



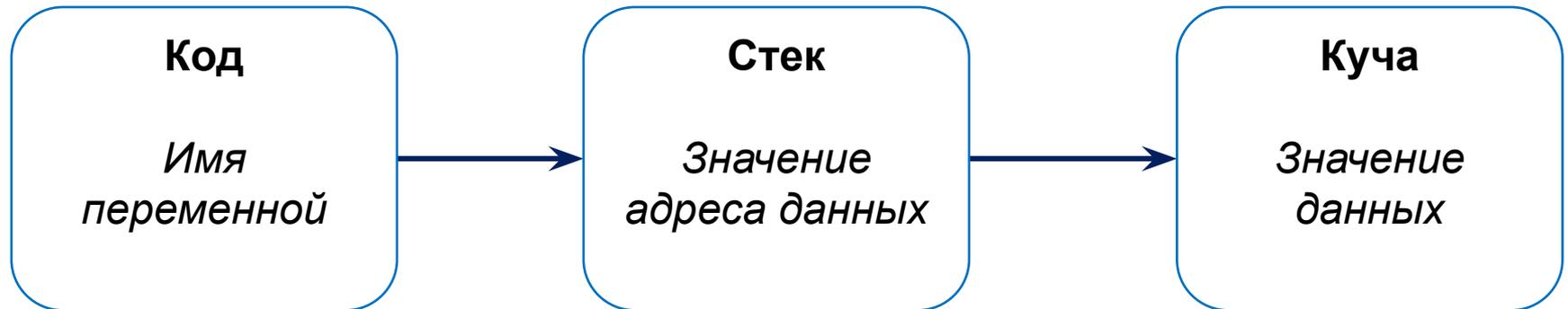
NATIONAL RESEARCH
UNIVERSITY

Введение в программирование

Лекция 4

Одномерные массивы

Массив - ссылочный тип



Объекты ссылочных типов размещаются в «куче»
[managed heap]



Массивы

Одномерный массив – набор однотипных элементов, доступ к которым осуществляется с помощью выражения с *операцией индексирования*

Объявление ссылки на массив  `<type>[] array_ref_name;`

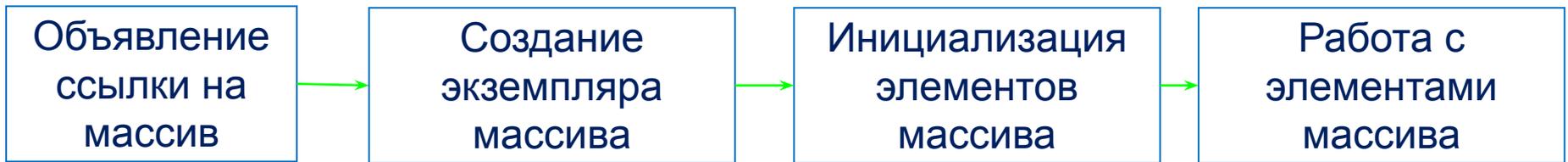
```
int[] intArray;  
double[] doubleArray;  
char[] charArray;
```

`new <type>[<size>];`  создание экземпляра массива конкретного

```
intArray = new int[100];  
doubleArray = new double[56];  
charArray = new char[26];
```

*Если ссылка не связана с данными, то ее значение - **null***

Одномерные массивы



Объявление ссылки на массив

тип[] имя_ссылки

Создание экземпляра объекта конкретного типа

new тип [размер_массива]

Операция индексирования

имя_ссылки_на_массив [индексирующее_выражение]

**Допустимо
объединение**

Описание одномерного массива

```
// Declare a single-dimensional array
```

```
int[] array1 = new int[5];
```

описание

```
// Declare and set array element values
```

```
int[] array2 = new int[] { 1, 3, 5, 7, 9 };
```

Описание с
инициализацией

```
// Alternative syntax
```

```
int[] array3 = { 1, 2, 3, 4, 5, 6 };
```

```
// Invalid syntax
```

```
int[] array2 = new int[5];
```

```
array2 = { 1, 3, 5, 7, 9 };
```

Источник:

[http://msdn.microsoft.com/ru-ru/library/9b9dty7d\(v=vs.90\).aspx](http://msdn.microsoft.com/ru-ru/library/9b9dty7d(v=vs.90).aspx)

Инициализация элементов массива

```
double[] ar = new double[10];  
ar[0] = ar[1] = ar[2] = ar[3] = ar[4] = 1.22;  
ar[5] = ar[6] = ar[7] = ar[8] = ar[9] = 1.22;
```

*Явная
инициализация*

```
double[] ar = new double[10];  
for (int i = 0; i < 10; i++) {  
    ar[i] = i * i + 1;  
    Console.Write(ar[i] + " ");  
}
```

