



CODE@FEIT

ЧАС 4:

НИЗИ

ПРАКТИЧНИ ПРИМЕРИ ЗА ЦИКЛУСИ СО НИЗИ

ПРИМЕРИ НА УПОТРЕБА НА НИЗИ



ДЕФИНИЦИЈА НА ПОЛИЊА (НИЗИ)

- Индексирана листа на вредности
- Групирање на елементи од ист тип
- Може да се креираат низи од било кој тип
- Најчесто се некакви броеви
- Должината на полето се одредува при дефинирање
- Вообичаено полињата во јава се алоцираат динамички
- Се користи индекс за пристапување до елементите



ПРИМЕР ЗА ДЕФИНИРАЊЕ НА ПОЛЕ

- `type []ime = new type [N];`
- `int []pole = new int[10];`





ПРИСТАП ДО ЕЛЕМЕНТИТЕ

- Индексот на секое поле започнува од нула
- Има вкупно N елементи
- Последниот елемент е на индекс N-1
- Пример:
 - `int []pole = new int[3];`
 - `pole[1] = 10`
 - `pole[2] = 20;`
 - **`pole[3] = 30; //` грешка**



ПРИМЕР

- Пример: Првите 10 троцифрени броеви да се сместат во едно поле. Потоа, полето да се отпечати на екран
- Решение:

```
public class Primer {  
    public static void main(String []arrays){  
        int []pole = new int[10];  
        pole[0] = 100;  
        pole[1] = 101;  
  
        ...  
    }  
}
```



ДРУГИ НАЧИНИ НА ДЕКЛАРИРАЊЕ НА ПОЛИЊА

- Одделно алоцирање
 - `int [] pole;`
 - `pole = new int[10];`
- Имплицитно креирање на поле
 - `int [] pole = {10, 20, 30};`
- Пример: да се креира поле во кое ќе се сместат имињата на месеците според нивниот индекс



ИЗМИНУВАЊЕ НА ПОЛИЊА

- Полињата најчесто се изминуваат со некаков циклус
- По автоматизам се користи for циклус, но може и други
- Пример:
 - Да се соберат сите броеви на една низа



ИЗМИНУВАЊА НА ПОЛИЊА

- Креирање на поле

- `int [] pole = new int[4];`
Вметнување вредности во полето

```
pole[0]=0;  
pole[1]=30;  
pole[2]=-2;  
pole[3]=7;
```

- Изминување на полето и сумирање на елементите

```
int suma=0;  
for(i=0;i<4;i++)  
    suma+=pole[i];
```




МАНИПУЛАЦИЈА СО ИНДЕКСИ

- Да се напише програма која ќе ги отфрли сите негативни броеви од крајот на низата. Да се реши



МАНИПУЛАЦИЈА СО ИНДЕКСИ (2)

- Вредности на полето:

```
int [] pole = new int[7];  
pole[0]=0;  
pole[1]=30;  
pole[2]=-2;  
pole[3]=7;  
pole[4]=-3;  
pole[5]=-2;  
pole[6]=-1;
```

- Одредување на бројот на негативни елементи од крајот на листата и
намалување на должината

Печатење:

```
int n=7;  
for(i=6;i>=0;i--)  
{  
    if(pole[i]>=0)break;  
    n--;  
}
```

```
for(i=0;i<n;i++){  
    System.out.println(pole[i]);}
```



ГОТОВИ ЕЛЕМЕНТИ И ФУНКЦИИ

- Свойството `length` може да се користи за да се дознае бројот на елементи на полето
- Функција за копирање на едно поле во друго поле
 - `public static void arraycopy(Object src, int srcPos, Object dest, int destPos, int length)`



ВЕЖБА 1

- Да се напише програма која за низа A составена од 10 целобројни елементи, ќе ја пресмета средната вредност на елементите на низата.



ВЕЖБА 2

- Да се напише функција која ќе го врати најголемиот број од некоја низа од цели броеви. Потоа да се напише главна програма во која ќе се иницијализира низата и ќе се повика функцијата. На екран да се отпечати најголемиот број.



