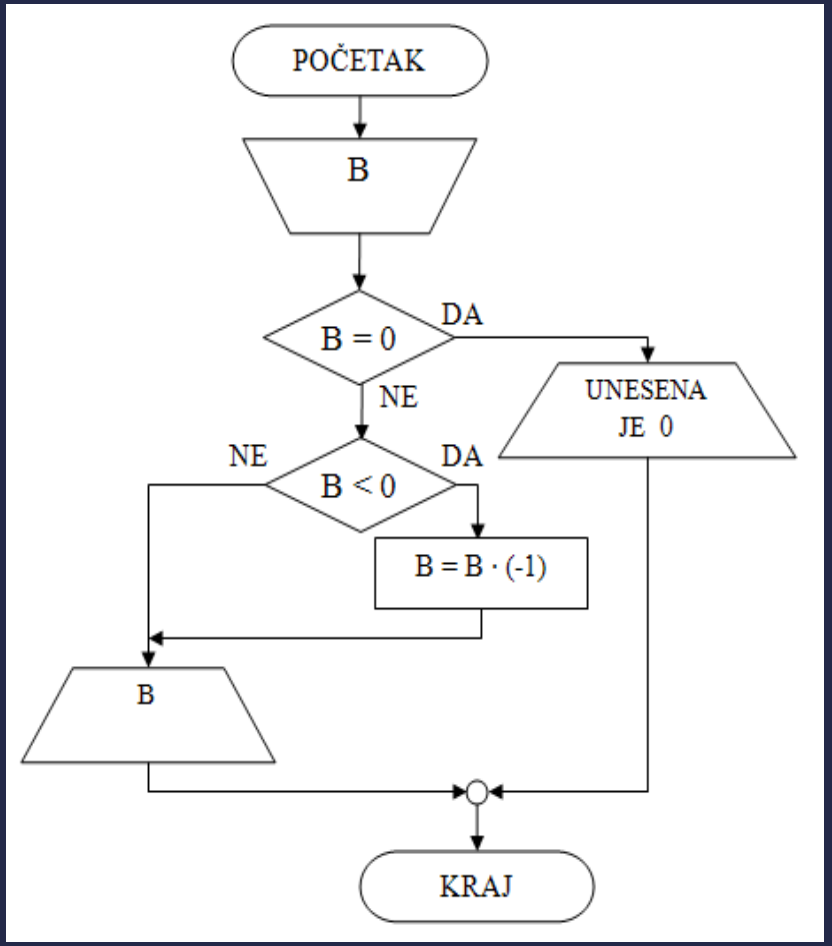
Instrukcije za određivanje tijeka programa

Programska struktura grananja

- Za rješavanje većine zadataka potrebne su programske strukture kod kojih redoslijed izvršavanja naredbi ovisi o vrijednostima podataka koji se obrađuju.
- Grananje je programska struktura koja omogućuje različit tijek programa, ovisno o rezultatu postavljenog uvjeta.

Programska struktura grananja

- Primjer programske strukture grananja:
  - Korisnik unosi cijeli broj, a zatim se računa apsolutna vrijednost tog broja i ispisuje rezultat.

