

STRUKTURA PETLJE



STRUKTURA PETLJE

- Struktura petlje koristi se kada neki blok naredaba treba višekratno ponavljati.
- Ponavljati se može:
 - **unaprijed zadani broj puta** (konačne petlje),
 - **sve dok je ispunjen zadani uvjet** (uvjetne petlje).

Sanda, 2015.

2

FOR PETLJA

- **for petlja** - petlja koja se izvršava unaprijed zadani, konačan broj puta.
- Osnovni oblik **for petlje**:

```
for (početno stanje; uvjet; prirast)
{
    blok naredbi
}
naredba iza bloka
```

Sanda, 2015.

3

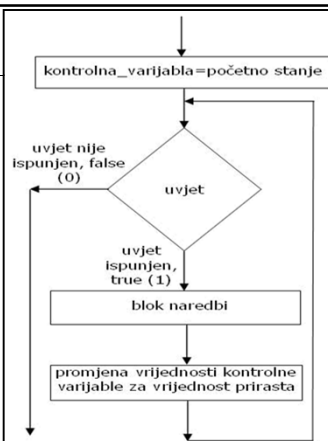
FOR PETLJA

- Svaka **for petlja** ima svoju **kontrolnu varijablu**.
- Kontrolnoj varijabli se na početku petlje pridružuje **početna vrijednost**.
- Vrijednost kontrolne varijable svakim se prolaskom kroz petlju mijenja za **vrijednost prirasta**.
- Petlja se izvršava sve **dok je uvjet ispunjen** (logička istina;1).

Sanda, 2015.

4

FOR PETLJA



Sanda, 2015.

5

PRIMJER 49



- Korištenjem **for petlje** ispisati brojeve od 1 do 20.
- Ispis neka bude oblika:

Ispis brojeva od 1 do 20:

```
1 2 3 4 5 ... 19 20
1 2 3 4 5 6 7 8 9 .....20
```

Sanda, 2015.

6

```
#include<iostream>
using namespace std;
int main()
{
    int brojac;
    cout<<"Ispis brojeva od 1 do 20:"<<endl;
    for (brojac=1;brojac<=20;brojac++)
    {
        cout<<brojac<<" ";
    }
    return 0;
}
```

Sanda, 2015.

7

```
#include<iostream>
using namespace std;
int main()
{
    int brojac;
    cout<<"Ispis brojeva od 1 do 20:"<<endl;
    for (brojac=1;brojac<=20;brojac++)
    {
        cout<<brojac<<" ";
    }
    cout<<endl;
    return 0;
}
```

8

PRIMJER 49

- Provjera programa:



Sanda, 2015.

9

PRIMJER 50

- Potrebno je dopuniti prethodni primjer tako da se ispisuju brojevi iz raspona od M do N (raspon bira korisnik).

Upisi pocetni broj raspona:
Upisi zavrzni broj raspona:

Ispis brojeva od ... do ... :

... ..

Sanda, 2015.

10

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,m,n;
    cout<<"Upisi pocetni broj raspona:";
    cin>>m;
    cout<<"Upisi zavrzni broj raspona:";
    cin>>n;
    cout<<endl<<"Ispis brojeva od "<<m<<" do "<<n<<:"<<endl;
    for (brojac=m;brojac<=n;brojac++)
    {
        cout<<brojac<<" ";
    }
    return 0;
}
```

Sanda, 2015.

11

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,m,n;
    cout<<"Upisi pocetni broj raspona:"; cin>>m;
    cout<<"Upisi zavrzni broj raspona:"; cin>>n;
    cout<<endl<<"Ispis brojeva od "<<m<<" do "<<n<<:"<<endl;
    for (brojac=m;brojac<=n;brojac++)
    {
        cout<<brojac<<" ";
    }
    return 0;
}
```

12

PRIMJER 50

- Provjera programa:

```
C:\Users\Sanda\Desktop>iss.exe
Upisi pocetni broj raspona:33
Upisi zavrzni broj raspona:65

Ispis brojeva od 33 do 65:
33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52
53 54 55 56 57 58 59 60 61 62 63 64 65
```

Sanda, 2015.

13

PRIMJER 51



- U prethodnom primjeru obrisati dio definicije petlje, ovako:

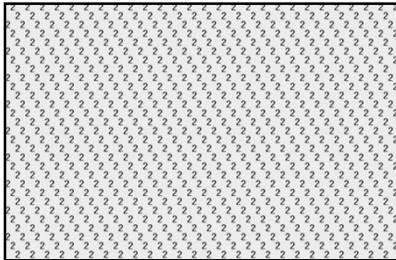
```
for (brojac=m;brojac<=n;)
```

Sanda, 2015.

14

OPREZ - BESKONAČNA PETLJA

- Petlja se izvodi neograničen broj puta jer je uvjet uvijek ispunjen.



Sanda, 2015.

15

KAKO IZBJEĆI BESKONAČNU PETLJU?

- Zadati uvjet koji jamči konačan broj ponavljanja petlje.
- Navesti sva tri argumenta u definiciji **for** petlje.
- Izbjegavati promjenu vrijednosti kontrolne varijable unutar bloka naredbi **for** petlje.

