

Објектно оријентисано програмирање



Владимир Филиповић
vladaf@matf.bg.ac.rs

Александар Картељ
kartelj@matf.bg.ac.rs

Управљачке структуре у програмском језику Јава



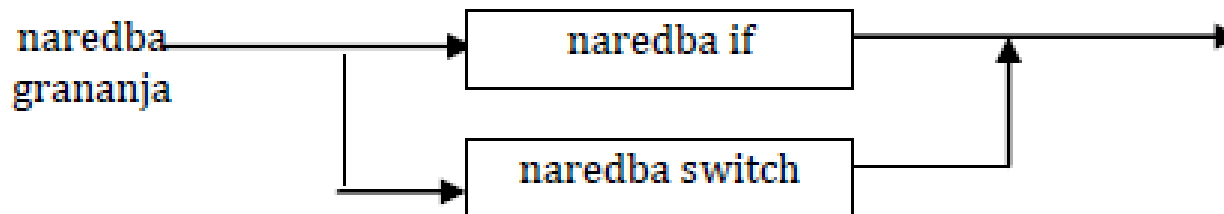
Владимир Филиповић
vladaf@matf.bg.ac.rs

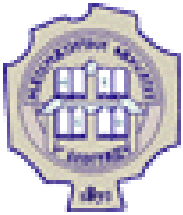
Александар Картељ
kartelj@matf.bg.ac.rs



Наредбе гранања

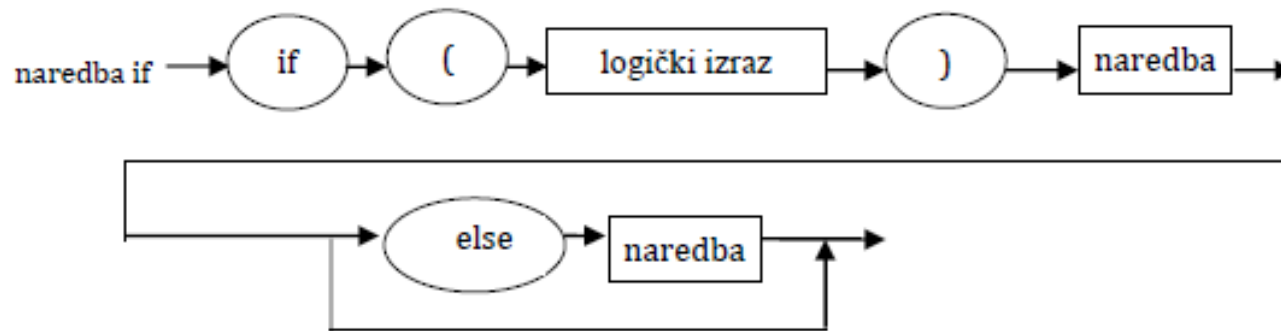
- Наредбе гранања омогућавају да се одабере извршавање једног дела програма у зависности од испуњења одређених услова.
- Користе се још и називи условне наредбе, односно наредбе одабирања.
- Постоје две наредбе гранања у језику Јава:
 1. наредба `if`
 2. и наредба `switch`





Наредба if

- За условно извршење наредбе или за избор између извршења две наредбе обично се користи наредба if.
- Синтакса наредбе if у Јава језику је следећа:



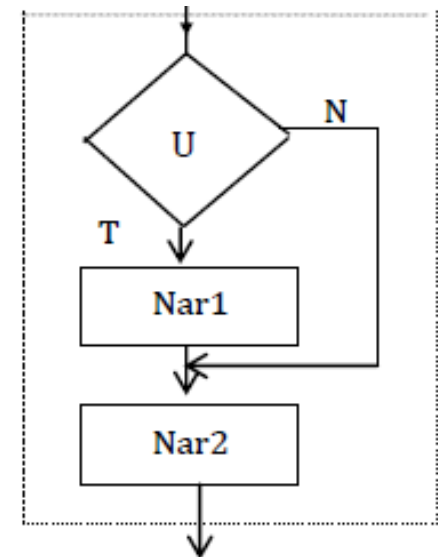
- Из наведеног описа закључујемо да се наредба if појављује у два облика:
 1. if (<izraz>) <naredba> - непотпуни облик
 2. if (<izraz>) <naredba> else <naredba> - потпуни облик.



