходными данными* является этот же массив натуральных случайных чисел, упорядоченный по убыванию (элементы массива выводятся на экран монитора).

Пример 3. Сортировка массива по убыванию. Метод решения.

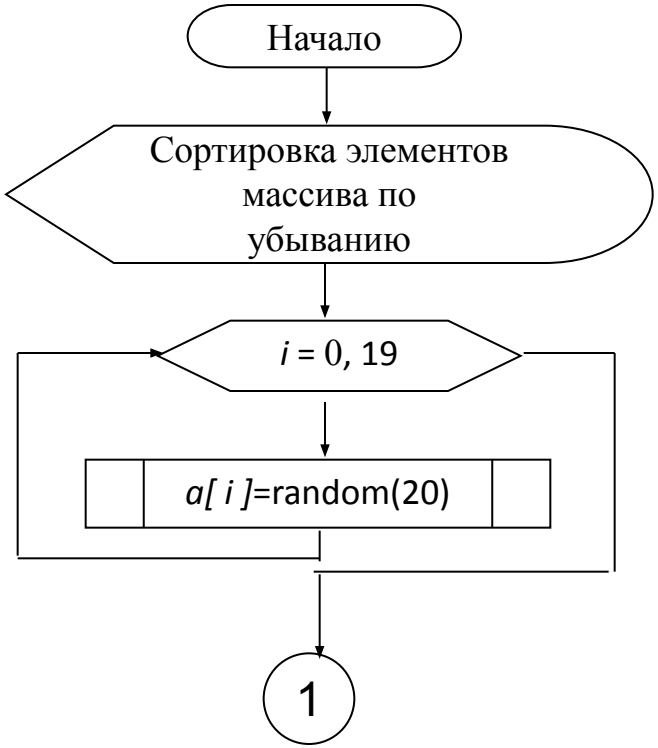
Метод решения

- Заполним массив натуральными числами с помощью датчика случайных чисел с параметром.
- Выполним процесс упорядочения:
  - Сначала просмотрим весь массив и выберем максимальный элемент. Поместим его в начало массива, т.е. поменяем местами этот найденный максимальный элемент с первым элементом массива. Для этого введем дополнительную (буферную) переменную, в которой сначала сохраним значение первого элемента массива.
  - Затем присвоим первому элементу значение максимального, а тому элементу, который содержал максимальный, присвоим значение буферной переменной.

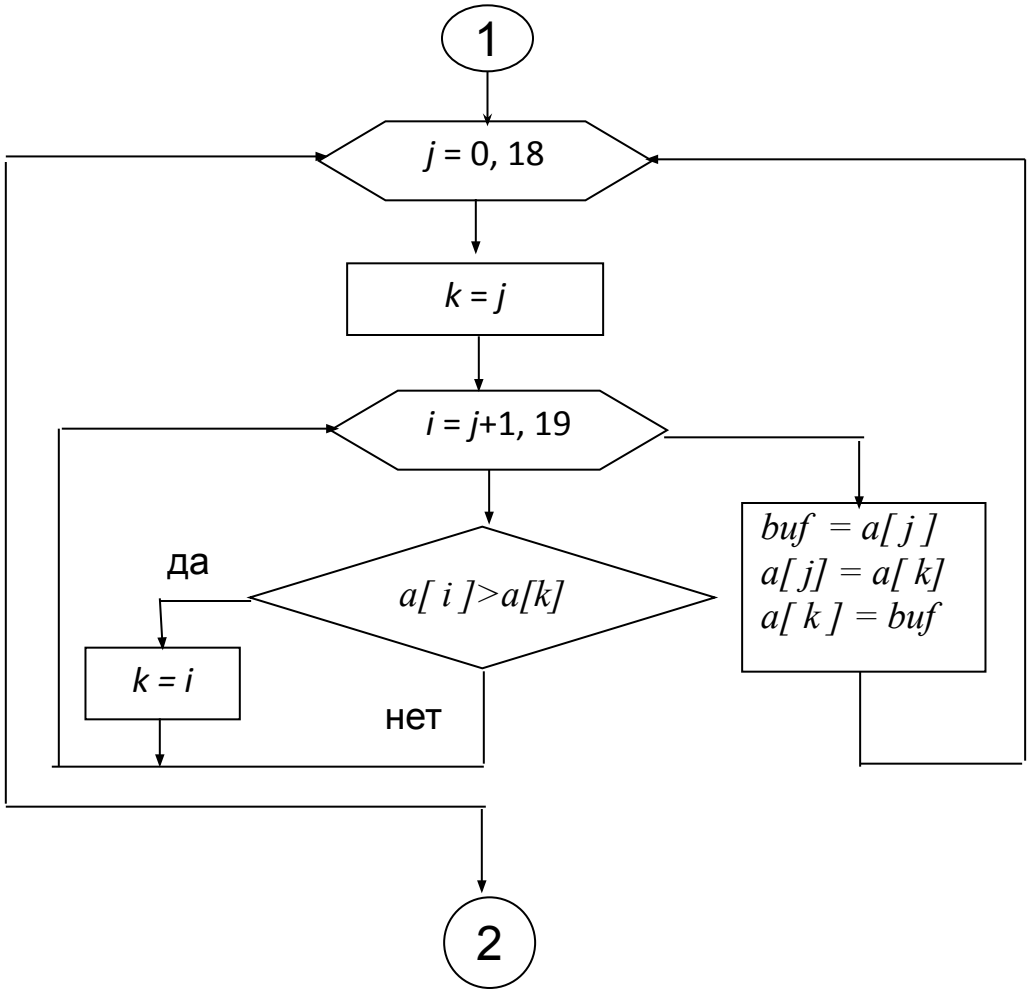
Пример 3. Сортировка массива по убыванию. Метод решения.