Sanda, 2015.

16

PRIMJER 52



- Treba ispisati parne brojeve iz raspona od 50 do N. N bira korisnik.

Upisi završnu vrijednost raspona:

Parni brojevi iz raspona od 50 do ... su:
50 52 54 56

Sanda, 2015.

17

PRIMJER 52

- Napomene:

- Pošto je razlika dva susjedna parna broja 2, problem je moguće riješiti tako da se vrijednost kontrolne varijable petlje svakim prolaskom kroz petlju mijenja za 2 (prirast 2).

```
brojac=brojac+2 ili brojac+=2
```

(Zadatak se može riješiti i provjerom djeljivosti sa 2!)

Sanda, 2015.

18

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,N;
    cout<<"Upisi završnu vrijednost raspona:";
    cin>>N;
    cout<<endl<<"Parni brojevi iz raspona od 50 do "
    <<N<<" su:" <<endl;
    for (brojac=50;brojac<=N;brojac+=2)
    {
        cout<<brojac<<" ";
    }
    return 0;
}
```

Sanda, 2015.

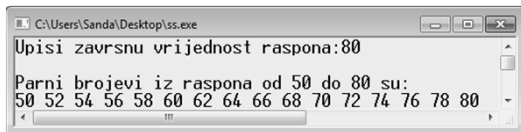
19

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,N;
    cout<<"Upisi završnu vrijednost raspona:";
    cin>>N;
    cout<<endl<<"Parni brojevi iz raspona od 50 do "
    <<N<<" su:" <<endl;
    for (brojac=50;brojac<=N;brojac+=2)
    {
        cout<<brojac<<" ";
    }
    return 0;
}
```

20

PRIMJER 52

- Provjera programa:



Sanda, 2015.

21

PRIMJER 53

- Treba ispisati parne brojeve iz raspona od 100 do 50.

Parni brojevi iz raspona od 100 do 50 su:
100 98 96 94 54 52 50

Sanda, 2015.

22

PRIMJER 53

- Napomene:
 - Prirast može biti i negativan, tada se vrijednost kontrolne varijable smanjuje.
 - Ako je prirast negativan, početna vrijednost kontrolne varijable mora biti veća od završne.

brojac=brojac-2 (brojac-=2)

Sanda, 2015.

23

```
#include<iostream>
using namespace std;
int main()
{
    int brojac;
    cout<<endl<<"Parni brojevi iz raspona od \
    100 do 50 su:"<<endl;
    for (brojac=100;brojac>=50;brojac-=2)
    {
        cout<<brojac<<" ";
    }
    return 0;
}
```

Sanda, 2015.

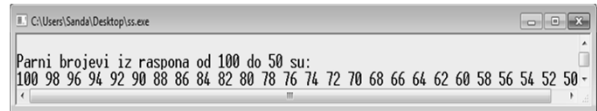
24

```
#include<iostream>
using namespace std;
int main()
{
    int brojac;
    cout<<endl<<"Parni brojevi iz raspona od \
100 do 50 su:"<<endl;
    for (brojac=100;brojac>=50;brojac-=2)
    {
        cout<<brojac<<" ";
    }
    return 0;
}
```

25

PRIMJER 53

- Provjera programa:



Sanda, 2015.

26

PRIMJER 54



- Treba potražiti pa ispisati brojeve djeljive sa 7 iz raspona od M do N.

Upisi pocetni broj raspona:
Upisi završni broj raspona:

Brojevi djeljivi sa 7 iz raspona od ... do ... su:
... ..

Sanda, 2015.

27

PRIMJER 54

- Napomene:
 - Blok naredbi u petlji izvršava se za svaki broj iz zadanog raspona.
 - Ispisuju se samo brojevi djeljivi s brojem 7.
 - Djeljivost brojeva se provjerava operatorom modulo (%).

Sanda, 2015.

28

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,m,n;
    cout<<"Upisi pocetni broj raspona:"; cin>>m;
    cout<<"Upisi završni broj raspona:"; cin>>n;
    cout<<endl<<"Brojevi djeljivi sa 7 iz raspona od "
    <<m<<" do "<<n<<" su:"<<endl;
    for (brojac=m;brojac<=n;brojac++)
    {
        if (brojac%7==0)
            cout<<brojac<<" ";
    }
    return 0;
}
```

Sanda, 2015.

29

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,m,n;
    cout<<"Upisi pocetni broj raspona:"; cin>>m;
    cout<<"Upisi završni broj raspona:"; cin>>n;
    cout<<endl<<"Brojevi djeljivi sa 7 iz raspona od "
    <<m<<" do "<<n<<" su:"<<endl;
    for (brojac=m;brojac<=n;brojac++)
    {
        if (brojac%7==0)
            cout<<brojac<<" ";
    }
    return 0;
}
```

30

PRIMJER 54

- Provjera programa:

```
C:\Users\Sanda\Desktop\ss.exe
Upisi pocetni broj raspona:40
Upisi završni broj raspona:100

Brojevi djeljivi sa 7 iz raspona od 40 do 100 su:
42 49 56 63 70 77 84 91 98
```

Sanda, 2015.