ВЕЖБА 2 РЕШЕНИЕ

- Сите останати елементи да се зголемат за вредноста на најголемиот елемент

```
int max=pole[0];
    for(i=0;i<7;i++){
        if(pole[i]>max) max=pole[i];
    }
```

```
System.out.println("Najgolem element e " + max);
```

```
for(i=0;i<7;i++) pole[i]+=max;

    for(i=0;i<n;i++){
        System.out.println(pole[i]);}
```

```
goran@goran-VirtualBox:~/javaa$ javac hello.java
goran@goran-VirtualBox:~/javaa$ java hello
Najgolem element e 30
30
60
28
37
27
28
29
```



ВЕЖБА 3

- Да се напише програма која за низа A со големина N (N не поголемо од 100) ќе ги отфрли сите елементи кои се помали од средната вредност на низата. Да се отпечатат оригиналната низа и ново добиената низа.



ВЕЖБА 3 – МОДИФИЦИРАНА

- Да се отфрлат од постоечката низа

```
for(i=0;i<7;i++){
    suma+=pole[i];
}

srv=suma/7;
System.out.println("Sredna vrednost e " + srv);

for(i=0;i<n;i++){
    if(pole[i]<srv){
        for(j=i;j<n-1;j++){
            pole[j]=pole[j+1];
        }
        n--;
    }
}
```




ВЕЖБА 4

- Да се напише функција која ќе ја пресмета најголемата разлика помеѓу два соседни броеви



ВЕЖБА 5

- Да се напише програма што најпрво ќе го најде најголемиот елемент. Потоа, програмата треба да ја зголеми вредноста на сите останати елементи за таа максимална вредност.



ВЕЖБА 6

- Да се креира фибоначи-S низа. На почеток од тастатура се внесува почетниот услов S и должината на низата.



ВЕЖБА 7

- Да се напише функција која добива како аргумент низа од цели броеви. Функцијата треба да ги отстрани дупликатите на низата.



ПРЕБАРУВАЊЕ И СОРТИРАЊЕ

- Начини на пребарување
 - Линеарно (секвенцијално)
 - Бинарно
- Сортирање (подредување)
 - Bubble sort
 - Simple sort



БИНАРНО ПРЕБАРУВАЊЕ

0	1	2	3	4	5	6	7	8
5	12	17	23	38	44	77	84	90
↑ п				↑ с				↑ к



БИНАРНО ПРЕБАРУВАЊЕ ВО JAVA

```
public static int bin_java(int kluc, int[] niza) {  
    int poc = 0; int kraj = niza.length - 1;  
    while (poc <= kraj) {  
        int sred = poc + (kraj - poc) / 2;  
        if (kluc < niza[sred]) kraj = sred - 1;  
        else if (kluc > niza[sred]) poc = sred + 1;  
        else return sred; }  
    return -1; }
```