*Вычисление значений
элементов по
соотношению*

```
double[] ar = new double[10];  
ar[0] = 0;  
for (int i = 1; i < 10; i++) {  
    ar[i] = ar[i - 1] + 3 * i;  
    Console.Write(ar[i] + " ");  
}
```

*Вычисление значений
элементов по
рекуррентной
формуле*

Генерация случайных чисел

Используем объект класса `System.Random`

```
Random rnd = new Random(5); // создаём объект-генератор
int X = rnd.Next(); // значение из [0;MaxInt)
Console.WriteLine(X);
```

```
Random rnd = new Random(); // создаём объект-генератор

int X = rnd.Next(); // значение из [0;MaxInt)
int Y = rnd.Next(100); // значение из [0;100)
int Z = rnd.Next(10, 20); // значение из [10;20)
```

Очень важно: объект-генератор достаточно создать один раз и использовать в программе.

Инициализация массива случайными числами

```
class Program {  
    // одно на весь класс статическое поле  
    static Random rnd = new Random();  
    static void Main(string[] args) {  
        double a = 0, b = 100;  
        double[] arr = new double[10];  
        for (int i = 0; i < 10; i++)  
            // масштабируем  
            arr[i] = a + (b - a) * rnd.NextDouble();  
        foreach (double el in arr)  
            Console.WriteLine($"{el:f3} ");  
    }  
}
```

Массив - ССЫЛОЧНЫЙ ТИП

```
int[] A = { 1, 2, 3, 4 };  
int[] B;  
B = A; // присваивание ссылки  
foreach (int a in A)  
    Console.Write(a + " ");  
B[1] = 13;  
Console.WriteLine();  
foreach (int a in A)  
    Console.Write(a + " ");
```

```
int[] ar = { 1, 2, 3, 4 };  
double[] ar2;  
ar2 = ar;
```



Что выведет на экран этот код?

```
int[] ar = new int[] { 1, 2, 3 };  
Console.Write(ar);
```

Что будет выведено?

```
int[] ar;  
if (ar == null)  
    Console.WriteLine("1");  
Console.WriteLine("2");
```

```
int[] ar = new int[10];  
if (ar == null)  
    Console.WriteLine("1");  
else  
    Console.WriteLine("2");
```

```
int[] ar = new int[10];  
int[] ar2 = ar;  
if (ar == ar2)  
    Console.WriteLine("1");  
else  
    Console.WriteLine("2");
```

Цикл foreach

```
foreach (тип идентификатор in ссылка)
{
    <тело_цикла>
}
```

```
int[] arInt = { 22, 5, 12, 63, -6, -52, 77, 41, 35, 23 };
```

```
foreach (int memb in arInt)
    Console.WriteLine(memb + " ");
```

По **memb** доступно только значение элемента

Сравните:

```
for (int i = 0; i < arInt.Length; i++)
    Console.WriteLine(arInt[i] + " ");
```

arInt[i] – доступ к значению
i – индекс элемента

Пример заполнения и обработки массива

```
int[] ar;
int n; // помним, что размер массива должен быть
положительным
do {
    Console.Write("Введите размер массива: ");
} while (!int.TryParse(Console.ReadLine(), out n) || n <= 0);
ar = new int[n];
for (int i = 0; i < ar.Length; i++) {
    Console.Write("ar[" + i + "] = ");
    ar[i] = int.Parse(Console.ReadLine());
}
int j = 1;
while (j < ar.Length) {
    Console.Write("ar[" + j + "] = " + ar[j]);
    j += 2;
}
```

Обработка массива

```
int[] ar = new int[] { -10, 0, 3, 2, 17, 99, -4};
int max = ar[0];
for (int i = 1; i < ar.Length; i++)
    if (ar[i] > max) max = ar[i];
    Console.Write(max);
```

*Поиск
максимального
элемента*

```
int[] ar = new int[] { -10, 0, 3, 2, 17, 99, -4};
int max = ar[0];
int maxInd = 0;
for (int i = 1; i < ar.Length; i++)
    if (ar[i] > max) {
        max = ar[i];
        maxInd = i;
    }
Console.Write("ar[{0}] = {1}", maxInd, max);
```

*Поиск
максимального и
сохранение его
индекса*

Обработка массива

```
int[] ar = new int[] { -10, 0, 3, 2, 17, 99, -4};
int i = 0;
double sumInverse = 0.0;
do {
    if (ar[i] == 0) continue;
    sumInverse += 1.0 / ar[i];
} while (++i < ar.Length);
Console.WriteLine("{0:f2}", sumInverse);
```