31

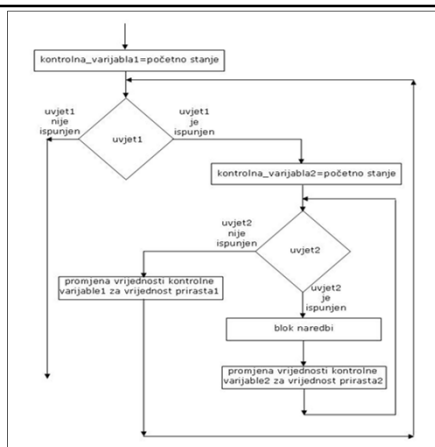
UGNIJEŽĐENE FOR PETLJE

- for petlje** mogu biti ugniježdene jedna unutar druge.
- Glavna značajka takvih petlji je **izvršavanje cjelokupne unutarnje petlje** za **svaku vrijednost kontrolne varijable vanjske petlje**.

Sanda, 2015.

32

UGNIJEŽĐENE FOR PETLJE



Sanda, 2015.

33

PRIMJER 55

- Vrijednost kontrolne varijable vanjske petlje mijenja se od 1 do 3, a unutarnje od 1 do 5.
- U programu se ispisuju trenutne vrijednosti kontrolnih varijabli vanjske i unutarnje petlje.

```
Vanjska petlja: i=1
Unutarnja petlja: j=1 j=2 j=3 j=4 j=5

Vanjska petlja: i=2
Unutarnja petlja: j=1 j=2 j=3 j=4 j=5

Vanjska petlja: i=3
Unutarnja petlja: j=1 j=2 j=3 j=4 j=5
```

Sanda, 2015.

34

PRIMJER 55

- Vanjska petlja omogućit će :
 - ispis teksta: Vanjska petlja:
 - ispis trenutne vrijednosti kontrolne varijable vanjske petlje (i)
 - ispis teksta: Unutarnja petlja:
- Unutarnja petlja :
 - ispisuje trenutne vrijednosti kontrolne varijable unutarnje petlje (j, od 1 do 5).
- Po ispisu svakoga od redaka, kazalo se prebacuje na početak novog retka i postupak se ponavlja.

Sanda, 2015.

35

```
#include <iostream>
using namespace std;
int main()
{
    int i, j;
    for ( i = 1; i <= 3; i++ )
    {
        cout<<endl<<"Vanjska petlja: i="<<i<<endl;
        cout<<"Unutarnja petlja: ";
        for ( j = 1; j <= 5; j++ )
        {
            cout<<" j="<<j;
        }
        cout<<endl;
    }
    return 0;
}
```

Vanjska petlja.

Kazalo se pomiče na početak novog reda.

Unutarnja petlja.

Sanda, 2015.

36

```
#include <iostream>
using namespace std;
int main()
{
    int i, j;
    for ( i = 1; i <= 3; i++ )
    {
        cout<<endl<<"Vanjska petlja: i="<<i<<endl;
        cout<<"Unutarnja petlja: ";
        for ( j = 1; j <= 5; j++ )
        {
            cout<<" j="<<j;
        }
        cout<<endl;
    }
    return 0;
}
```

37

PRIMJER 55

- Provjera programa:

```
C:\Users\Sanda\Desktop>ss.exe
Vanjska petlja: i=1
Unutarnja petlja: j=1 j=2 j=3 j=4 j=5
Vanjska petlja: i=2
Unutarnja petlja: j=1 j=2 j=3 j=4 j=5
Vanjska petlja: i=3
Unutarnja petlja: j=1 j=2 j=3 j=4 j=5
```

Sanda, 2015.

38

PRIMJER 56

- Treba ispisati tablicu množenja za brojeve od 1 do 10.

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

Sanda, 2015.

39

PRIMJER 56

- Napomene:

- Vanjska** petlja omogućit će stvaranje **10 redaka**.
- Unutarnja** petlja će u svakome redu stvoriti **10 stupaca**.
- Naredba koja se izvršava u unutarnjoj petlji je **ispis umnoška** trenutnih vrijednosti kontrolnih varijabli vanjske i unutarnje petlje.
- Po ispisu svakoga od redaka, kazalo se prebacuje **na početak novog retka**.

Sanda, 2015.

40

```
#include<iostream>
using namespace std;
int main()
{
    int stup,red;
    for(red=1;red<=10;red++)
    {
        for(stup=1;stup<=10;stup++)
        {
            cout<<red*stup<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

Sanda, 2015.

41

```
#include<iostream>
using namespace std;
int main()
{
    int stup,red;
    for(red=1;red<=10;red++)
    {
        for(stup=1;stup<=10;stup++)
        {
            cout<<red*stup<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

42

PRIMJER 56

- Ispis je dosta nepregledan, trebalo bi ga oblikovati tako da bude u pravilnim stupcima.