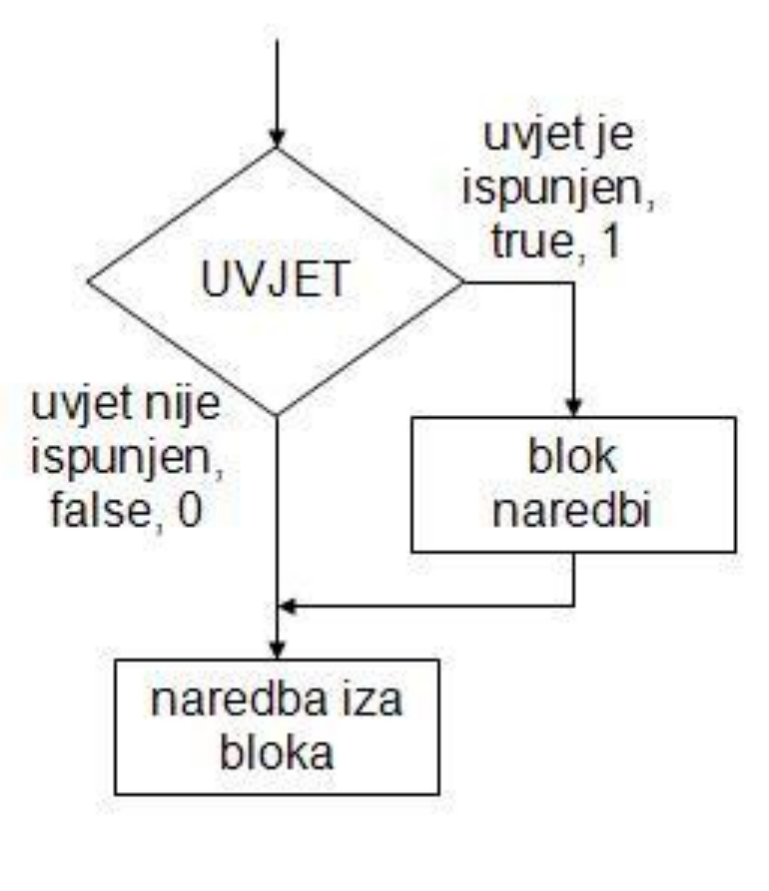


Instrukcije grananja

Ključne riječi pseudokoda	Primjeri
ako je uvjet **onda** ... **inače** ...	**ako je** (x < 0) **onda ispiši** (“Negativan”); **ako je** (b <> 0) **onda** kvocijent = a / b; **ispiši** (kvocijent); **inače** **ispiši** (“Dijeljenje s nulom”);

Jednostruko uvjetno grananje

- Jednostruko uvjetno grananje omogućava izvršenje bloka naredbi samo ako je zadani uvjet ispunjen.
- Ako uvjet nije ispunjen izvršava se prva naredba nakon bloka.



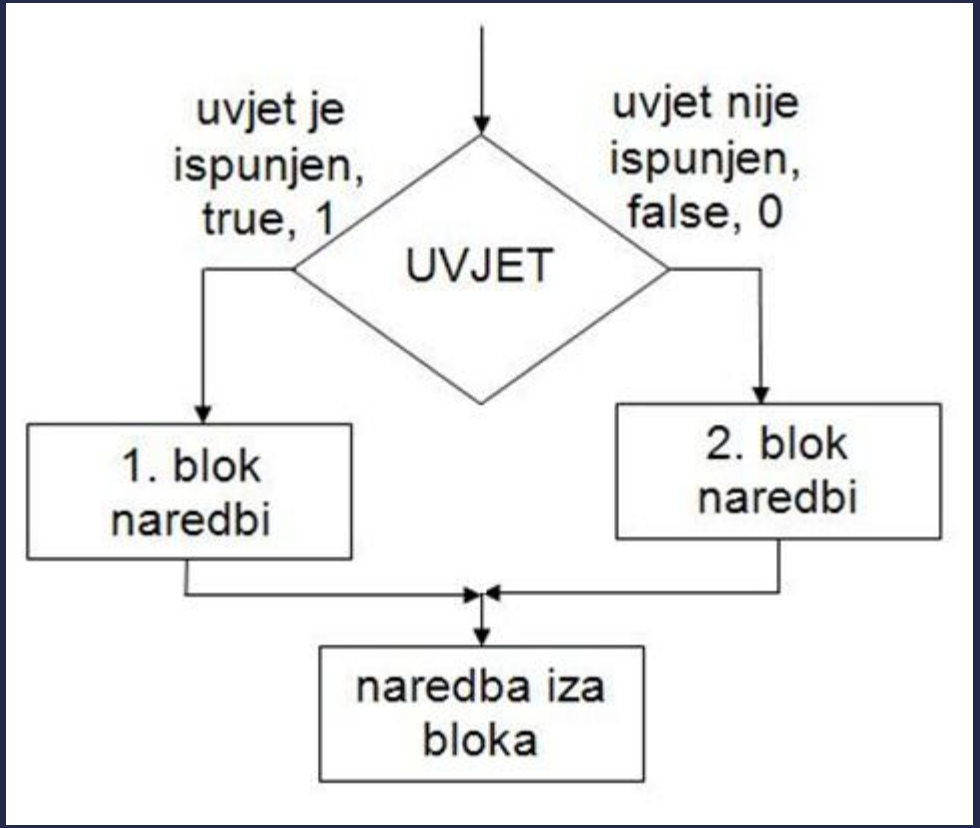
Primjer 1

- Provjeriti je li uneseni broj jednak 5 i ispisati “Broj je jednak 5” ukoliko je zadovoljen uvjet. Potom broj umanjiti za 2 i ispisati vrijednost broja.

```
početak
        učitaj (a);
        ako je (a==5) onda
                ispiši (“Broj je jednak 5”);
        a = a – 2;
        ispiši (a);
kraj
```