- После этой перестановки просмотрим оставшуюся часть массива, начиная со второго до 20, и снова выберем максимальный элемент. Поместим его на второе место в массиве, поменяв местами со вторым элементом. Снова просмотрим оставшуюся часть массива, начиная с третьего элемента, и т.д. до конца. В итоге получим массив, упорядоченный по убыванию значений элементов.
- Упорядочение по возрастанию значений элементов выполняется аналогично, только при каждом просмотре ищется не максимальный, а минимальный элемент.

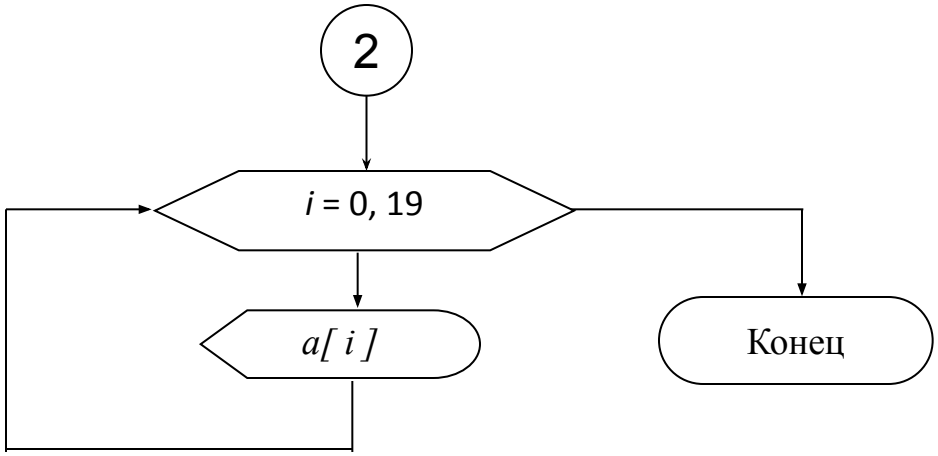
Пример 3. Сортировка массива по убыванию. Блок-схема.



Пример 3. Сортировка массива по убыванию. Блок-схема.

