Массивы

Бекарыстанкызы Акбаян

- Массив – структура взаимосвязанных элементов и последовательная группа ячеек памяти
- Элементы хранятся по одним именем и типом
- int c[10];
- string l[200];

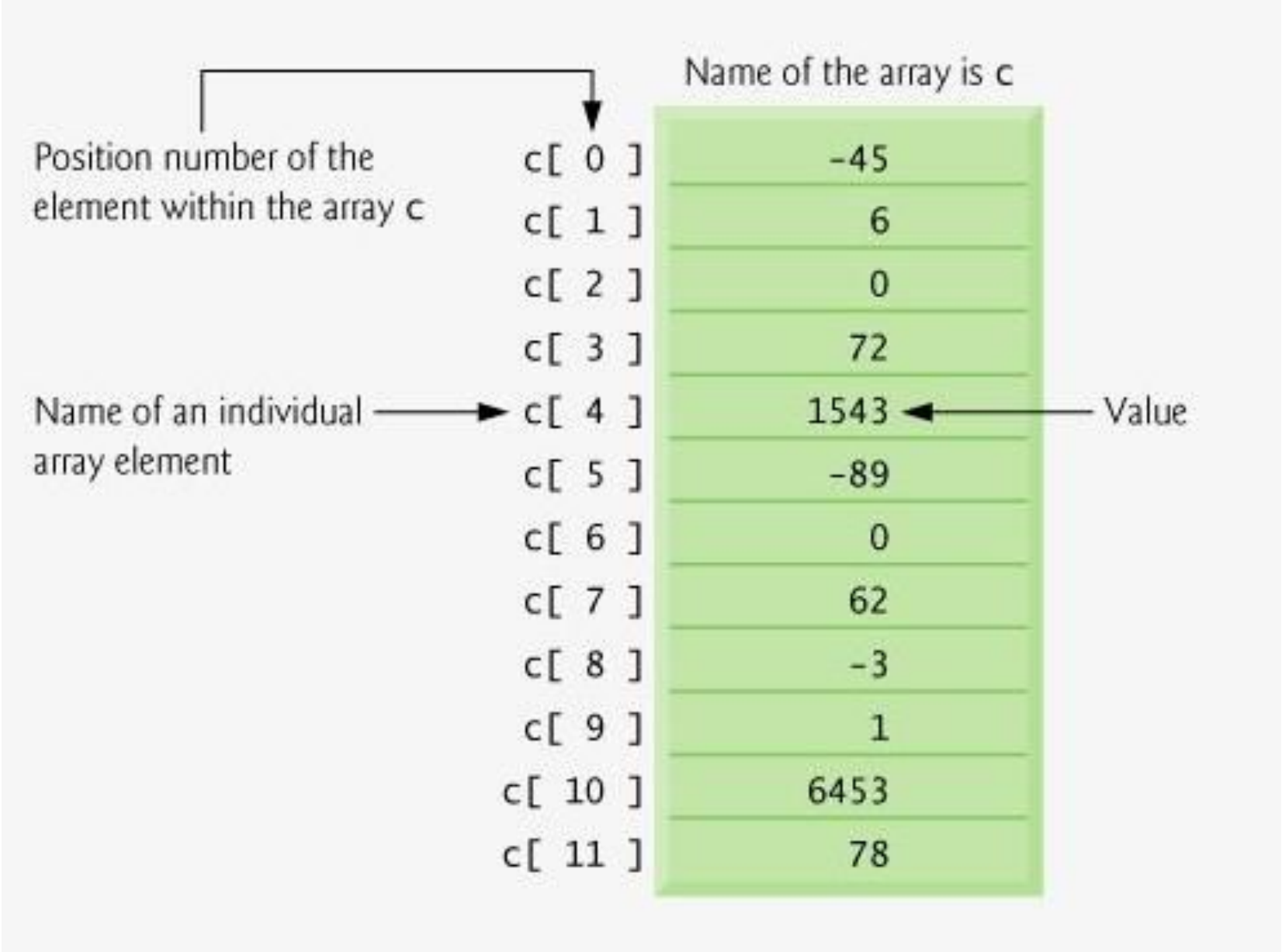
Объявление массива

- Тип название [размер]

- Размер должен задаваться константой, то есть:

1. int arr[10];

2. const int n = 10;
   int arr[n];


Name of the array is c
Position number of the element within the array c
Name of an individual array element
Value
c[0] -45
c[1] 6
c[2] 0
c[3] 72
c[4] 1543
c[5] -89
c[6] 0
c[7] 62
c[8] -3
c[9] 1
c[10] 6453
c[11] 78

Инициализация массивов

- Цикл For

- Определите каждый элемент при объявлении массива
- **int n[5] = { 1, 2, 3, 4, 5 };**

- Если не указан размер массива, инициализаторы определяют его размер

- **int n[] = { 1, 2, 3, 4, 5 };**
- 5 инициализаторов и поэтому 5 элементов
-

Доступ к элементам массива

- Индексирование в C++ начинается с 0
- Указывается имя массива, затем в квадратных скобках порядковый номер элемента в массиве

```
x=a[5];
```

Форматирование вывода

- #include <iomanip>
- using std::setw;
- int main(){
- //lines of code
- Cout<<a<<setw(10)<<b<<“\n”;
- //lines of code
- }

Строки как массивы

- Strings
  Массив символов
  Все строки заканчиваются на **null ('\0')**
- Примеры
  char string1[] = "hello";
    Неявно добавляется символ **Null**
    Строка имеет 6 элементов
  char string1[] = { 'h', 'e', 'l', 'l', 'o', '\0' };
- Индексирование то же самое
  String1[0] - 'h'
  string1[2] - 'l'

Строки как массивы

- Ввод из клавиатуры

```
char string2[ 10 ];
cin >> string2;
```

  Добавляет ввод пользователя в строку

  Останавливается при первой встрече пробела

  Добавляет символ **null**

  Если вводится очень много текста, то он может выйти за разделы нашего массива

  Этого нужно избегать

- Вывод строк

```
cout << string2 << endl;
```

  Не работает для других типов массивов

  Символы выводятся до нахождения символа **null**

Ввод строк с пробелами

```
int main (){
const int a=10;
char str[a];
cin.getline(str, 10, ',');
cout<<str;

  return 0;
}
```

Swap(перестановка)

- **a<->b**

- //Значение в a записываем в temp
- temp = a;
- // Значение b a записываем в a
- a = b;
- // Значение temp a записываем в b
- b = temp;

Сортировка массива

- n2
- for (int pass = 0; pass < 9; pass++)
- for(int i=0; i<9; i++)
- if (a[i]>a[i+1])
- {temp = a[i];
- a[i] = a[i+1];
- a[i+1] = temp;}

12 1 3 4 13 5 1 3

1 12 3 4 5 13 1 3

Поиск в массиве

```
for(int i=0; i<10; i++)
if (a[i]==temp) c++;
```

Многомерные массивы

- Множественные индексы

  a[i][j]
- Таблицы с колонками и строками
- Определяется строка, затем колонка
- «Массивы массивов»
- a[0] – массивы из 4-х элементов
- a[0][0] – первый элемент этого массива

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Column subscript

Array name

Row subscript

Инициализация

- **int b[2][2] = { { 1, 2 }, { 3, 4 } };**

1	2
3	4

- **intb[2][2] = { { 1 }, { 3, 4 } };**

1	0
3	4

Массивы как параметры функции

- Определяется имя без скобок

  Передать массив **myArray** в **myFunction**

  int myArray[24];

  myFunction(myArray, 24);

- Иногда передается размер, но это не обязательно

- Но это полезно для прохождения по всем элементам массива

Массивы как параметры функции

- Массивы передаются через ссылки
- Функции могут модифицировать изначальные данные в массиве
- Название массива ссылается на его первый элемент
- Функции знают местонахождение массива
- Может менять исходное местоположение массива

- Элементы массива могут быть переданы по их значению

  Как обычные переменные

  square(myArray[3]);

Передача массивов в функции

- Прототип функции
- **void modifyArray(int b[], int arraySize);**
- **void modifyArray(int[], int);**

- Можно не указывать размер массива в скобках
  - Игнорируется компилятором
- Если объявит параметр массива как **const**
- Не может быть изменен(ошибка компиляции)
- **void doNotModify(const int[]);**

Многомерные массивы и функции

- **intfunc(int a[][7]);**