L#7

Основы алгоритмизации и программирования.

Срезы и списки

Педагогическое образование, 3 семестр

Mayer Svetlana Fyodorovna

+ и * операторы для массивов

a + b – конкатенация двух массивов

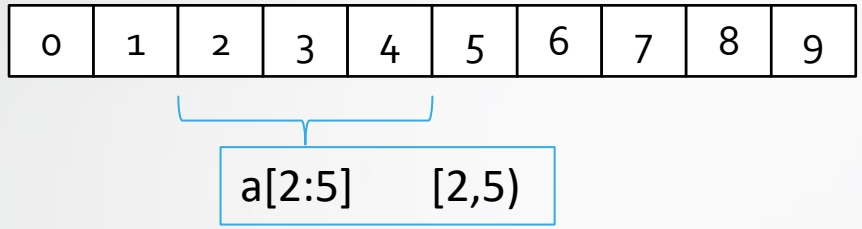
a * N – конкатенация **N** копий массива **a** в результирующий массив

```
var a := Arr(1,2,3);
var b := Arr(4,5,6);

a + b // 1 2 3 4 5 6
a * 3 // 1 2 3 1 2 3 1 2 3
Arr(1) * 10 // 1 1 1 1 1 1 1 1 1 1
Arr(1)*5 + Arr(2)*5 // 1 1 1 1 1 2 2 2 2 2
(Arr(1) + Arr(2))*5 // 1 2 1 2 1 2 1 2 1 2
```

Срезы массива

```
var a := Arr(0,1,2,3,4,5,6,7,8,9);
```



Срез массива - это подмножество исходного массива

Два варианта: **a[x:y]** or **a[x:y:step]**. **x** и **y** могут быть опущены.

```
a[:4] – 0 1 2 3
a[4:] – 4 5 6 7 8 9
a[:a.Length-1] – 0 1 2 3 4 5 6 7 8
a[:]       – 0 1 2 3 4 5 6 7 8 9  (copy of a)
a[::2]     – 0 2 4 6 8
a[1::2]    – 1 3 5 7 9
a[4:1:-1] – 4 3 2
a[::-1]    – 9 8 7 6 5 4 3 2 1 0  (reverse of a)
```

Пример:

- **Задача:** Дан массив A размера N и целое число K (1 <= K <= N). Выведите его элементы с порядковыми номерами (т. е. индексами), кратными K :
- A_k, $A_{2 * k}$, $A_{3 * k}$ …
- Не использовать оператор if.

```
begin
var n:=ReadInteger('сколько элементов?'); // 4
var a:=ReadArrReal(n); // 2  6  7  5
var k:=ReadInteger('K='); // 2
a[k-1 : : k].Print;  // 6  5
end.
```

Пример

- **Задача:** Дан массив A размером N. Сначала выведите его элементы с четными порядковыми номерами (в порядке возрастания порядковых номеров), а затем — элементы с нечетными порядковыми номерами (также в порядке возрастания порядковых номеров):

$a_2, a_4, a_6, \ldots a_1, a_3, a_5 \ldots$

- Не использовать условные оператор.

```
begin
var n:=ReadInteger('сколько элементов?');
var a:=arrRandomInteger(n);
a.Println;
var slice:=a[1::2]+a[::2];
slice.Print
end.
```

```
сколько элементов? 10
8 96 24 61 80 60 24 40 95 15
96 61 60 40 15 8 24 80 24 95
```

Пример

- Задача: Дан массив размера N и целые числа K и L (1 <= K<= L<= N). Найдите среднее арифметическое (среднее значение) элементов массива с индексами от K до L включительно.

```
begin
var n:=ReadInteger;
var a:=arrrandominteger(n);
a.Println;
var k:=ReadInteger('K = ');
var l:=ReadInteger('L = ');
var slice:=a[k-1:l].Average;
slice.Print;
end.
```

```
>> 10
59 87 0 37 57 69 79 19 100 5
K =  >> 2
L =  >> 4
41.3333333333333
```

Пример

- Задача: Дан массив размера N. Найдите минимальный элемент из его четных элементов: a_2, a_4, a_6, ...

```
begin
var n:=ReadInteger;
var a:=arrRandomInteger(n);
a.Println;
println('slice: ',a[1::2]);
print(a[1::2].min);
end.
```

```
>> 10
96 79 71 87 61 21 51 74 67 89
slice: [79,87,21,74,89]
21
```