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

Sanda, 2015.

43

OPERATORI ZA RUKOVANJE ISPISOM

- Ispis u pravilnim stupcima može se oblikovati operatorom za rukovanje ispisom (**manipulatorom**):
 - setw(int)**
- setw(int)** određuje koliki će prostor biti predviđen za ispis podatka koji slijedi u izlaznom toku.
- Svi su manipulatori za rukovanje ispisom definirani u biblioteci **iomanip**.

Sanda, 2015.

44

setw(int)

- setw(int)** se upotrebljava unutar naredbe izlaznog toka, npr.:

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
    cout<<setw(8)<<"PRVI"<<setw(10)<<"DRUGI"<<endl;
    return 0;
}
```

8	10
PRVI	DRUGI

Sanda, 2015.

45

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
    int stup, red;
    for (red=1; red<=10; red++)
    {
        for (stup=1; stup<=10; stup++)
        {
            cout<<setw(7)<<red*stup;
        }
        cout<<endl;
    }
    return 0;
}
```

Uključivanje biblioteke.

Oblikovanje ispisa u stupcima željene širine.

Sanda, 2015.

46

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
    int stup, red;
    for (red=1; red<=10; red++)
    {
        for (stup=1; stup<=10; stup++)
        {
            cout<<setw(7)<<red*stup;
        }
        cout<<endl;
    }
    return 0;
}
```

47

PRIMJER 56

- Provjera programa:

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

Sanda, 2015.

48

setprecision(int)

- **setprecision(int)** manipulator je koji omogućava zadavanje decimalne preciznosti pri ispisu realnih brojeva.
- Svi su manipulatori za rukovanje ispisom definirani u biblioteci **iomanip**.

```
#include<iostream>
#include<iomanip>

cout<<setprecision(5)<<B<<endl;
```

Sanda, 2015.

49

PRIMJER 57



- Realni broj **8.987656** ispisati četiri puta, svaki put sa različitom decimalnom preciznošću: 1, 3, 5, 7 decimalnih znamenaka.
- Promisliti o ispisu. Može li se uočiti pravilnost?

Sanda, 2015.

50

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
    float B;
    B=8.987656;
    cout<<setprecision(1)<<B<<endl;
    cout<<setprecision(3)<<B<<endl;
    cout<<setprecision(5)<<B<<endl;
    cout<<setprecision(7)<<B<<endl;
    return 0;
}
```

Sanda, 2015.

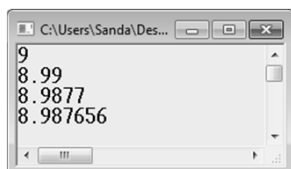
51

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
    float B;
    B=8.987656;
    cout<<setprecision(1)<<B<<endl;
    cout<<setprecision(3)<<B<<endl;
    cout<<setprecision(5)<<B<<endl;
    cout<<setprecision(7)<<B<<endl;
    return 0;
}
```

52

PRIMJER 57

- Provjera programa:



Sanda, 2015.

53

PRIMJER 58



- Potrebno je zbrojiti prvih N prirodnih brojeva.

Upisi završni broj raspona:

Zbroj prirodnih brojeva iz intervala od 1 do ...
je

Sanda, 2015.

54

PRIMJER 58

- Prije ulaska u petlju potrebno je:
 - Deklarirati varijablu koja će sadržavati trenutne vrijednosti kontrolne varijable petlje (npr. **zbroj**) pri svakom prolasku kroz petlju.
 - Varijabli **zbroj** pridružiti vrijednost 0.
- Naredba u bloku omogućava uvećavanje vrijednosti varijable **zbroj** za tekuću vrijednost kontrolne varijable petlje (**brojac**) pri svakom prolasku kroz petlju.

Sanda, 2015.

55

```
#include<iostream>
using namespace std;
int main()
{
    int N,brojac,zbroj;
    zbroj=0;
    cout<<"Upisi završni broj raspona: ";cin>>N;
    cout<<"Zbroj prirodnih brojeva iz raspona \
od 1 do " <<N<<" je ";
    for (brojac=1;brojac<=100;brojac++)
    {
        zbroj=zbroj+brojac;
    }
    cout<<zbroj<<endl;
    return 0;
}
```

Sanda, 2015.

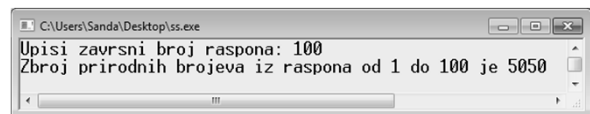
56