*Сумма величин
обратных
значениям
элементов
массива*

```
int[] ar = new int[] { -10, 0, 3, 2, 17, 99, -4};
int summ = 0;
for (int i = 0; i < ar.Length; i += 2)
    summ += ar[i];
Console.Write(summ);
```

*Сумма величин,
стоящих на
позициях с
чётными
индексами*

Преобразование массива

Сортировка

Сдвиги

Пример: Некольцевые сдвиги

Не кольцевой сдвиг элементов на две позиции влево:

I	0	1	2	3
A[I]	-8	1	-8	1

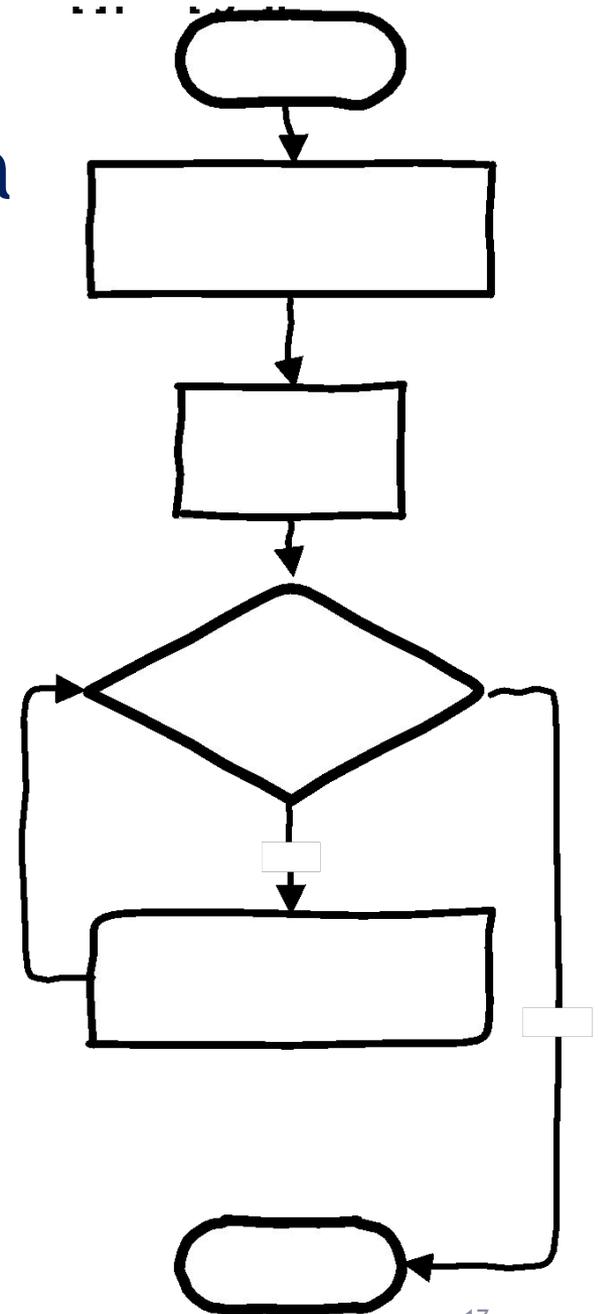
I	0	1	2	3
A[I]	12	3	-8	1

Не кольцевой сдвиг элементов на одну позицию вправо:

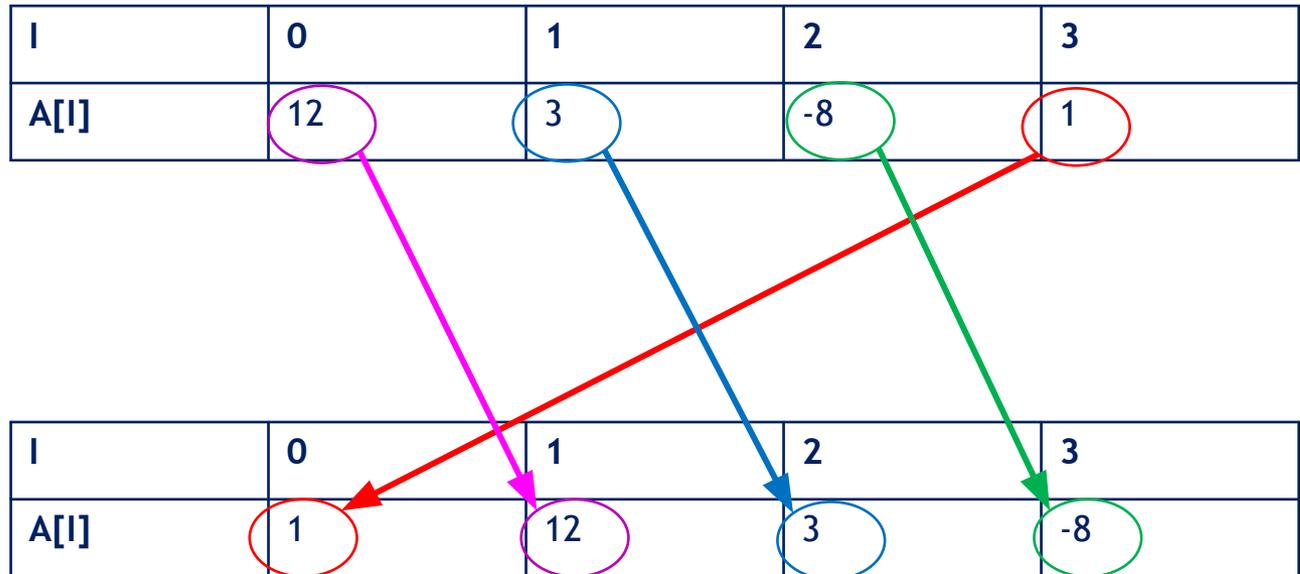
I	0	1	2	3
A[I]	12	12	3	-8

Некользящий сдвиг массива

```
1.  static void ArrPrintConsole (int[] arr) {
2.      foreach (int x in arr)
3.          Console.Write(x + " ");
4.  }
5.  static void Main(string[] args) {
6.      int[] arr = { 1, 2, 3, 4, 5 };
7.      ArrPrintConsole(arr);
8.      Console.WriteLine();
9.      for (int i = 0; i < arr.Length-1;i++) {
10.         arr[i] = arr[i + 1];
11.     }
12.     ArrPrintConsole(arr);
13. }
```

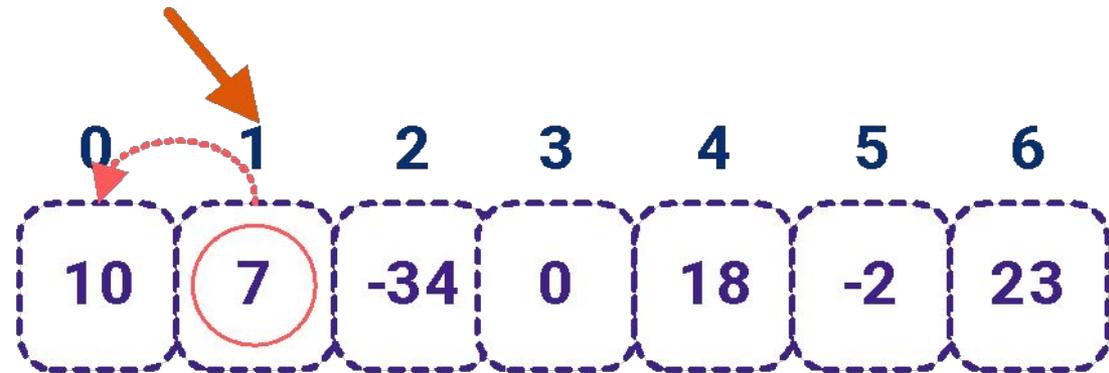


Пример: Кольцевые сдвиги

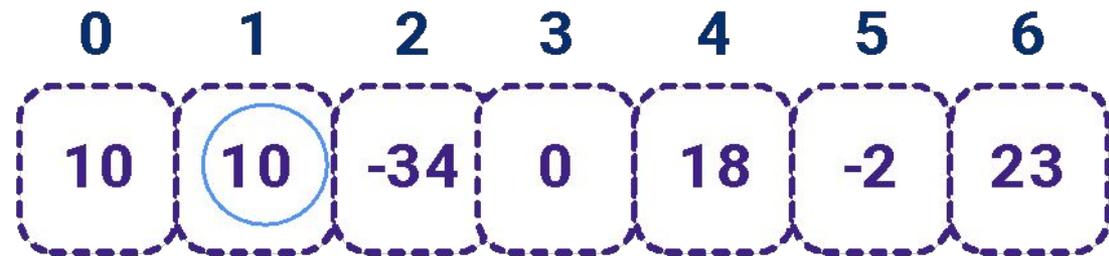


Самостоятельно на семинаре реализуйте кольцевой сдвиг на одну позицию вправо и на одну позицию влево