Reverse массива

```
begin
var a:=new integer[10];
a:=arrRandomInteger(10);
print(a);// [41,81,84,63,12,26,88,25,36,72]
Reverse(a);
print(a) //[72,36,25,88,26,12,63,84,81,41]
end.
```

Reverse(a) =

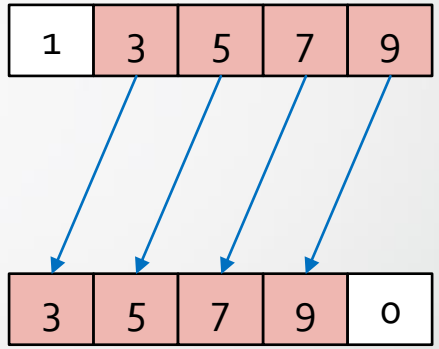
Срезы: a := a[::-1]

Сдвиг влево

Задача: Создайте пользовательскую процедуру для сдвига элементов влево

```
procedure ShiftLeft(a: array of integer);
begin
  for var i := 0 to a.Length - 2 do
    a[i] := a[i + 1];
  a[a.Length - 1] :=0;
end;

begin
  var a := new integer[5];
  a := arrRandomInteger(5); // [56,28,33,57,25]
  shiftLeft(a);
  print(a)// [28,33,57,25,0]
end.
```



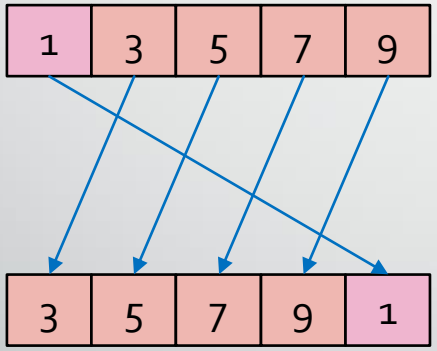
Срезы:

a := a[1:] + Arr(0);

[3, 5, 7, 9] + [0]

Цикличный сдвиг влево

```
procedure CircularShiftLeft(a: array of integer);
begin
  var v := a[0];
  for var i:=0 to a.Length-2 do
    a[i] := a[i+1];
  a[a.Length-1] := v;
end;
```



Срезы:

a := a[1:] + a[:1];

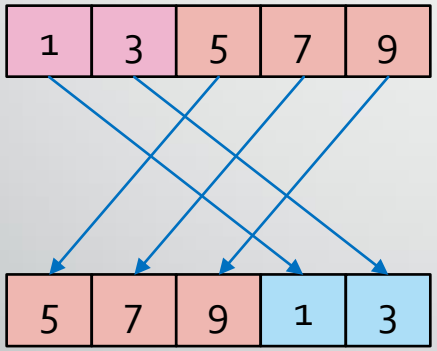
[3, 5, 7, 9] + [1]

Цикличный сдвиг на k позиций

```
1. loop k do
     CircularShiftLeft(a); // неэффективно

2. With second array

3. With partial reverse
```



k = 2

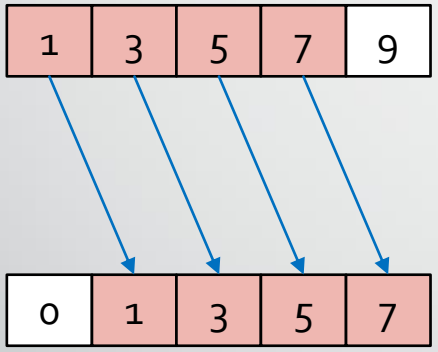
Срезы:

a := a[k:] + a[:k];

[5, 7, 9] + [1, 3]

Сдвиг вправо

```
procedure ShiftRight<T>(a: array of T);
begin
  for var i:=a.Length-1 downto 1 do
    a[i] := a[i-1];
  a[0] := default(T);
end;
```