```
#include<iostream>
using namespace std;
int main()
{
    int N,brojac,zbroj;
    zbroj=0;
    cout<<"Upisi završni broj raspona: ";cin>>N;
    cout<<"Zbroj prirodnih brojeva iz raspona od 1 do "
    <<N<<" je ";
    for (brojac=1;brojac<=100;brojac++)
    {
        zbroj=zbroj+brojac;
    }
    cout<<zbroj<<endl;
    return 0;
}
```

57

PRIMJER 58

- Provjera programa:



Sanda, 2015.

58

PRIMJER 59



- Treba prebrojiti koliko prirodnih brojeva iz raspona od M do N ima znamenku jedinice vrijednosti 9.

Upisi početni broj raspona:
Upisi završni broj raspona:

U rasponu prirodnih brojeva od ... do ... ima ...
brojeva sa znamenkom jedinice vrijednosti 9. ...

Sanda, 2015.

59

PRIMJER 59

- Prije ulaska u petlju treba:
 - Deklarirati varijablu (npr. **N**) u kojoj će se prebrojavati pronađeni brojevi koji zadovoljavaju uvjet.
 - Varijabli **N** pridružiti vrijednost 0.
- Naredba u bloku omogućava uvećavanje vrijednosti varijable **N** za 1 svaki put kada se pronađe broj koji zadovoljava uvjet.

Sanda, 2015.

60

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,m,n,N;
    cout<<"Upisi pocetni broj raspona: ";cin>>m;
    cout<<"Upisi zavrzni broj raspona: ";cin>>n;
    N=0;
    for (brojac=m;brojac<=n;brojac++)
    {
        if(brojac%10==9)
            N=N+1;
    }
    cout<<"U rasponu prirodnih brojeva od "<<m
    <<" do "<<n<<" ima "<<N<<"brojeva sa znamenkom \
    jedinice vrijednosti 9. ";
    return 0;
}
```

Sanda, 2015.

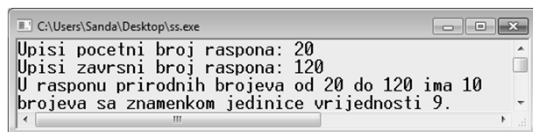
61

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,m,n,N;
    cout<<"Upisi pocetni broj raspona: ";cin>>m;
    cout<<"Upisi zavrzni broj raspona: ";cin>>n;
    N=0;
    for (brojac=m;brojac<=n;brojac++)
    {
        if(brojac%10==9)
            N=N+1;
    }
    cout<<"U rasponu prirodnih brojeva od "<<m<<" do "
    <<n<<" ima "<<N;
    cout<<" brojeva sa znamenkom jedinice vrijednosti 9. ";
    return 0;
}
```

62

PRIMJER 59

- Provjera programa:



Sanda, 2015.

63

WHILE PETLJA

- while petlja** se najčešće koristi ako **broj ponavljanja** bloka naredbi u petlji **nije unaprijed poznat**.
- Izvršava se sve **dok je ispunjen zadani uvjet** (uvjet je logički izraz).

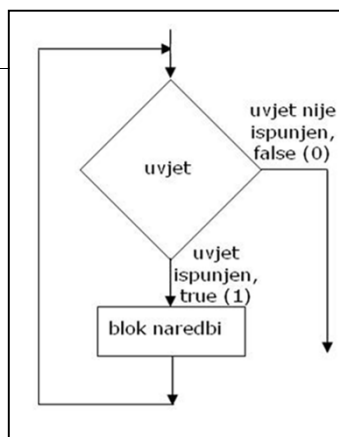
- Osnovni oblik:

```
while (uvjet)
{
    blok naredbi
}
naredba iza bloka
```

Sanda, 2015.

64

WHILE PETLJA



Sanda, 2015.

65

PRIMJER 60

- while petljom** ispisati brojeve iz raspona od M do N (raspon bira korisnik).

Upisi pocetni broj raspona:
Upisi zavrzni broj raspona:

Ispis brojeva od ... do ... :
.....

Sanda, 2015.

66

PRIMJER 60

- Petlja **while** može zamijeniti **for** petlju.
- Prije ulaska u petlju valja zadati **početnu vrijednost** varijable iz uvjeta (**brojac=m**).
- Zadati **uvjet** petlje (**brojac<=n**),
- U bloku naredbi u petlji uz ostalo zadati **prirast** (**brojac++**).

Sanda, 2015.

67

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,m,n;
    cout<<"Upisi pocetni broj raspona: ";cin>>m;
    cout<<"Upisi zavrzni broj raspona: ";cin>>n;
    brojac=m;
    cout<<"Ispis brojeva od "<<m<<" do "<<n<<endl;
    while(brojac<=n)
    {
        cout<<brojac<<" ";
        brojac=brojac+1;
    }
    return 0;
}
```

Početna
vrijednost.

Uvjet.

Prirast.

Sanda, 2015.

68

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,m,n;
    cout<<"Upisi pocetni broj raspona: ";cin>>m;
    cout<<"Upisi zavrzni broj raspona: ";cin>>n;
    brojac=m;
    cout<<"Ispis brojeva od "<<m<<" do "<<n<<endl;
    while(brojac<=n)
    {
        cout<<brojac<<" ";
        brojac=brojac+1;
    }
    return 0;
}
```

69

PRIMJER 60

- Provjera programa:

```
C:\Users\Sanda\Desktop>iss.exe
Upisi pocetni broj raspona: 1
Upisi zavrzni broj raspona: 20
Ispis brojeva od 1 do 20
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

Sanda, 2015.