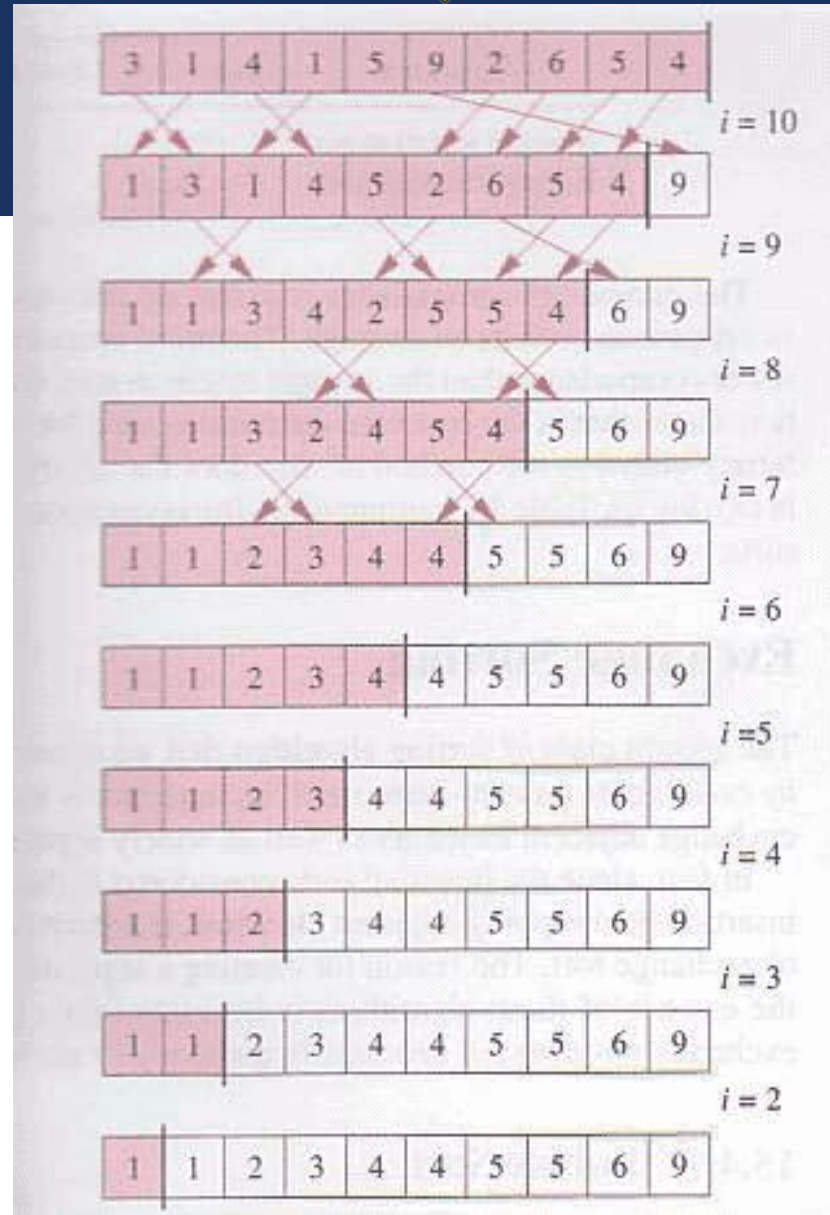


BUBBLESORT

```
public int[] bubbleSort(int[] podatoci){  
    int dolzina = podatoci.length;  
    int pom = 0;  
    for(int i = 0; i < dolzina; i++){  
        for(int j = 0; j < dolzina - i - 1; j++){  
            if(podatoci[j] > podatoci[j+1]){  
                pom = podatoci[j];  
                podatoci[j] = podatoci[j+1];  
                podatoci[j+1] = pom;  
            }  
        }  
    }  
    return podatoci;  
}
```




SIMPLE SORT





ПРАКТИЧЕН ПРИМЕР ЗА ПОДРЕДУВАЊЕ

- Дадени се две низи со податоци за тежина и висина на неколку луѓе. Да се провери дали луѓето можат да формираат кула од луѓе, т.е. секој човек да биде понизок и полесен од оној под него.
-
- **Забелешка:** Двете низи треба да се сортираат во **растечки** редослед, најпрво по **тежина**, а потоа по **висина**. По сортирањето, двете низи **истовремено** се изминуваат и се проверува дали соседните членови го задоволуваат условот (секој човек да биде и полесен и понизок од оној после него).



ПРАКТИЧЕН ПРИМЕР ЗА ПОДРЕДУВАЊЕ (2)

- За внесени вредности во облик **(тежина, висина)** = (55,155), (50,150), (80,200), (69,170) сортираните вредности и по двата параметри се: (50,150), (55,155), (69,170), (80,200) и го задоволуваат условот, **но**, ако се внесат вредности **(тежина, висина)** = (55,155), (50,160), (80,200), (69,170), сортираните вредности се: (55,155), (50,160), (69,170), (80,200) и **не** го задоволуваат условот.
- **За дома:** ако сортираната низа не го задоволува условот, од неа да се исфрлат сите членови кои го нарушуваат истиот, а остатокот од низата да го задоволува условот