Наредба if (2)

- Наредба if у непотпуном облику извршава се на следећи начин:
 1. израчунава се вредност израза;
 2. ако је вредност израза истинита, извршава се наредба која следи иза израза;
 3. ако је вредност израза неистинита, извршава се прва наредба која следи иза наредбе if.

```
if (U)
    Nar1;
Nar2;
```

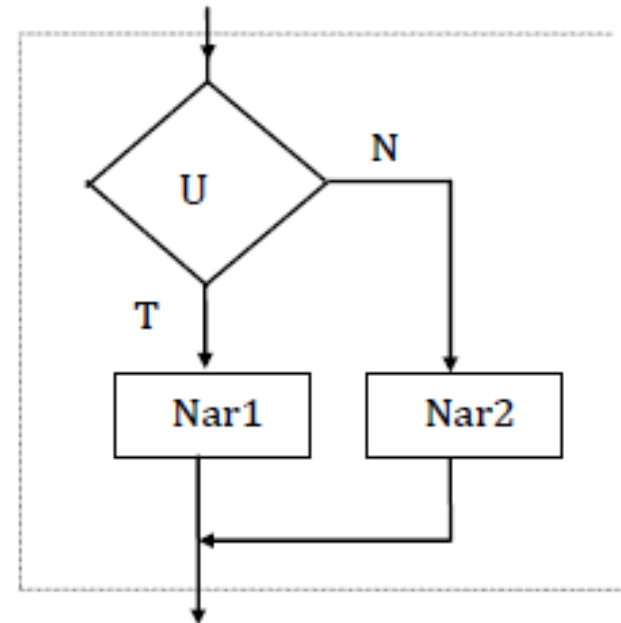




Наредба if (3)

- Наредба if у потпуном облику се извршава на следећи начин:
 1. израчунава се вредност израза;
 2. ако је вредност израза истинита, извршава се наредба која следи иза израза;
 3. у супротном, извршава се наредба иза резервисане речи else.

```
if (U)
    Nar1;
else
    Nar2;
```





Наредба if (4)

- У случају када се појављује наредба if унутар наредбе if, са једном придруженом наредбом else, поставља се питање да ли је наредба else придружена првој наредби if или другој наредби if.

- Другачије формулисано, ако имамо конструкцију:

```
if (U1) if (U2) Nar1 else Nar2
```

- да ли ће се Nar2 извршити ако је:

- (а) логички израз U1 тачан, а логички израз U2 нетачан, или
- (б) ако је логички израз U1 нетачан?



Наредба if (5)

Пример.

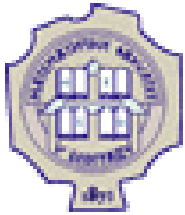
```
if (a < b ) System.out.println("a je manje od b");  
if (tezina>200) {  
    tezina = 200;  
    ind = true;  
}else  
    ind =false;
```

Уместо наредбе if у неким ситуацијама може се употребити условни оператор
?:

```
(uslov) ? then-izraz : else-izraz
```