70

PRIMJER 61



- Potrebno je unesenom prirodnom broju prebrojiti znamenke.

Upisi prirodni broj:

Broj ... ima ... znamenaka.

Sanda, 2015.

71

PRIMJER 61

- Broj znamenaka broja **B** nije unaprijed poznat.
- Algoritam bi trebao funkcionirati na sljedeći način:

345/10=34,	br=br+1=0+1=1,	34>0, DA
34/10=3,	br=br+1=1+1=2,	3>0, DA
3/10=0,	br=br+1=2+1=3,	0>0, NE

Sanda, 2015.

72

PRIMJER 61

- Uvjet petlje: $B > 0$.
- U petlji:
 - Od vrijednosti varijable **B** odvaja se znamenka najmanje težinske vrijednosti, a potom se dobiveni rezultat vraća u varijablu **B**.
 - Varijabla koja prebrojava znamenke uvećava se za 1.
- Provjerava se uvjet petlje.

Sanda, 2015.

73

```
#include<iostream>
using namespace std;
int main()
{
    int br,b,a;
    cout<<"Upisi prirodni broj: ";
    cin>>b;
    a=b;
    br=0;
    while(b>0)
    {
        b=b/10;
        br=br+1;
    }
    cout<<"Broj "<<a<<" ima "<<br<<" znamenki."<<endl;
    return 0;
}
```

Pohrana vrijednosti unesenog broja.

Varijablu koja će prebrojavati znamenke valja isprazniti.

Sanda, 2015.

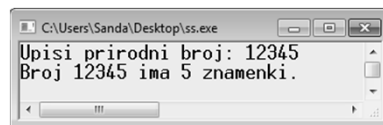
74

```
#include<iostream>
using namespace std;
int main()
{
    int br,b,a;
    cout<<"Upisi prirodni broj: "; cin>>b;
    a=b;
    br=0;
    while(b>0)
    {
        b=b/10;
        br=br+1;
    }
    cout<<"Broj "<<a<<" ima "<<br<<" znamenki."<<endl;
    return 0;
}
```

75

PRIMJER 61

- Provjera programa:



Sanda, 2015.

76

PRIMJER 62



- Treba unijeti N realnih brojeva (N NJE unaprijed poznat!!), a potom izračunati njihovu srednju vrijednost. Unos brojeva traje sve dok korisnik ne upiše broj 0.

Nakon posljednjeg broja unesi nulu!
Unesi broj:
Unesi broj:
...
Uneseno je ... brojeva. Srednja vrijednost je

Sanda, 2015.

77

PRIMJER 62

- Prije ulaska u petlju unosi se prvi broj (**N**) kao početna vrijednost varijable iz uvjeta petlje.
- U uvjetu se sadržaj varijable **N** uspoređuje s 0 ($N == 0$).
- U petlji:
 - uneseni broj **N** pribraja se varijabli **zbroy** (zbrajanje),
 - varijabla **broj** uvećava se za jedan (prebrojavanje),
 - unosi se sljedeći broj **N**.

Sanda, 2015.

78

```
#include<iostream>
using namespace std;
int main()
{
    float N,broj,zbroj;
    broj=0;
    zbroj=0;
    cout<<"Nakon posljednjeg broja unesi nulu!"<<endl;
    cout<<"Unesi broj:"; cin>>N;
    while(N!=0)
    {
        broj=broj+1;
        zbroj=zbroj+N;
        cout<<"Unesi broj:";
        cin>>N;
    }
    cout<<"Uneseno je "<<broj<<" brojeva. Srednja \
vrijednost je "<<zbroj/broj<<endl;
    return 0;
}
```

Sanda, 2015.

79

```
#include<iostream>
using namespace std;
int main()
{
    float N,broj,zbroj;
    broj=0;
    zbroj=0;
    cout<<"Nakon posljednjeg broja unesi nulu!"<<endl;
    cout<<"Unesi broj:";
    cin>>N;
```

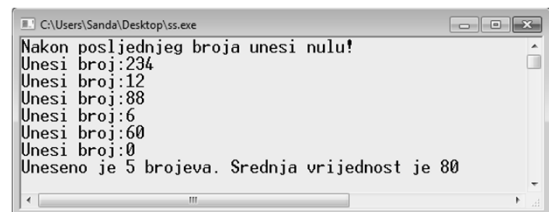
80

```
while(N!=0)
{
    broj=broj+1;
    zbroj=zbroj+N;
    cout<<"Unesi broj:";
    cin>>N;
}
cout<<"Uneseno je "<<broj<<" brojeva. Srednja \
vrijednost je "<<zbroj/broj<<endl;
return 0;
}
```

81

PRIMJER 62

- Provjera programa:



```
C:\Users\Sanda\Desktop\ss.exe
Nakon posljednjeg broja unesi nulu!
Unesi broj:234
Unesi broj:12
Unesi broj:88
Unesi broj:6
Unesi broj:60
Unesi broj:0
Uneseno je 5 brojeva. Srednja vrijednost je 80
```

Sanda, 2015.

