Информатика и программирование

Массивы

Понятие массива

Массив – конечная последовательность однотипных величин, имеющая общее имя.

Аналог понятия массива – **вектор**.

Формат описания массива:

```
тип имя_переменной[размер];
```

тип – тип элементов, хранящихся в массиве;

имя_переменной – название массива;

размер – константное положительное целочисленное выражение, определяющее количество элементов массива.

Элементы массива размещаются в памяти **последовательно.**

Примеры описания массивов

```
const int MAX_SIZE = 20;
const int MAX_LENGTH = 50;

int values[100];
double mas[MAX_SIZE];
short int a[2 * MAX_SIZE];
char s[MAX_LENGTH + 1];
```

Нумерация элементов массива

Все элементы массива нумеруются с 0 до размера массива минус 1

Пример

```
int mas[5];
```

Элементы массива располагаются в памяти **последовательно** – один за другим.

Доступ к элементу массива осуществляется по его номеру (**индексу**).

Доступ к элементам массива по индексу (присвоение значения)

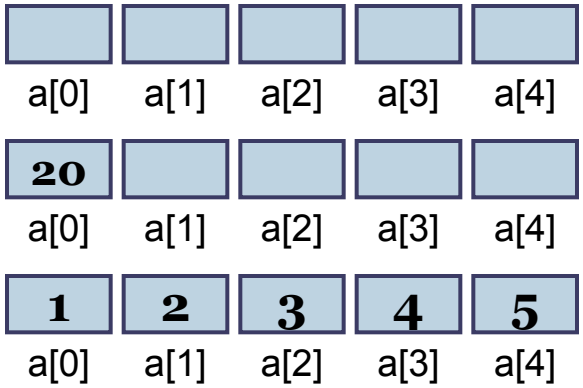
Присвоение значения элементу массива осуществляется аналогично тому, как это делается с обычной (скалярной) переменной, только добавляются квадратные скобки [] с индексом

Формат записи:

```
имя_переменной[индекс] = выражение;
```

Пример

```
int a[5];
a[0] = 20;
for (int i = 0; i < 5; i++)
  a[i] = i + 1;
```



Доступ к элементам массива по индексу (чтение значения)

Чтение значения элемента массива осуществляется аналогично чтению значения обычной переменной.

Пример:

```
int fib[20];
fib[0] = fib[1] = 1;
for (int i = 2; i < 20; i++)
  fib[i] = fib[i – 1] + fib[i – 2];
```

Инициализация массива

С указанием размера массива

```
//mas[0] = 2, mas[1] = 4, mas[2] = 8, mas[3] = 16
//mas[4] = … = mas[9] = 0
int mas[10] = {2, 4, 8, 16};

//d[0] = 12.4, d[1] = 3.45, d[2] = 1.0, d[3] = 3.2
double d[4] = {12.4, 3.45, 1.0, 3.2};
```

Без указания размера массива

```
long a[] = {-2, -1, 0, 1, 2};
```

Основные операции над массивами

- Ввод элементов массива с консоли
- Задание элементам массива случайных значений
- Определение размера массива
- Печать элементов на экран
- Поиск минимального (максимального) элемента
- Сортировка элементов массива

Ввод элементов массива с консоли

Пример

```
const int MAX_SIZE = 20;

int mas[MAX_SIZE];
int n;
cin >> n;
for (int i = 0; i < n; i++)
  cin >> mas[i];
```

Задание элементам массива случайных значений

Пример

```
#include <stdlib.h>
#include <time.h>
…
srand(time(NULL));
for (int i = 0; i < n; i++)
{
  mas[i] = rand() % 100;
}
```

Справка по функциям:

`void srand(unsigned int seed);`

- задает начальное значение для последовательности псевдослучайных чисел.

`int rand();`

- возвращает следующее псевдослучайное число из диапазона `0` до `RAND_MAX` (`32767`).

`time_t time(time_t *timer);`

- возвращает количество секунд, прошедших с 0:00:00 1 января 1970 г.

Определение размера массива

Пример

```
int values[] = {5, 33, 22, 4, 5, 6, 7};
int valuesCount = sizeof(values) / sizeof(int);
cout << valuesCount << endl;
```

Печать элементов на экран

Пример

```
int mas[100];
int n; //Количество элементов в массиве
……
cout << “mas : “ << endl;
for (int i = 0; i < n; i++)
  cout << mas[i] << “ “;
cout << endl;
```

Поиск минимального (максимального) элемента

Пример

```
int min = mas[0];
for (int i = 1; i < n; i++)
{
  if (mas[i] < min)
  {
   min = mas[i];
  }
}
cout << "min = " << min << endl;
```

Нахождение максимального элемента в массиве осуществляется аналогично.