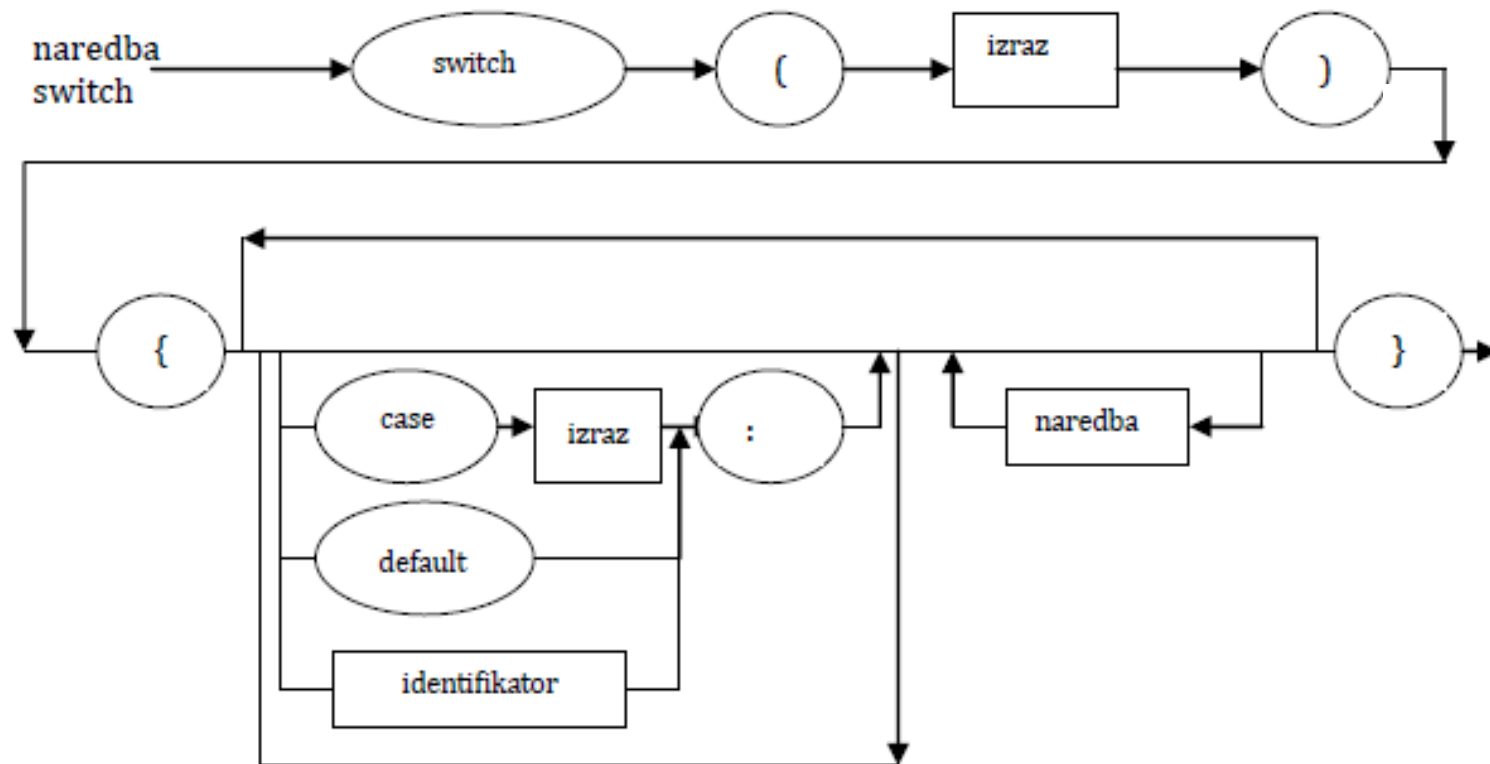
Пример.

```
min = (x<y) ? x : y;
```

Наредба switch

- Наредба `switch` служи за избор једне наредбе из скупа од неколико могућих, а на основу вредности неког израза.





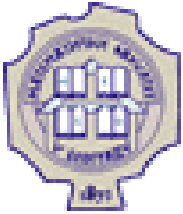
Наредба switch (2)

- Резервисана реч `default` може се појавити само једнапут, што није посебно назначено.
- Израз иза резервисане речи `switch` назива се селектор и мора бити типа:
 1. `byte`, `char`, `short` или `int`;
 2. Енумерисани тип (од верзије 5);
 3. `String`, као и типа неке од класа-омотача простих типова тј.: `Character`, `Byte`, `Short` или `Integer` (од верзије 7).
- Вредности (ознаке) које се појављују иза наредбе `case` морају бити истог типа као и селектор и међусобно различите.



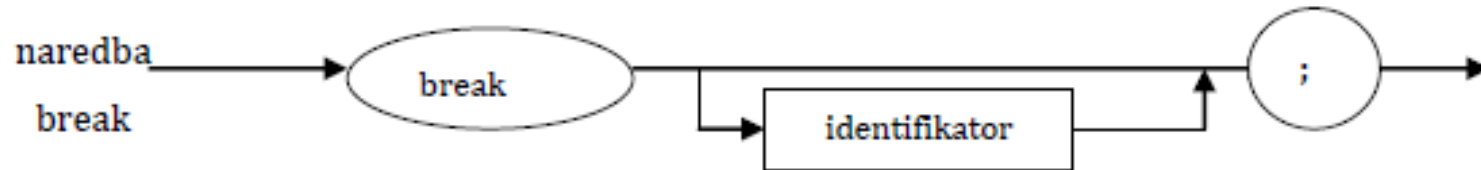
Наредба switch (3)

- Наредба switch се извршава на следећи начин:
 1. Израчунава се вредност селектора и ако није типа `int`, вредност се преводи у тип `int`.
 2. Упореди се израчуната вредност са вредностима (ознакама) иза наредби `case`.
 3. Ако се деси поклапање извршава се поклапајућа `case` наредба.
 4. Ако се не поклапа ни са једном `case` наредбом онда се извршава `default` наредба.
 5. Ако нема наредбе која има ознаку `default`, наредба после наредбе `switch` биће следећа која се извршава.