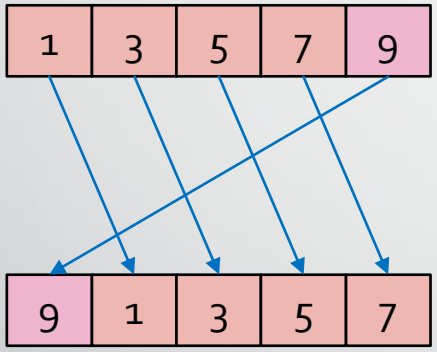
Срезы:

a := Arr(0) + a[:a.Length-1];

[0] + [1, 3, 5, 7]

Цикличный сдвиг вправо

```
procedure CircularShiftRight<T>(a: array of T);
begin
  var v := a[a.Length-1];
  for var i:=a.Length-1 downto 1 do
    a[i] := a[i-1];
  a[0] := v;
end;
```



Срезы:

var m := a.Length-1;

a := a[m:] + a[:m];

[9] + [1, 3, 5, 7]

Вставка и удаление в массиве

срезы

Задача 1. Дан массив из N целых чисел. Необходимо вставить элемент x в k-й индекс, k<=N.

```
begin
  var a := arr(5, 12, 1, 3, 11, 19);
  var x := ReadInteger ('enter a number to insert');
  var k := ReadInteger ('enter an order number');
  a := a[:k] + Arr(x) + a[k:];
  print(a) // [5,12,3,1,3,11,19]
end.
```

Задача 2. Дан массив из N целых чисел. Необходимо удалить элемент с индексом k, k<N.

```
begin
  var a := arr(5, 12, 1, 3, 11, 19);
  var k := ReadInteger ('enter an order number');
  a := a[:k] + a[k+1:];
  print(a) //[5,12,3,11,19]
end.
```

Объединение двух отсортированных массивов

Объединение двух отсортированных массивов

Задача. Даны два отсортированных массива a и b.

Необходимо объединить их в третий отсортированный массив

Решение (неэффективное):

```
var c := a + b;
Sort(c);
```

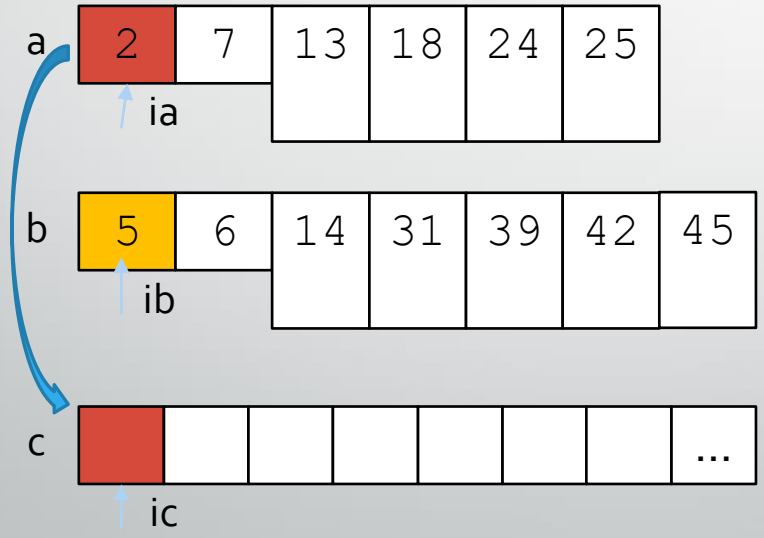
```
begin
  var a:=arr(1,3,11,19);
  var b:=arr(1,13,21);
  var c:=a+b; // [1,3,11,19,1,13,21]
  Sort(c); //[1,1,3,11,13,19,21]
end.
```

Объединение двух отсортированных массивов

Задача. Даны два отсортированных массива a и b.

Необходимо объединить их в третий отсортированный массив

Решение (частями). Будем использовать счетчик **ia** как индекс первого элемента массива **a** и счетчик **ib** – как индекс первого элемента **b**. Если **a[ia]<b[ib]**, тогда копируем **a[ia]** в **c** и увеличиваем счетчик **ia**. Иначе копируем **b[ib]** и увеличиваем **ib**.