82

DO - WHILE PETLJA

- **do-while petlja** se najčešće koristi ako **broj ponavljanja** bloka naredbi u petlji **nije unaprijed poznat**.
- Uvjet se ispituje **nakon izvođenja bloka naredbi u petlji**.
- Osnovni oblik:

```
do
{
    blok naredbi
}
while (uvjet);
```

Sanda, 2015.

83

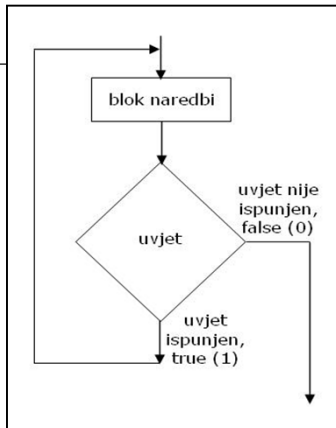
DO - WHILE PETLJA

- Blok naredbi u petlji **do-while izvodi se bar jednom** bez obzira **je li uvjet ispunjen ili ne**.
- U tome jest i glavna razlika **do-while** petlje u odnosu na petlje **for** i **while** kod kojih je moguće da se blok naredbi u petlji **ne izvede niti jednom** ako uvjet petlje nije ispunjen.

Sanda, 2015.

84

DO - WHILE PETLJA



Sanda, 2015.

85

PRIMJER 63

- Primjer 62. riješiti **do-while** petljom.

Nakon posljednjeg broja unesi nulu!
Unesi broj:
Unesi broj:
...
Uneseno je ... brojeva. Srednja vrijednost je

Sanda, 2015.

86

PRIMJER 63

- Prije ulaska u petlju **nije potrebno unijeti** prvi **broj (N)** jer se uvjet ispituje tek nakon izvršenja naredbi iz bloka petlje.
- U ovome algoritmu prebrojava se i 0 kao uneseni broj, pa je po izlasku iz petlje sadržaj varijable **broj** neispravan, veći je za 1.
- Stoga, nakon petlje, varijabla **broj** umanjuje se za 1.

Sanda, 2015.

87

```
#include<iostream>
using namespace std;
int main()
{
    float N,broj,zbroj;
    broj=0;
    zbroj=0;
    cout<<"Nakon posljednjeg broja unesi nulu!"<<endl;
    do
    {
        cout<<"Unesi broj:"; cin>>N;
        broj=broj+1;
        zbroj=zbroj+N;
    }
    while (N!=0);
    broj=broj-1;
    cout<<"Uneseno je "<<broj<<" brojeva. Srednja \
vrijednost je "<<zbroj/broj<<endl;
    return 0;
}
```

Sanda, 2015.

88

```
#include<iostream>
using namespace std;
int main()
{
    float N,broj,zbroj;

    broj=0;

    zbroj=0;

    cout<<"Nakon posljednjeg broja unesi nulu!"<<endl;
```

89

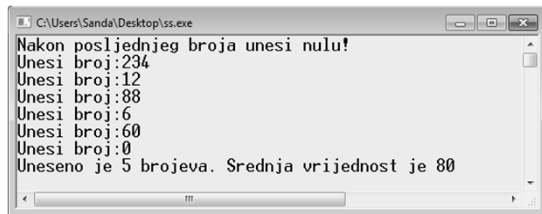
```
while (N!=0)
{
    cout<<"Unesi broj:";
    cin>>N;
    broj=broj+1;
    zbroj=zbroj+N;
}

broj=broj-1;
cout<<"Uneseno je "<<broj<<" brojeva. Srednja \
vrijednost je "<<zbroj/broj<<endl;
return 0;
}
```

90

PRIMJER 63

- Provjera programa:



```
Nakon posljednjeg broja unesi nulu!
Unesi broj:234
Unesi broj:12
Unesi broj:88
Unesi broj:6
Unesi broj:60
Unesi broj:0
Uneseno je 5 brojeva. Srednja vrijednost je 80
```

Sanda, 2015.

91

NAREDBA BREAK

- Naredba **break** rabi se kada treba **prekinuti izvođenje bloka naredbi** unutar petlje.
- Nakon naredbe **break** program izvršenje nastavlja **od prve naredbe iza programske petlje**.
(Sastavni je dio **switch-case** naredbe.)

Sanda, 2015.

92

PRIMJER 64



- Treba unijeti 10 prirodnih brojeva pa ih zbrojiti.
- U slučaju unosa broja 5, unos se prekida. Ispisuje se poruka o unosu broja 5, a potom i broj unesenih brojeva do unosa petice te njihov zbroj.

Upisi broj:
Upisi broj:
...

Prekid zbog unosa broja 5.
Uneseno je ... brojeva. Njihov je zbroj ...

Sanda, 2015.