Наредба switch (4)

- У уској вези са наредбом switch је наредба break.
- Следећим дијаграмом описана је синтакса наредбе break:

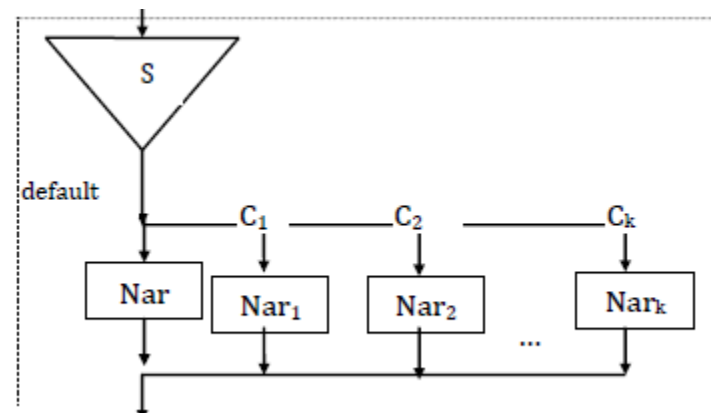


- Ако наредба break не садржи идентификатор, управљање се преноси на прву следећу наредбу иза наредбе у којој се налази.

```

switch (S)
{
    case C1:
        Nar1;
        break;
    case C2:
        Nar2;
        break;
    ...
    case Ck:
        NarK;
        break;
    default:
        Nar;
        break;
}

```





Наредба switch (5)

Пример. Уместо:

```
if (oper == '+')
    sabrati(arg1, arg2);
else if (oper == '-')
    oduzeti(arg1, arg2);
else if (oper == '*')
    pomnoziti(arg1, arg2);
.....
```

прегледније је користити:

```
switch(test) {
    case vrednost1:
        Rezultat1;
        break;
    case vrednost2:
        Rezultat2;
        break;
    .....
    default:
        PodrazumevaniRezultat; // opciono
}
```



Наредба switch (6)

Пример.

```
switch (doba) {
    case 'p' :
        System.out.println("Setnja");
        break;
    case 'l' :
        System.out.println("Kupanje");
        break;
    case 'j' :
        System.out.printl("Branje grozdja");
        break;
    case 'z' :
        System.out.println("Skijanje");
        break;
    default:
        System.out.println("Greska");
}
```



Наредба switch (7)

Пример.

```
switch (doba) {  
    case "prolece":  
        System.out.println("Setnja u prirodi");  
        break;  
    case "leto":  
        System.out.println("Kupanje");  
        break;  
    case "jesen":  
        System.out.println("Branje voca");  
        break;  
    case "zima":  
        System.out.println("Skijanje");  
        break;  
}
```



Наредба switch (7)

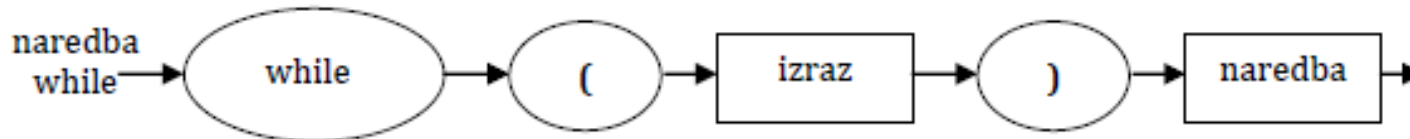
Пример.

```
switch (broj) {  
    case 2:  
    case 4:  
        f = broj*broj+3;  
        break;  
    default:  
        f = broj;  
}
```




Наредба while

- Наредба `while` се још назива наредба циклуса са предусловом.
- У њој се најпре проверава да ли је испуњен услов и ако јесте извршавају се наредбе циклуса.



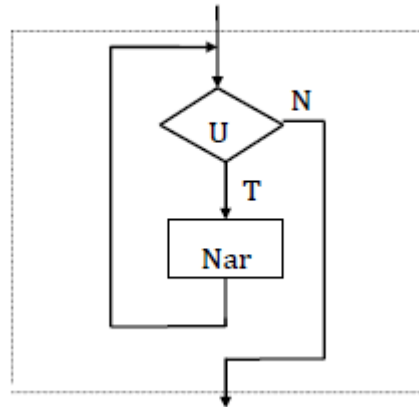
- Израз у наредби `while` је логичког типа. Ефекат наредбе `while` је следећи:
 1. израчунава се вредност израза;
 2. ако је вредност израза истинита, извршава се наредба и понављају кораци 1. и 2. ;
 3. ако је вредност израза неистинита, завршава се извршавање наредбе `while` и прелази се на прву следећу наредбу иза `while`.



Наредба while (2)

Наредба while се извршава на следећи начин:

```
while (U) Nar
```



Пример.