Сортировка элементов массива (метод выбора)

```
for (int i = 0; i < n - 1; i++)
{
  //Поиск минимального элемента среди mas[i],...,mas[n-1]
  int minIndex = i;
  for (int j = i + 1; j < n; j++)
  {
   if (mas[j] < mas[minIndex])
      minIndex = j;
  }
  //Обмен местами элементов mas[i] и mas[minIndex]
  int tmpValue = mas[i];
  mas[i] = mas[minIndex];
  mas[minIndex] = tmpValue;
}
```

Переменная массива, как константный указатель

Переменная массива с точки зрения языка программирования C++ рассматривается, как **константный указатель на нулевой элемент массива.**

```
int a[10];
a == &a[0]; //Всегда истинное утверждение
```

Возможность индексации через указатель.

Выражения **a[i] и *(a + i) – эквиваленты**.

Многомерные массивы

Многомерные массивы – это массивы, элементы которых могут быть в свою очередь массивами.

Формат описания:

```
тип имя_переменной[размер1][размер2]…[размерN];
```

Пример:

```
double matrix[5][6];
int cube[7][7][7];
```

Доступ к элементам многомерного массива

Обращение к элементу с индексами (i1, i2, .., iN) в многомерном массиве осуществляется с помощью следующего выражения:

имя_переменной[i1][i2]…[iN]

Индекс каждого измерения меняется от 0 до размера - 1.

Пример:

```
int a[3][3];
for (int i = 0; i < 3; i++)
  for (int j = 0; j < 3; j++)
   a[i][j] = i * 3 + j;
```

0	1	2
a[0][0	a[0][1	a[0][2
3	4	5
a[1][0	a[1][1	a[1][2
6	7	8
a[2][0	a[2][1	a[2][2

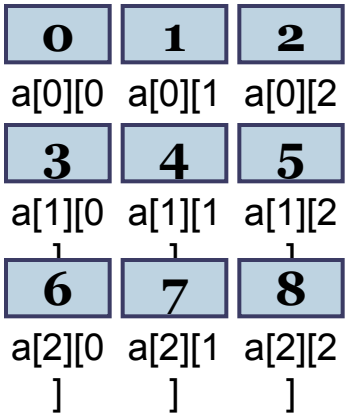
Расположение многомерных массивов в памяти

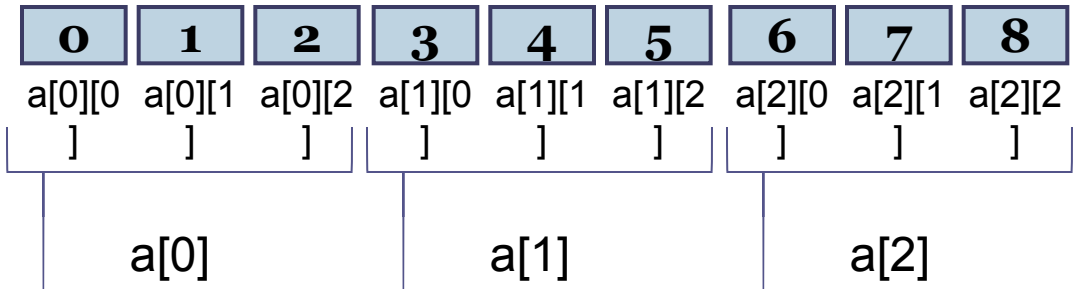
Пример для двумерного массива:

int a[3][3];

Логическая структура массива

Физическое расположение массива в памяти





Строки двумерного массива располагаются в памяти **последовательно** друг за другом.

Пример – Печать всех элементов массива с помощью указателя

```
int m[2][2] = { {1, 2}, {3, 4} };
for (int* p = (int*)m; p < (int*)m + 4; p++)
  cout << *p << " ";
cout << endl;
```

Инициализация многомерных массивов

Инициализация с группировкой значений по измерениям

```
int mas[3][2] = { { 0, -2}, {1, 1}, {-5, 45} };
```

Инициализация без группировки значений по измерениям

```
int mas1[3][2] = { 0, -2, 1, 1, -5, 45 };
```

Переменная двумерного массива, как константный указатель

Переменная двумерного массива с точки зрения языка программирования C++ рассматривается, как **константный указатель на элемент [0][0].**

```
int a[10][10];
a == &a[0][0]; //Всегда истинное
  утверждение
```

Возможность индексации через указатель.

Выражения **a[i][j] и *(*(a + i) + j) – эквиваленты**.