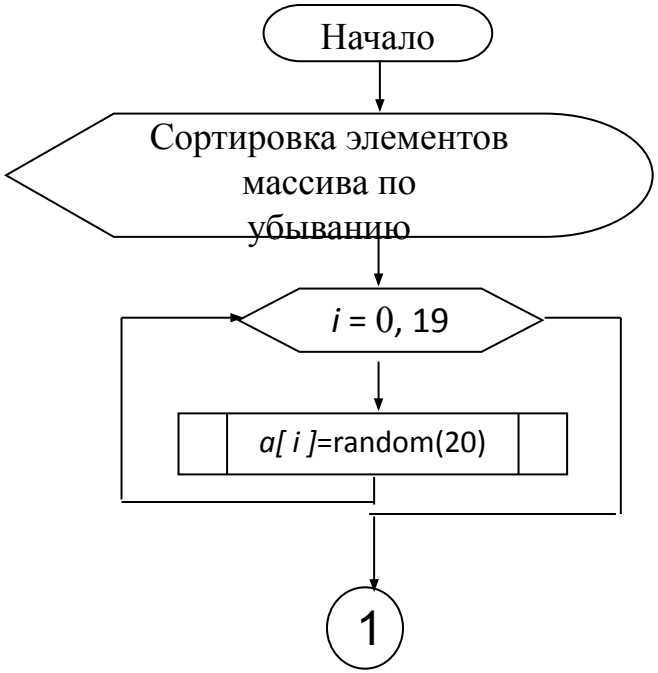


Пример 3. Сортировка массива по убыванию. Блок-схема.



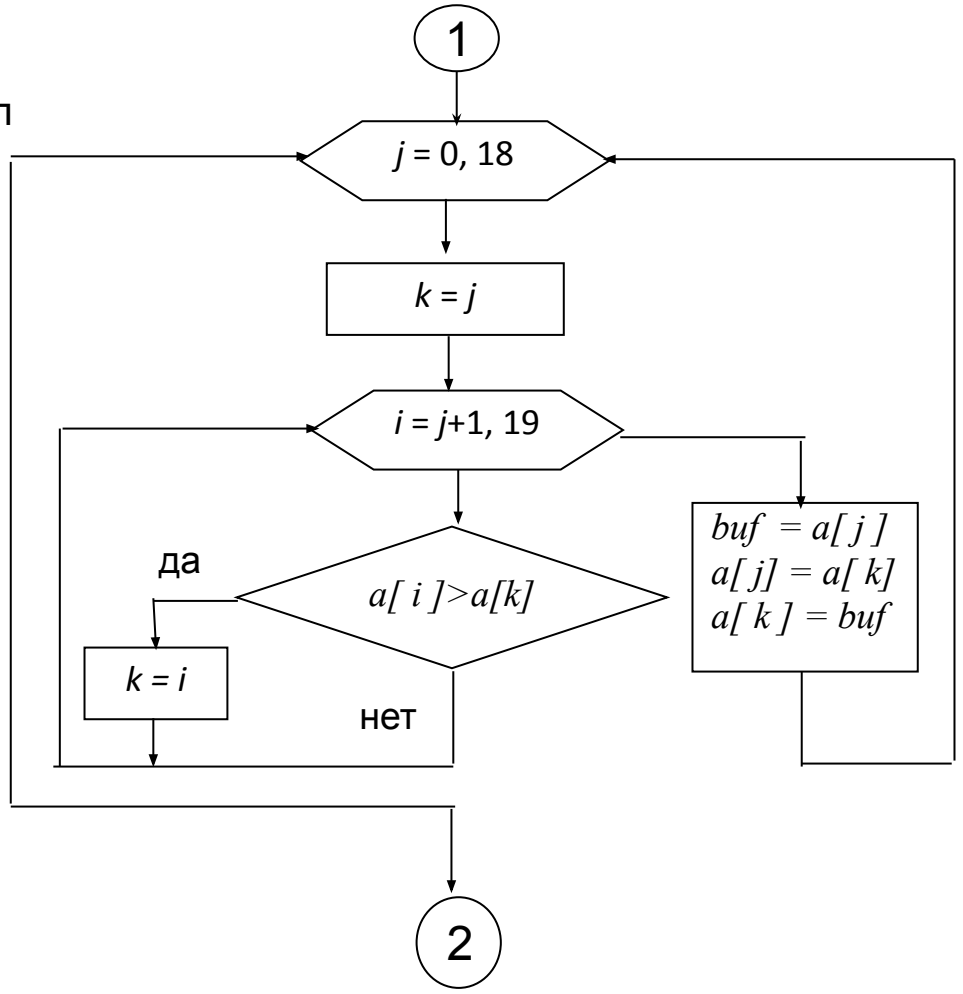
Пример 3. Сортировка массива по убыванию. Текст программы.

```
#include <stdio.h>
#include <stdlib.h>

main(){
int buf, k, a[20];
printf("\n Сортировка массива
    по убыванию \n");
//заполнение массива
 for (int i=0; i<20; i++)
    a[i]=random(100);
```