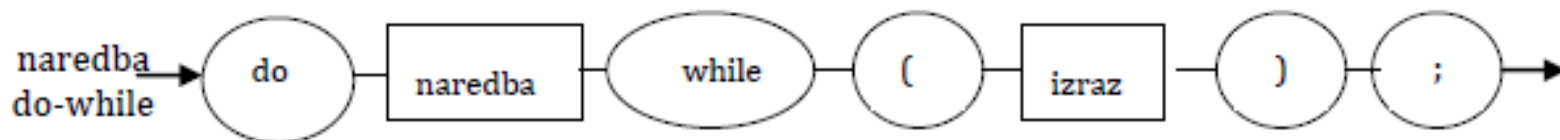
```
{  
    ...  
    double x=5.0;  
    while (x>0.1) {  
        x=Math.random();  
        System.out.println("x="+x);  
    }  
}
```



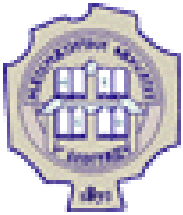
Наредба do-while

- Наредба do-while се још назива наредба циклуса са постусловом.

Синтакса наредбе do-while је следећа:



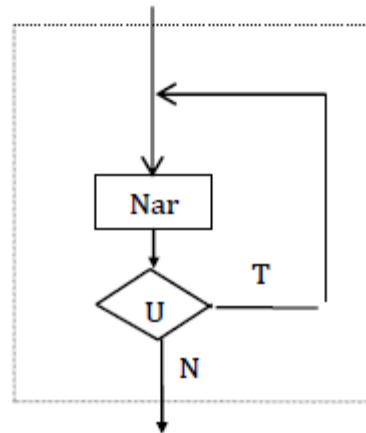
- Израз је логичког типа.
- Извршавање наредбе do-while реализује се преко следећих корака:
 1. извршава се наредба иза резервисане речи do;
 2. израчунава се вредност израза у заградама;
 3. ако је вредност израза истинита, процес се наставља; у супротном, прелази се на прву наредбу иза наредбе do-while.



Наредба do while (2)

Наредба do while се извршава на следећи начин:

```
do Nar while (U)
```



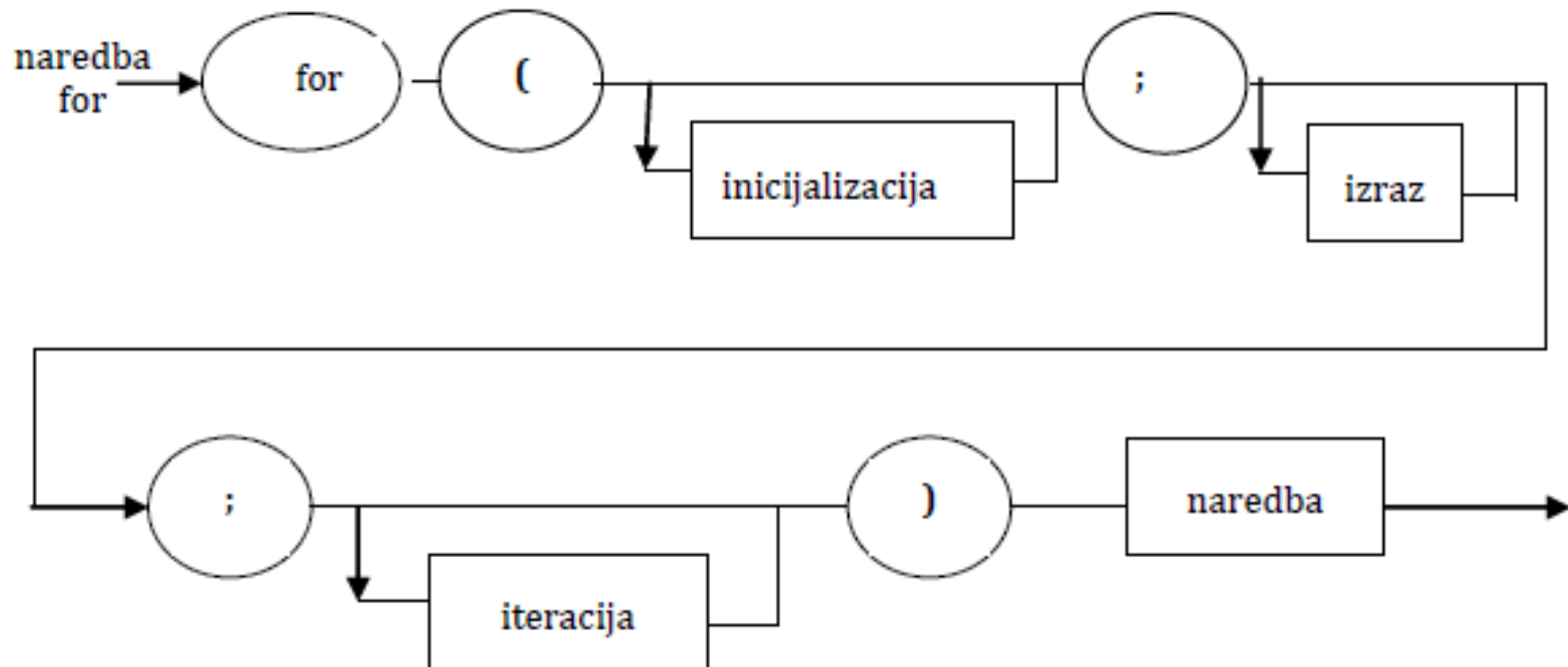
Пример.

```
{ ...  
    do  
        x=Math.random();  
        System.out.println("x="+x);  
    while (x<0.9);  
    ...  
}
```