Пример (вычисления суммы двух матриц) – шаг 1

Объявляем константы максимальных размеров матриц и двумерные массивы для матриц A, B и C:

```
……
const int MAX_ROW_COUNT = 10;
const int MAX_COL_COUNT = 10;

int main()
{
  int n, m;
  int masA[MAX_ROW_COUNT][MAX_COL_COUNT],
  masB[MAX_ROW_COUNT][MAX_COL_COUNT],
  masC[MAX_ROW_COUNT][MAX_COL_COUNT];
```

Пример (вычисления суммы двух матриц) – шаг 2

Вводим размеры и элементы матриц A и B:

```
cout << "n = ";  cin >> n;
cout << "m = ";  cin >> m;
cout << "Enter matrix A :" << endl;
for (int i = 0; i < n; i++)
  for (int j = 0; j < m; j++)
   cin >> masA[i][j];
cout << "Enter matrix B :" << endl;
for (int i = 0; i < n; i++)
  for (int j = 0; j < m; j++)
   cin >> masB[i][j];
```

Пример (вычисления суммы двух матриц) – шаг 3

Вычисление суммы двух матриц и вывод результатов:

```
for (int i = 0; i < n; i++)
  for (int j = 0; j < m; j++)
   masC[i][j] = masA[i][j] + masB[i][j];
cout << "matrix C = A + B :" << endl;
for (int i = 0; i < n; i++)
{
  for (int j = 0; j < m; j++)
   cout << masC[i][j] << " ";
  cout << endl;
}
return 0;
} //Main
```

Динамические массивы

Динамические массивы – это массивы, размеры которых могут задаваться в процессе исполнения программы, описываются как указатели и создаются с помощью операции new.

Создание динамического массива

```
тип* переменная_указатель = new тип [размер];
```

Память под динамические массивы выделяется в куче.

Освобождение памяти динамического массива

```
delete[] переменная_указатель;
```

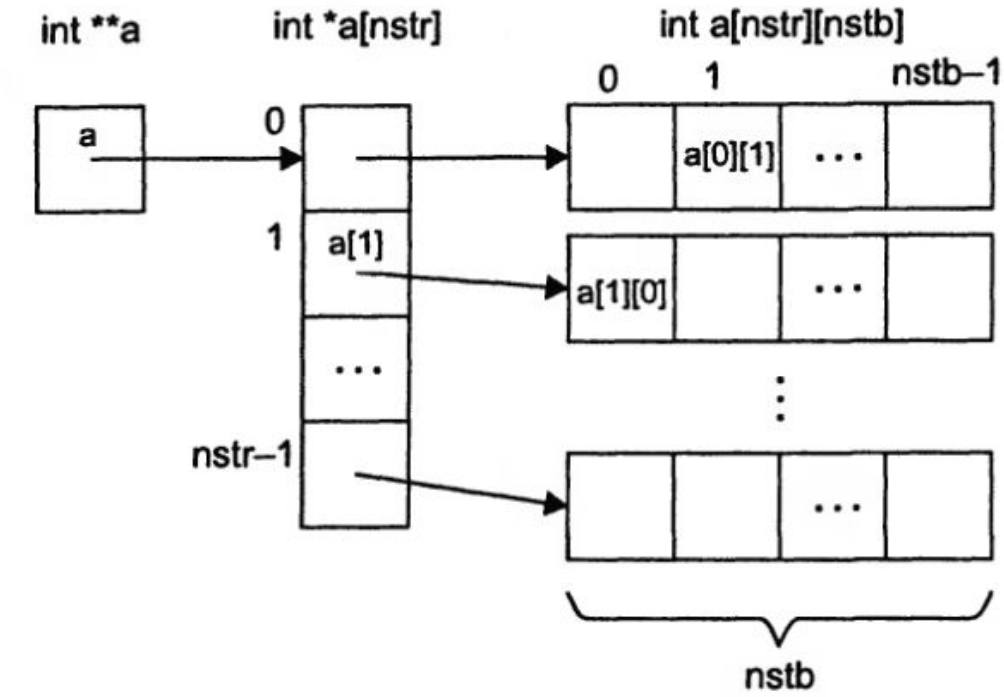
Пример использования динамических массивов

```
int n;
cin >> n;
int* mas = new int [n];
for (int i = 0; i < n; i++)
  mas[i] = i*i;
for (int i = 0; i < n; i++)
  cout << mas[i] << “ ”;
cout << endl;
delete[] mas;
```

Многомерные динамические массивы

Пример

```
int nstr = 3;
int nstb = 5;
int ** a = new int* [nstr];
for (int i = 0; i < nstr; i++)
   a[i] = new int [nstb];
for (int i = 0; i < nstr; i++)
   for (int j = 0; j < nstb; j++)
    a[i][j] = i – j;
….
for (int i = 0; i < nstr; i++)
   delete[] a[i];
delete[] a;
```



Типичные ошибки при работе с массивами

- **Выход за границу массива**

```
int a[10];
for (int i = 0; i <= 10; i++)
 a[i] = i + 1;
```

- **Неправильное освобождение памяти для динамических массивов**

```
int* a = new int [10];
…
delete a;
```

- **Отсутствие освобождения памяти для динамических массивов**

```
void f() {
 double * mas = new double [20];
}
```

Вопросы?

- Q&A