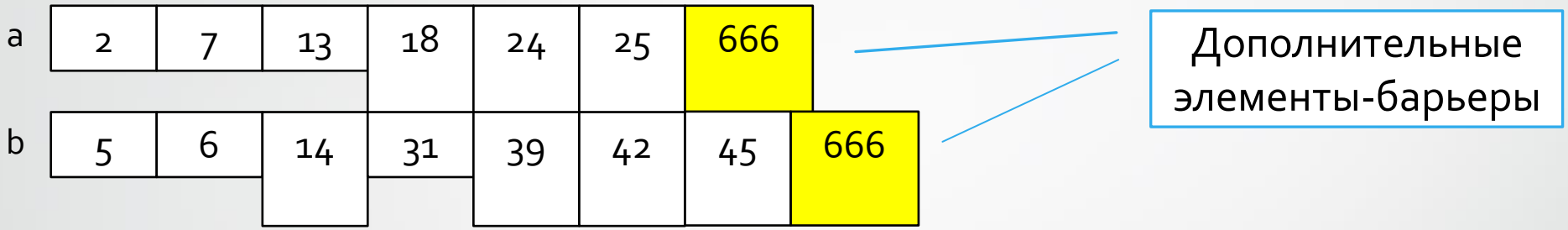


Это не работает! Почему?

```
var (ia,ib) := (0,0);
for var ic:=0 to c.Length-1 do
  if a[ia]<b[ib] then
  begin
    c[ic] := a[ia];
    ia += 1
  end
  else
  begin
    c[ic] := b[ib];
    ib += 1
  end;
```

Объединение двух отсортированных массивов

Решение (полное). Добавим элемент-барьер в конец каждого массива^



Предыдущий код будет работать!

Объединение двух отсортированных массивов

```
function Merge(a, b: array of integer; n, m: integer): array of real;
begin
  Assert((0 < n) and (n < a.Length));
  Assert((0 < m) and (m < b.Length));
  a[n] := integer.MaxValue; // барьер
  b[m] := integer.MaxValue; // барьер
  SetLength(Result, m + n);
  var (ia, ib) := (0, 0);
  for var ic := 0 to n + m - 1 do
    if a[ia] < b[ib] then
    begin
      Result[ic] := a[ia];
      ia += 1;
    end
    else
    begin
      Result[ic] := b[ib];
      ib += 1;
    end;
end;
```

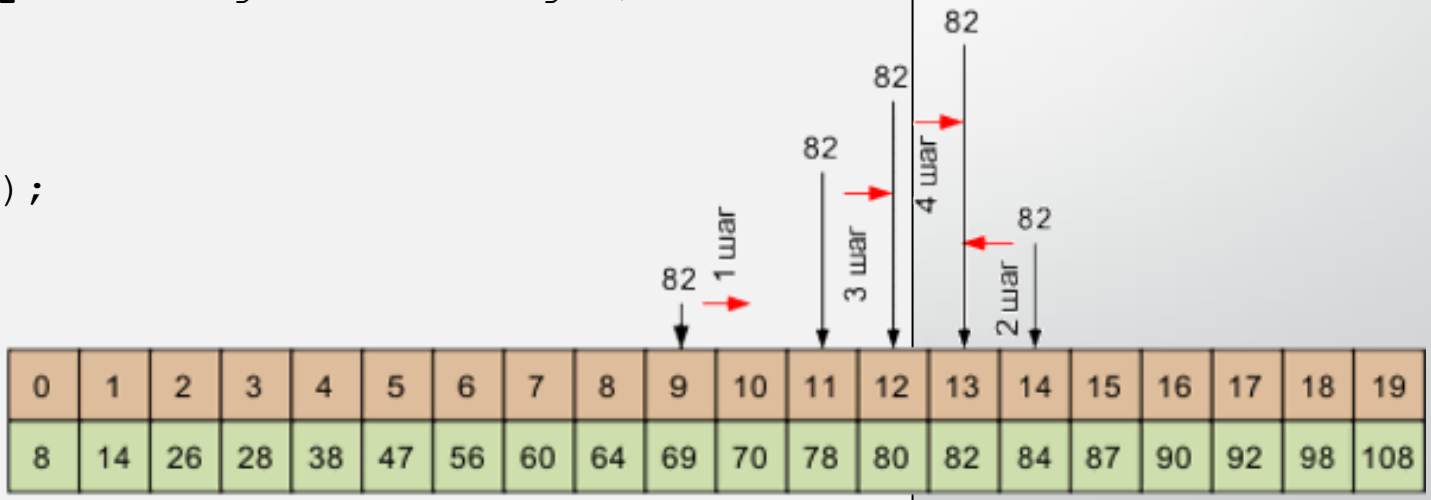
```
begin
  var a := arr(1, 3, 11, 19);
  var b := arr(1, 13, 21);
  setLength(a, 5); // для дополнительного элемента
  setLength(b, 4); // для дополнительного элемента
  print(Merge(a, b, 4, 3)) // [1,1,3,11,13,19,21]
end.
```