Наредба for

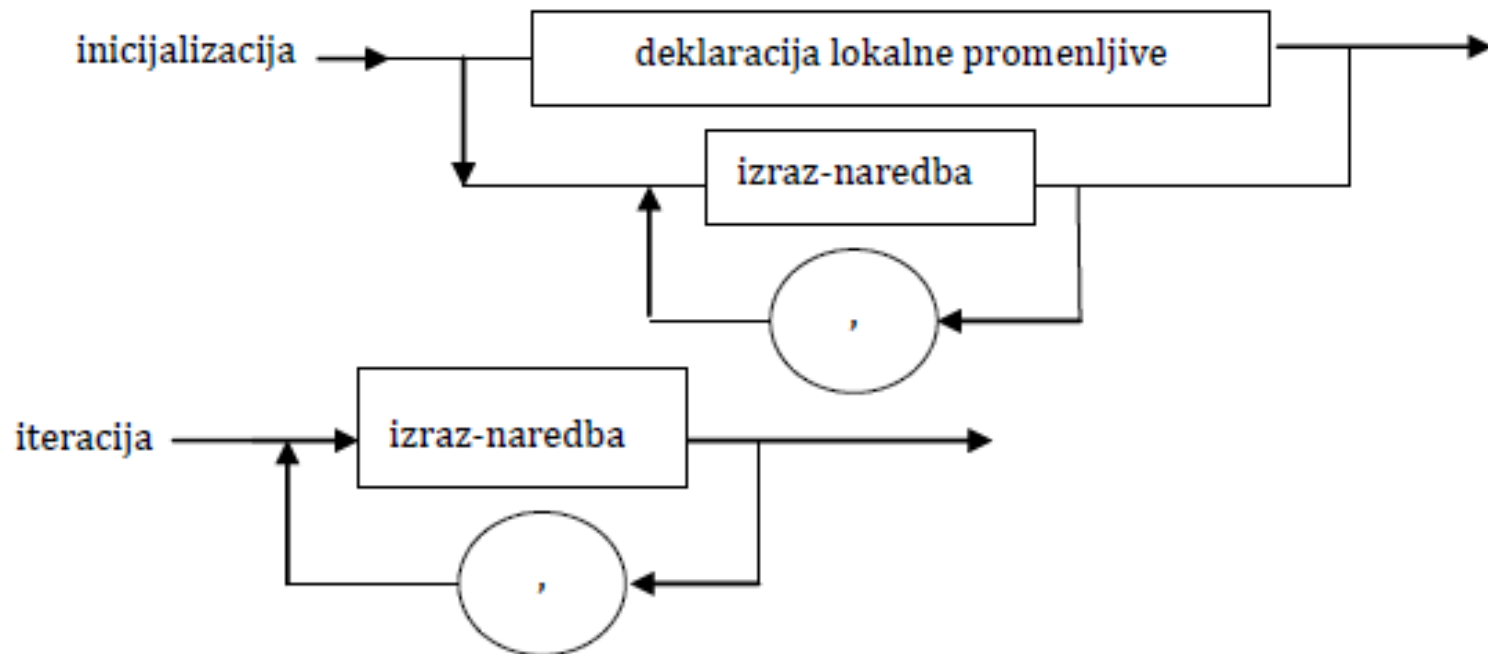
- Наредба for је моћна наредба за опис циклуса.
- Најчешће се користи за опис циклуса код којих је број понављања наредби унапред познат.
- Синтакса наредбе for је следећа:





Наредба for (2)

- Свака од секција наредбе for може бити изостављена, али знак ; мора постојати.
- Синтакса секције за иницијализацију и итерацију код наредбе for је следећа:

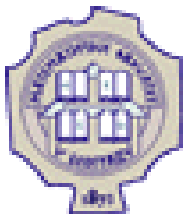




Наредба for (3)

Наредба for се извршава на следећи начин:

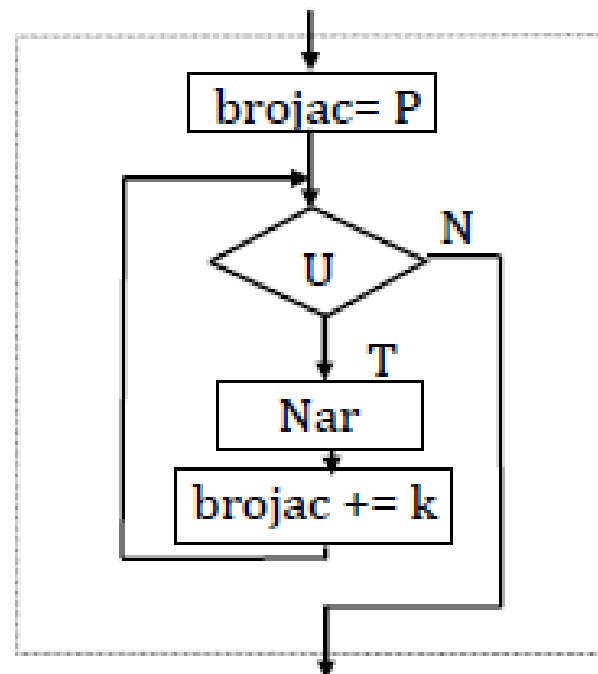
1. Прво се извршава иницијализациони део петље for:
 - Подешава се вредност управљачке променљиве петље for.
 - Иницијализација се извршава само једанпут.
2. Након иницијализације:
 - Израчунава се вредност израза (ако постоји) који мора бити логичког типа.
 - Ако израз не постоји, подразумевана вредност је true.
3. У зависности од вредности израза:
 - Ако има вредност **true**, извршава се тело петље, а затим се извршава итерација петље и поново се иде на корак 2.
 - Уколико израз има вредност **false**, окончава се извршавање петље.



Наредба for (4)

- Извршавање наредбе for показује и следећи дијаграм:

```
for (brojac = P; U; brojac += K)  
    Nar
```



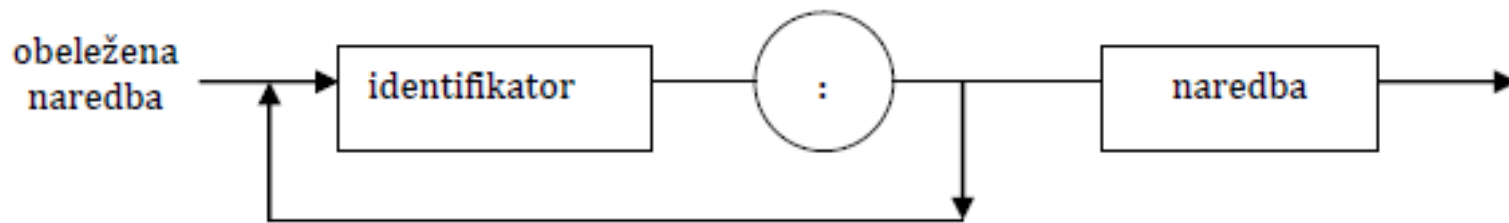
Пример.

```
{  
    ...  
    for (int i=1; i<=10; i++)  
        System.out.println("Ana voli programiranje");  
    ...  
}
```




Обележена наредба

- Наредба у програмском језику Јава може имати једно или више обележја (лабела) и онда се назива обележеном наредбом.
- Синтакса обележене наредбе је следећа:

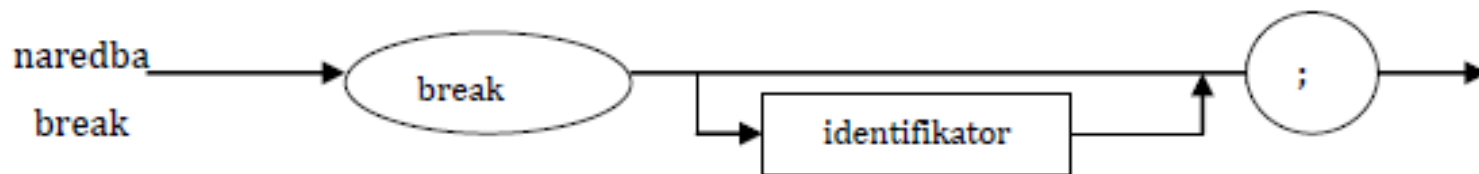


- Обележена наредба користи се у комбинацији са наредбама `break` и `continue` када се жели безусловни пренос управљања на одређено место у програму.