Dvostruko uvjetno grananje

- Dvostruko uvjetno grananje omogućava da se ovisno o ispunjenju postavljenog uvjeta izvodi jedan od dva neovisna bloka naredbi.



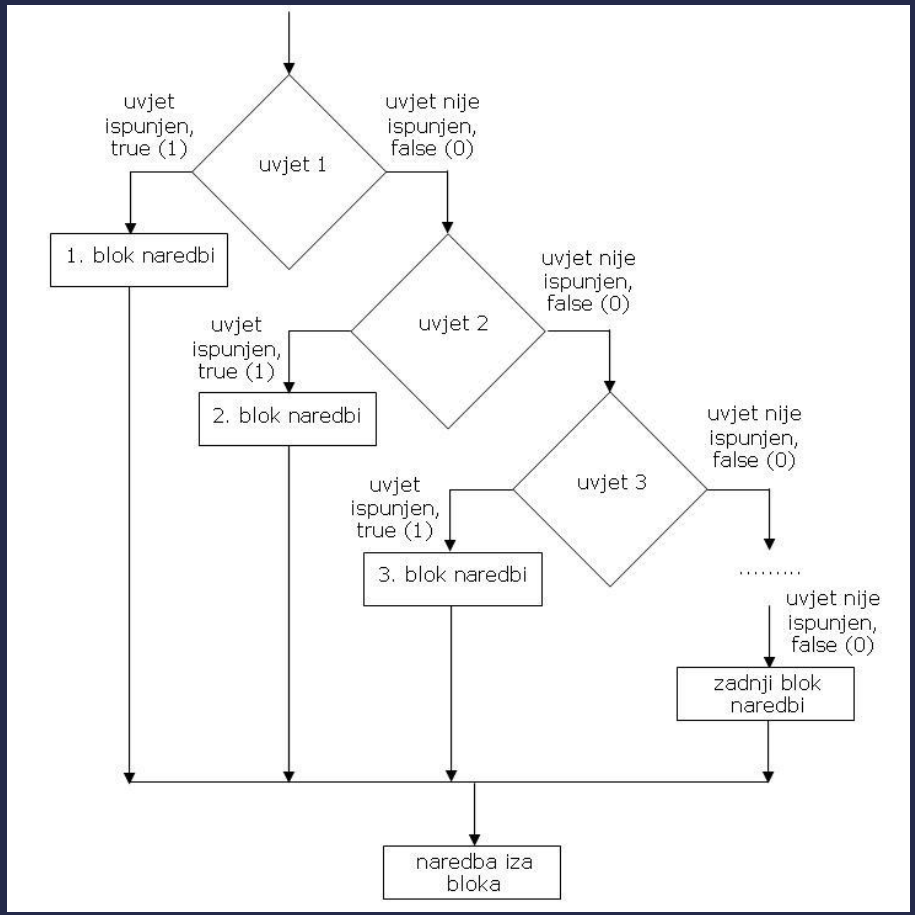
Primjer 2

- Provjeriti je li uneseni broj jednak 5 i ispisati “Broj je jednak 5” ukoliko je zadovoljen uvjet. Ukoliko broj nije jednak 5 ispisati “Broj nije jednak 5”.

```
početak
        učitaj (a);
        ako je (a==5) onda
                ispiši (“Broj je jednak 5”);
        inače
                ispiši (“Broj nije jednak 5”);
kraj
```

Višestruko uvjetno grananje

- Višestruko grananje omogućava ispitivanje više uvjeta.
- Ovisno o ispunjenju postavljenih uvjeta izvodi se odgovarajući blok naredbi.