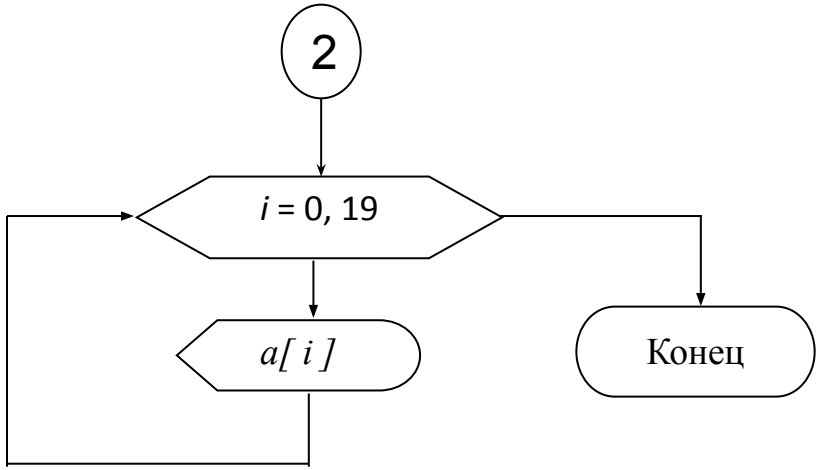
Пример 3. Сортировка массива по убыванию. Текст программы.

```
//вывод массива на экран
printf("\n Исходный массив случайных чисел
      \n\n");
for (i=0; i<20; i++)   printf ("%d  ", a[i] );
//сортировка
for (int j=0; j<19; j++)
   {
       k=j;
       //поиск максимального элемента
       for (i=j+1; i<20; i++) if (a[i]>a[k]) k=i;
       // перестановка элементов
       buf=a[j];
       a[j]=a[k];
       a[k]=buf;
    }
```



Пример 3. Сортировка массива по убыванию. Текст программы.

```
//вывод отсортированного массива
    на экран
printf ("\n\n Упорядоченный массив
    случайных чисел \n\n");
for (i=0; i<20; i++)   printf("%d ", a[i] );
return 0;
 }
```



Итоги
Рассмотренные вопросы:

- Составные типы в языке С
- Одномерные массивы
- Двумерные массивы
- Описание массивов
- Доступ к отдельным элементам массива

Определение некоторых понятий

Динамическая память – объекты, память для которых распределяется при вхождении в блок и высвобождается при выходе из блока.

Статическая память- объекты, жизненный цикл которых продолжается от запуска программы до ее завершения, и которые инициализируются до запуска программы.

Тип массива- тип объекта, состоящий из нескольких однотипных объектов, называемых элементами массива.

Упорядоченная последовательность- последовательность, упорядоченная по предикату таким образом, что ее элементы, расположенные ранее любого фиксированного элемента, упорядочены до него.

Библиографический список

- Подбельский В.В. Язык СИ++. Учебное пособие. М.: Финансы и статистика, 2003. – 560 с.
- Павловская Т.А. C/C++. Программирование на языке высокого уровня: учебник для студентов вузов, обучающихся по направлению "Информатика и вычисл. техника" СПб.: Питер, 2005. - 461 с.
- Березин Б.И. Начальный курс C и C++ / Б.И. Березин, С. Б. Березин. - М.: ДИАЛОГ-МИФИ, 2001. - 288 с
- Каширин И.Ю., Новичков В.С. От С к С++. Учебное пособие для вузов. – М.: Горячая линия – Телеком, 2005. – 334 с.

Автор:

Саблина Наталья Григорьевна

Ст. преподаватель

каф. РТС УГТУ-УПИ