Бинарный поиск в отсортированном массиве

Стандартный метод **a.BinarySearch(x)**

Код метода (не рассматривать).

```
function BinarySearch(a: array of integer; x: integer):
integer;
begin
  var k: integer;
  var (l,r) := (0, a.Length-1);
  repeat
    k := (l+r) div 2;
    if x>a[k] then
      l := k+1
    else r := k-1;
  until (a[k]=x) or (l>r);
  Result := a[k]=x ? k : -1;
end;
```



Списки

List<T>

(Список) List представляет собой динамический массив с динамическим изменением его размера во время выполнения программы.

Объявление списка List :

real, array of integer, ...

```
var l := new List<integer>; // l.Count = 0
```

Короткое объявление с инициализацией:

```
var L := Lst(25, -23, 47, 100, 0, 14);
```

Or:

```
var L2 := Lst(Arr(14, 172, -5, 0, 39)); // [14,172,-5,0,39]
var L3 := Lst(ArrRandom(5, -99, 99)); // [0,-6,2,-81,10]
var L4 := Lst(L3); // в L4 один элемент – это список list
[[0,-6,2,-81,10]]
```

List<T>

Добавление элемента в конец списка:

```
var L := Lst(Arr(14, 172, -5, 0, 39));
L.Add(5); // расширение списка
L.Add(3);
L.Add(4);
L += 8; // синоним l.Add(8)
println(L);  // [14,172,-5,0,39,5,3,4,8]
```

Проход по списку:

```
for var i:=0 to L.Count-1 do
  Print(L[i]);
foreach var x in L do
  Print(x);
```

Методы списка (List)

Операции со списками:

```
var L := Lst(5,2,3);
Print(2 in L); // True
Print(2 * L); // [5,2,3,5,2,3]
L.Print; // 5 2 3
Print(L + Lst(7,6,8)); // 5 2 3 7 6 8
```

Методы списков:

```
L.Insert(ind,x); // вставка элемента x с позицией индекса ind
L.RemoveAt(ind); // удаление элемента по индексу ind
L.RemoveRange(ind,count) // удаление диапазона элементов
L.RemoveAll(x -> x.IsOdd); // удаление с условием
L.IndexOf(3); // индекс первого найденного указанного знач-я или -1
L.FindIndex(x -> x > 4);//индекс первого найденного с условием или -1
L.Clear;
L.Reverse;
L.Sort;
```

Примеры

Задача. Дан массив из **N** целых чисел. Вставьте все четные элементы массива в список L1, а все нечетные элементы – в список L2

Решение.

```
begin
  var a := arrrandominteger(10, 5, 25);
  println(a); // [17,25,8,17,21,9,19,22,19,24]
  var L1 := new List<integer>;
  var L2 := new List<integer>;

foreach var x in a do
    if x.IsEven then
      L1 += x
    else L2 += x;
  L1.println; // 8 22 24
  L2.println; // 17 25 17 21 9 19 19
end.
```

Вставка и удаление в массиве

с использованием срезов и списков

Задача 1. Дан массив из N целых чисел. Необходимо вставить элемент x на k-ую позицию, k<=N.

```
begin
  var a := arr(5, 12, 1, 3, 11, 19);
  var x := ReadInteger ('enter a number to insert');
  var k := ReadInteger ('enter an order number');
  a := a[:k] + Arr(x) + a[k:];
  print(a) // [5,12,3,1,3,11,19]
end.
```

Со списком:

```
L.Insert(k,x);
```

Задача 2. Дан массив из N целых чисел. Необходимо удалить элемент с индексом k, k<N.

```
begin
  var a := arr(5, 12, 1, 3, 11, 19);
  var k := ReadInteger ('enter an order number');
  a := a[:k] + a[k+1:];
  print(a) //[5,12,3,11,19]
end.
```

Со списком:

```
L.RemoveAt(k);
```

Q & A