93

```
#include <iostream>
using namespace std;
int main ()
{
    int n,A,zbroj,broj;
    zbroj=0;
    broj=0;
    for (n=1; n<=10; n++)
    {
        cout <<"Upisi broj:";
        cin>>A;
        if (A==5)
        {
            cout<<"Prekid zbog unosa broja 5"<<endl;
            break;
        }
        broj=broj+1;
        zbroj=zbroj+A;
    }
    cout<<endl<<"Uneseno je "<<broj<<" brojeva. Njihov \
je zbroj "<<zbroj<<endl;
    return 0;
}
```

U slučaju unosa
broja 5, unos se
prekida i iskakuje se
iz petlje.

Sanda, 2015.

94

```
#include <iostream>
using namespace std;
int main ()
{
    int n,A,zbroj,broj;
    zbroj=0;
    broj=0;
```

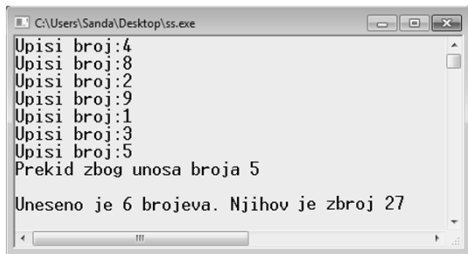
95

```
for (n=1; n<=10; n++)
{
    cout <<"Upisi broj:";
    cin>>A;
    if (A==5)
    {
        cout<<"Prekid zbog unosa broja 5"<<endl;
        break;
    }
    broj=broj+1;
    zbroj=zbroj+A;
}
cout<<endl<<"Uneseno je "<<broj<<" brojeva. Njihov \
je zbroj "<<zbroj<<endl;
return 0;
}
```

96

PRIMJER 64

- Provjera programa:



```
C:\Users\Sanda\Desktop\ss.exe
Upisi broj:4
Upisi broj:8
Upisi broj:2
Upisi broj:9
Upisi broj:1
Upisi broj:3
Upisi broj:5
Prekid zbog unosa broja 5

Uneseno je 6 brojeva. Njihov je zbroj 27
```

Sanda, 2015.

97

NAREDBA CONTINUE

- Ponekad je potrebno **preskočiti dio bloka naredbi do kraja programske petlje**, pa nakon toga **petlju nastaviti**.
- U tu se svrhu koristi naredba **continue**.

(Bolje je, umjesto naredbe **continue** rabiti naredbu grananja **if** jer nepažljiva uporaba naredbe **continue** može uzrokovati beskonačnu petlju.)

Sanda, 2015.

98

PRIMJER 65

- Treba unijeti 10 prirodnih brojeva, pa ih zbrojiti.
- U slučaju unosa broja 5, ispisuje se poruka o unosu broja 5, preskače se dio bloka do kraja programske petlje pa se nakon toga petlja nastavlja.
- Na kraju se ispisuje broj ispravno unesenih brojeva te njihov zbroj.

Sanda, 2015.

99

PRIMJER 65

Upisi broj:
Upisi broj:
Upisi broj:
...

Uneseno je ... brojeva koji nisu 5. Njihov je zbroj ...

Ako se unese broj 5, ispisat će se poruka:

Unesen je broj 5. On se ne pribraja i ne prebrojava.

Sanda, 2015.

100

```
#include <iostream>
using namespace std;
int main ()
{
    int n,A,zbroj,broj;
    zbroj=0;
    broj=0;
    for (n=1; n<=10; n++)
    {
        cout <<"Upisi broj:";
        cin>>A;
        if (A==5)
        {
            cout<<"Unesen je broj 5. On se ne \
            pribraja i ne prebrojava."<<endl;
            continue;
        }
        broj=broj+1;
        zbroj=zbroj+A;
    }
    cout<<endl<<"Uneseno je "<<broj<<" brojeva \
    koji nisu 5. Njihov je zbroj "<<zbroj<<endl;
    return 0;
}
```

U slučaju unosa broja 5, preskače se dio bloka, a potom se unos nastavlja.

Sanda, 2015.

101

```
#include <iostream>
using namespace std;
int main ()
{
    int n,A,zbroj,broj;
    zbroj=0;
    broj=0;
```

102

```

for (n=1; n<=10; n++)
{
    cout <<"Upisi broj:";
    cin>>A;
    if (A==5)
    {
        cout<<"Unesen je broj 5. On se ne
        \ pribraja i ne prebrojava."<<endl;
        continue;
    }
    broj=broj+1;
    zbroj=zbroj+A;
}
cout<<endl<<"Uneseno je "<<broj<<" brojeva koji \
nisu 5. Njihov je zbroj "<<zbroj<<endl;
return 0;
}

```

103

PRIMJER 65

■ Provjera programa:

```

C:\Users\Sanda\Desktop\ss.exe
Upisi broj:4
Upisi broj:8
Upisi broj:2
Upisi broj:5
Unesen je broj 5. On se ne pribraja i ne prebrojava.
Upisi broj:1
Upisi broj:3
Upisi broj:5
Unesen je broj 5. On se ne pribraja i ne prebrojava.
Upisi broj:3
Upisi broj:5
Unesen je broj 5. On se ne pribraja i ne prebrojava.
Upisi broj:7
Uneseno je 7 brojeva koji nisu 5. Njihov je zbroj 28

```

Sanda, 2015.

104