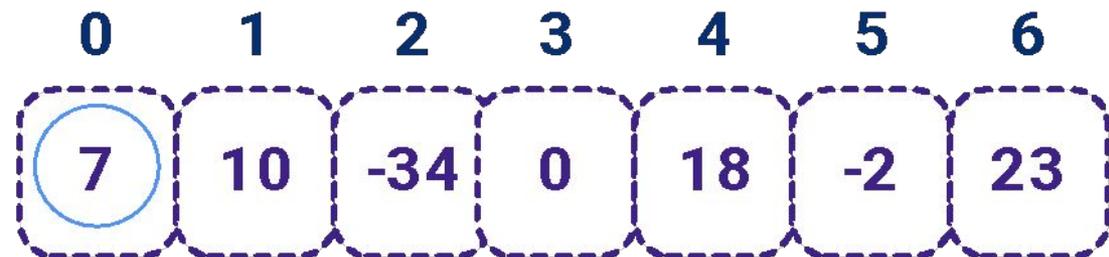
Сортировка вставками: шаг 1



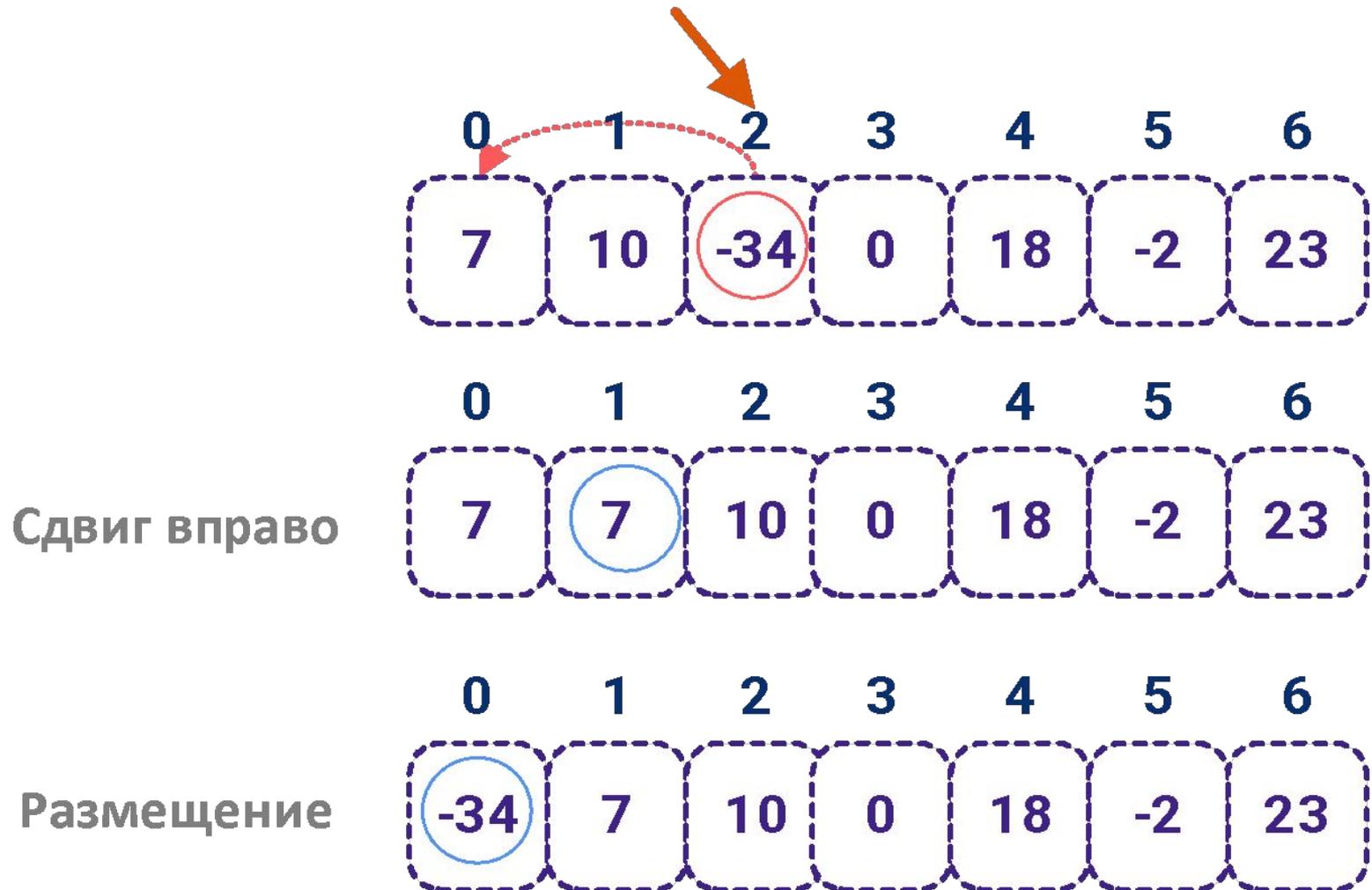
Сдвиг вправо



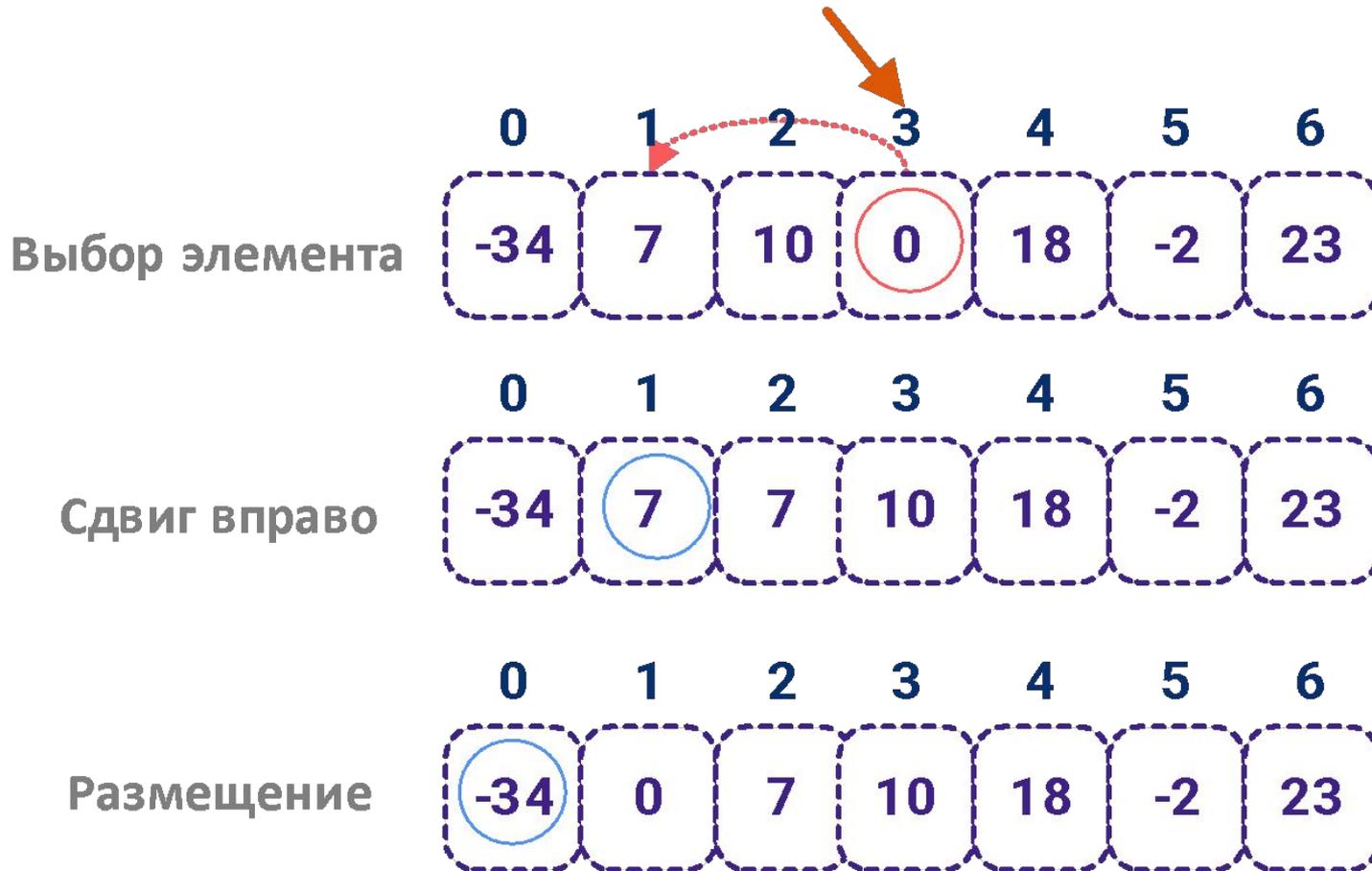
Размещение



Сортировка вставками: шаг 2



Сортировка вставками: шаг 3



*И так
далее...*

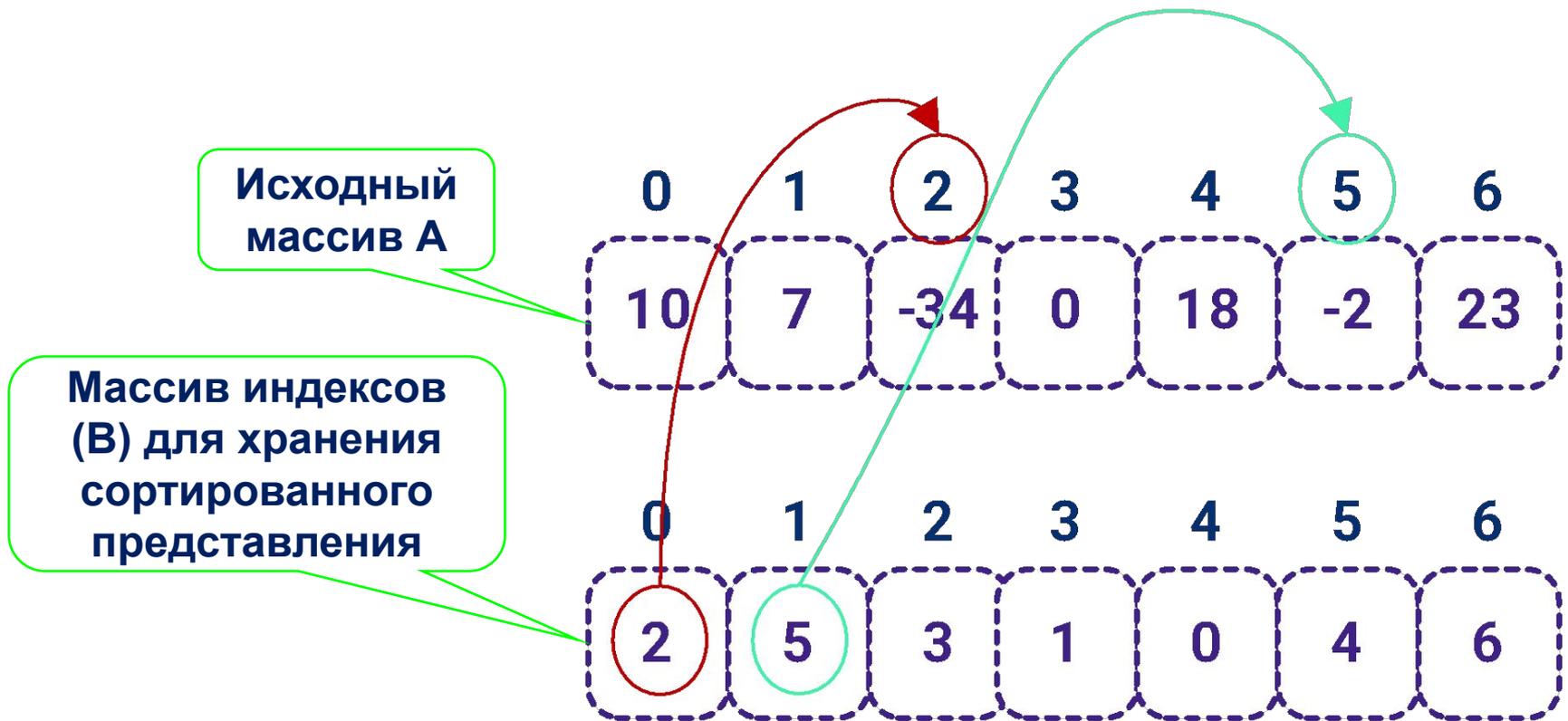
Бинарный поиск

```
do Console.Write("Введите целое число: ");
while (!int.TryParse(Console.ReadLine(), out numb));
int index = -1; // индекс найденного элемента массива
// Алгоритм двоичного поиска в упорядоченном массиве:
for (int i = 0, j = arInt.Length - 1, k = j / 2; i <= j; k = (i + j) / 2)
    if (arInt[k] == numb) {
        index = k;
        break;
    }
    else
        if (numb > arInt[k]) i = k + 1;
        else j = k - 1;
if (index == -1)
    Console.WriteLine("В массиве нет такого элемента!");
else
    Console.WriteLine("Результат поиска: arInt[{0}]={1}",
index, arInt[index]);
```

Оглавление и его использование

- **Сортировка** [*sorting*] – упорядочение набора некоторых объектов по некоторому критерию
 - Обычно критерий является формализацией отношения на *ключах* элементов
 - Сортировку можно выполнять путём физического перемещения исходных данных, а можно построить оглавление
- **Оглавление** (индексный массив) [*index array*] – массив *курсоров*, то есть индексов элементов некоторой последовательности
 - Отметим, что часто (например, в теории баз данных) *оглавление* называют «**индексом**», что приводит ещё к одной терминологической путанице
- Использование оглавления основано на косвенной адресации исходных данных:
 - Вместо $A[i]$ пишем $A[\mathit{ReMap}[i]]$, где $\mathit{ReMap}[i]$ – индекс элемента, который должен стоять на месте i в соответствии с оглавлением ReMap

Индексные массивы



Как получить значение элемента в массиве А по элементу из массива В?