Primjer 3

- Provjeriti je li uneseni broj jednak 5 i ispisati “Broj je jednak 5” . Ukoliko nije zadovoljen prethodni uvjet provjeriti je li broj veći od 5 te ispisati “Broj je veći od 5”. U slučaju da nisu zadovoljena dva prethodna uvjeta ispisati “Broj je manji od 5”.

```
početak
        učitaj (a);
        ako je (a==5) onda
                ispiši (“Broj je jednak 5”);
        ako je (a > 5) onda
                ispiši (“Broj je veći od 5”);
        inače
                ispiši (“Broj je manji od 5”);
kraj
```

Zadatak 1

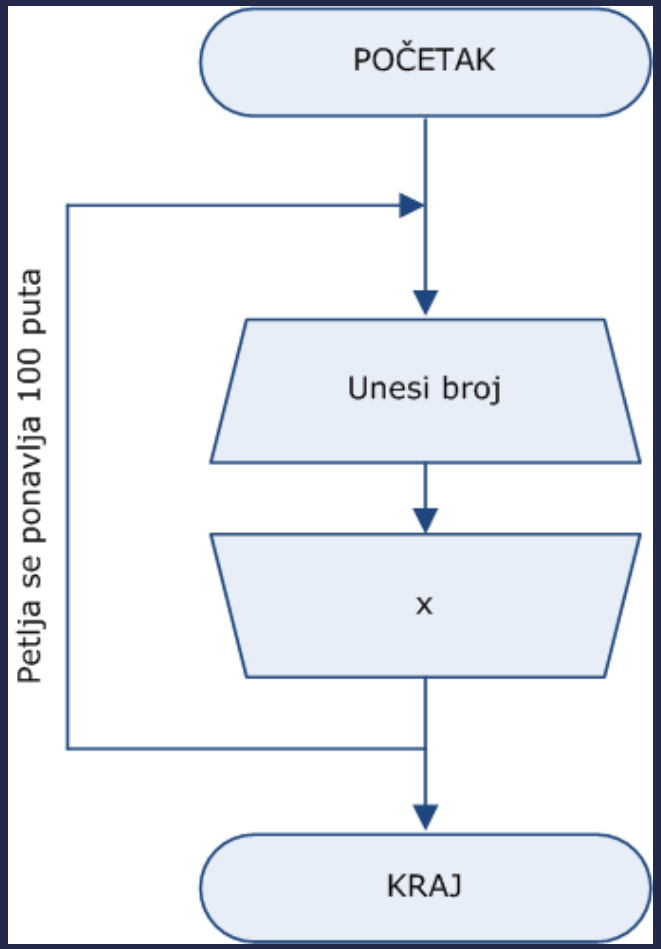
- Provjeriti je li uneseni broj djeljiv s dva, ukoliko jest ispisati “Broj je paran”, ukoliko nije ispisati “Broj je neparan”.

Programska struktura petlje

- Često je u programu potrebno ponoviti neku radnju više puta (iterirati). Takva se programska struktura naziva petlja.
- Ponavljati se može:
  - unaprijed zadani broj puta,
  - sve dok je ispunjen zadani uvjet.

Programska struktura petlje

- Kao primjer programske strukture petlje može poslužiti zadatak:
- Korisnik treba unijeti 100 cijelih brojeva, a prije svakog unesa broja ispisuje se tekst "Unesi broj".



Petlje

- Struktura petlje se može ostvariti naredbama:
  - *for* – instrukcije za ponavljanje radnje unaprijed zadani broj puta
  - *while* – instrukcije za ponavljanje s ispitivanjem uvjeta na početku
  - *do – while* – instrukcije za ponavljanje s ispitivanjem uvjeta na kraju

for petlja

- *for* petlja se najčešće koristi ako se dijelovi programa trebaju ponoviti unaprijed poznati broj puta.

Ključne riječi pseudokoda	Primjer

za brojac = pv **do** kv
činiti ...
 	