Наредба `break`

- Наредба `break` је у уској вези не само са наредбом `switch`, већ и са наредбама циклуса.
- Синтакса наредбе `break` описана је следећим дијаграмом:



- Поред тога што се наредба `break` може користити:
 1. за излазак из наредбе `switch`,
 2. она се може искористити за излазак из петље,
 3. али и за безусловни пренос управљања на обележену наредбу (под одређеним условима).Ово представља једну врста “контролисане” `goto` наредбе.



Наредба `break` (2)

Пример.

- Да би се прекинуло извршавање наредби унутрашње петље и вратило се у спољашњу, користи се наредба `break` без обележја

```
for(int i=0; i<100; i++) {  
    for(int j=0; j<100; j++) {  
        if (uslov)  
            break;  
    }  
}
```



Наредба `break` (3)

Пример.

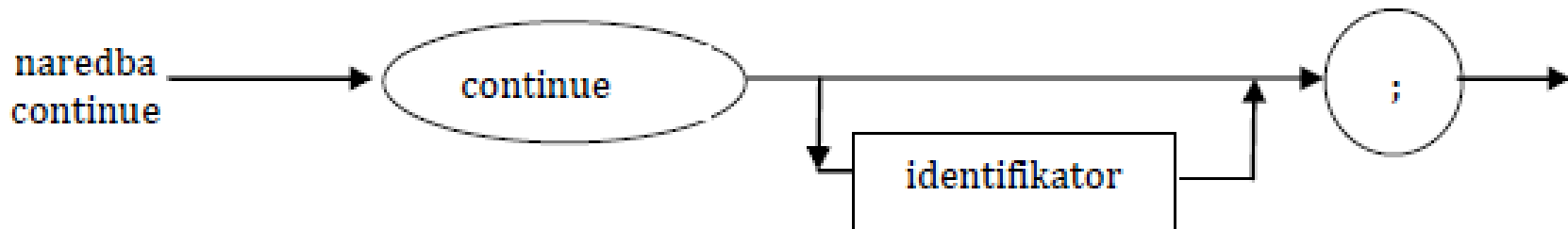
- Да би се прекинуло извршавање наредби обе петље, користи се наредба `break` са обележјем.

```
...
izlaz:
for(int i=0; i<100; i++) {
    for(int j=0; j<100; j++) {
        if (uslov)
            break izlaz;
    }
}
```



Наредба `continue`

- Наредба `continue` је по синтакси слична наредби `break`.
- Она се може користити само у наредбама за опис циклуса.
- Наредба `continue` је корисна када треба да се настави извршавање циклуса, уз прескакање одређеног скупа наредби.
- Помоћу ове наредбе управљање се преноси на израз за проверу услова изласка из петље.
- Наредба `continue` има следећу синтаксу:





Наредба `continue` (2)

Пример.

Да би се прескочио препис оних елемената низа који су једнаки нули, користи се наредба `continue` без обележја.

```
...
br1 = 0;
br2 = 0;
while (br1 < niz.length) {
    if (niz[br1] == 0){
        br1++;
        continue;
    }
    vektor[br2++] = niz[br1++];
}
```



Наредба `continue` (3)

Пример.

- Учитавање низа од четири бинарне цифре и спајање учитаних цифара у један стринг, те приказ стринга.
- Наставља се са следећом итерацијом петље обележене обележјем ако цифра није бинарна.

```
...
String s="";
System.out.println("Unesite cetiri binarne cifre ");
vrh:
do {
    d=sc.nextInt();
    if((d!=0)&&(d!=1))
        continue vrh;
    s +=d;
}
while(s.length()<4);
System.out.println("Formiran je binarni string: "+s);
```



Наредба continue (4)

Пример.

- Наставља се са следећом итерацијом спољне (обележене) ПЕТЉЕ.

```
String text = "Trazimo neki podstring u ovom stringu";
String podstring = "pod";
boolean pronadjen = false;
int max = text.length() - podstring.length();

test:
for (int i = 0; i <= max; i++) {
    int n = podstring.length();
    int j = i;
    int k = 0;
    while (n-- != 0) {
        if (text.charAt(j++) != podstring.charAt(k++)) {
            continue test;
        }
    }
    pronadjen = true;
    break test;
}
System.out.println(pronadjen ? "Pronadjen" : "Nije pronadjen");
```




Захвалница

Велики део материјала који је укључен у ову презентацију је преузет из презентације коју је раније (у време када је он држао курс Објектно оријентисано програмирање) направио проф. др Душан Тошић.

Хвала проф. Тошићу што се сагласио са укључивањем тог материјала у садашњу презентацију, као и на помоћи коју ми је пружио током конципирања и реализације курса.