Передача массивов в методы

- При передаче массива в метод, передаётся ссылка на массив (элементы массива не копируются)
- Сама ссылка на массив передаётся по значению

```
public static void Change(string[] changingArr) {
    for (int i = 0; i < changingArr.Length; i++) {
        changingArr[i] = i.ToString();
    }
}
```

Изменяются
значения элементов

```
public static void Main() {
    string[] arr = { "Foo", "Bar", "Baz", "Quux" };
    Console.WriteLine("Before calling Change:");
    foreach (string str in arr) Console.Write(str + " ");
    Change(arr);
    Console.WriteLine("\nAfter calling Change:");
    foreach (string str in arr) Console.Write(str + " ");
}
```

```
public static void Change(string[] changingArr) {
    string[] newAr = { "One", "Two", "Three" };
    changingArr = newAr;
}
```

Изменяется
значение ссылки

```
public static void Main() {
    string[] arr = { "Foo", "Bar", "Baz", "Quux" };
    Console.WriteLine("Before calling Change:");
    foreach (string str in arr) Console.Write(str + " ");
    Change(arr);
    Console.WriteLine("\nAfter calling Change:");
    foreach (string str in arr) Console.Write(str + " ");
}
```

```

// метод обработки массива          Вариант 1
public static void ArrayProceed(int[] ar) {
    int[] locAr = new int[ar.Length]; //
выделяем память
    // копируем исходный массив
    Array.Copy(ar, locAr, ar.Length - 1);
    for (int i = 0; i < locAr.Length; i++)
        locAr[i] += locAr.Length;
    ar = locAr; // переприсваиваем
ССЫЛКИ
}
// метод печати массива
public static void ArrayPrint(int[] ar) {
    Console.WriteLine();
    foreach (int el in ar)
        Console.Write(el + " ");
    Console.WriteLine();
}

```

```

public static void Main() {
    int[] mainAr = { 1, 2, 3, 4 };
    ArrayPrint(mainAr);
    Console.WriteLine();
    ArrayProceed(mainAr);
    ArrayPrint(mainAr);
}

```

```

// метод обработки массива          Вариант 2
public static void ArrayProceed(int[] ar) {
    int[] locAr = ar; // присваиваем ссылку
    for (int i = 0; i < locAr.Length; i++)
        locAr[i] += locAr.Length;
    ar = locAr; // переприсваиваем
ССЫЛКИ
}
// метод печати массива
public static void ArrayPrint(int[] ar) {
    Console.WriteLine();
    foreach (int el in ar)
        Console.Write(el + " ");
    Console.WriteLine();
}

```

Вывод различен.
В обоих вариантах по значению передаётся ссылка

```
// метод обработки массива          Вариант 1
public static void ArrayProceed(int[] ar) {
    int[] locAr = new int[ar.Length];
    // копируем исходный массив
    Array.Copy(ar, locAr, ar.Length - 1);
    for (int i = 0; i < locAr.Length; i++)
        locAr[i] += locAr.Length;
    ar = locAr; // переприсваиваем
```

ССЫЛКИ

```
}
// метод печати массива
public static void ArrayPrint(int[] ar) {
    Console.WriteLine();
    foreach (int el in ar)
        Console.Write(el + " ");
    Console.WriteLine();
}
```

```
public static void Main() {
    int[] mainAr = { 1, 2, 3, 4 };
    ArrayPrint(mainAr);
    Console.WriteLine();
    ArrayProceed(mainAr);
    ArrayPrint(mainAr);
}
```

Вывод одинаков.

*В варианте 1 ссылка передаётся по значению
В варианте 2 ссылка передаётся по ссылке*

```
// метод обработки массива          Вариант 2
public static void ArrayProceed(ref int[] ar) {
    int[] locAr = ar; // присваиваем ссылку
    for (int i = 0; i < locAr.Length; i++)
        locAr[i] += locAr.Length;
    ar = locAr; // переприсваиваем ссылки
}
// метод печати массива
public static void ArrayPrint(int[] ar) {
    Console.WriteLine();
    foreach (int el in ar)
        Console.Write(el + " ");
    Console.WriteLine();
}
```

Важные мелочи

- Индексация элементов массива начинается с нуля
- Если нужно вернуть из метода ссылку на массив, а он пуст - верните **NULL**
- Контролируйте выход индекса за границы массива

ССЫЛКИ

- **Статья Random Class**

[https://msdn.microsoft.com/en-us/library/system.random\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.random(v=vs.110).aspx)