Brojač određuje broj ponavljanja!
pv = početna vrijednost
kv = konačna vrijednost
	Izračunavanje umnoška (produkta) 10 učitanih brojeva.
	
	produkt = 1;
	za i = 1 **do** 10 **činiti**
	učitaj (x);
	produkt = produkt * x;
	ispiši (produkt);

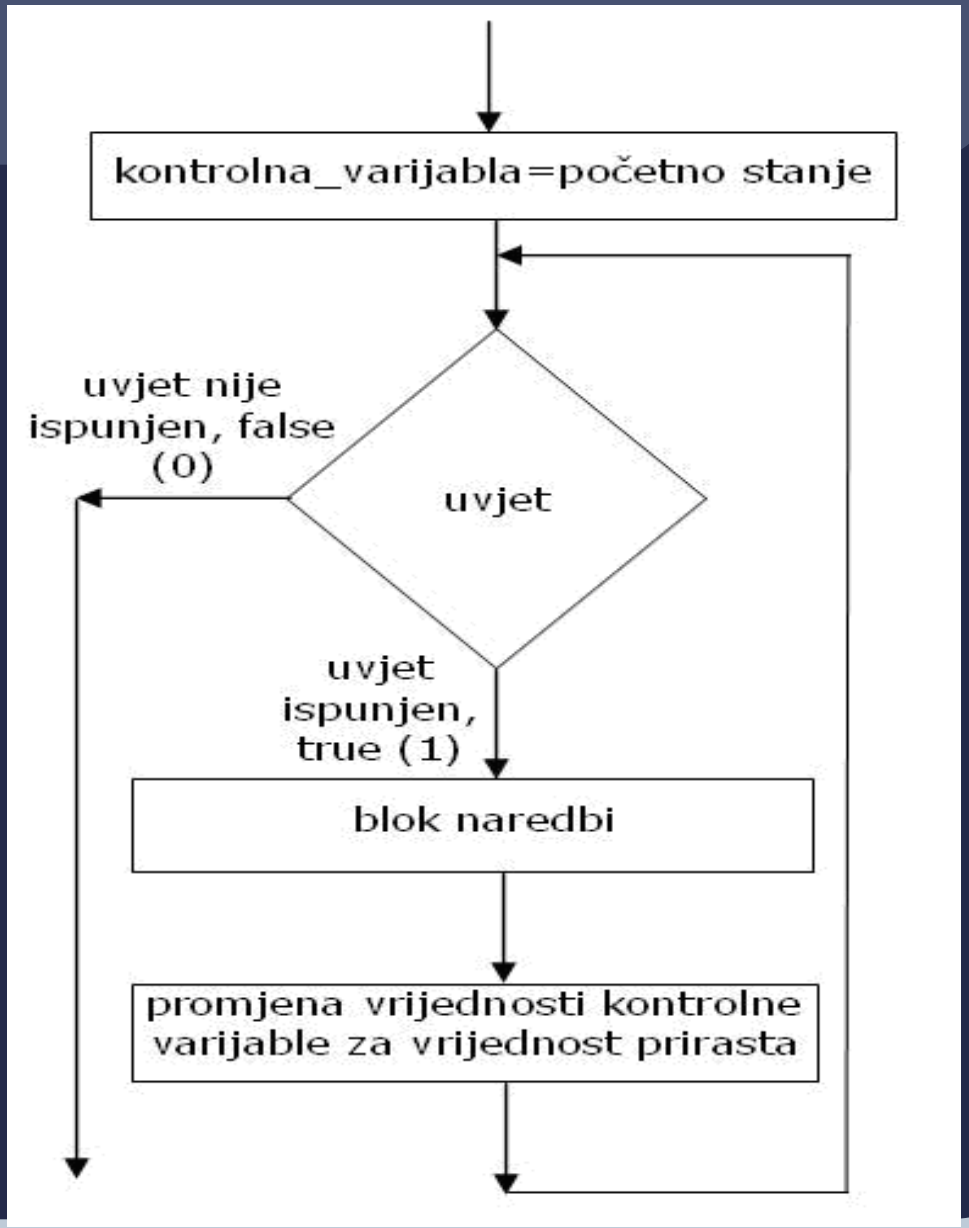
for petlja

- Svaka *for* petlja ima svoju kontrolnu varijablu.
- Njena se vrijednost svakim prolaskom kroz petlju mijenja ovisno o vrijednosti prirasta.
- U petlji se najprije zadaje vrijednost početnog stanja kontrolne varijable petlje (početno stanje)
  - npr. `brojac = 1;` - početna vrijednost brojača

for petlja

- Po zadavanju početnog stanja kontrolne varijable petlje zapisuje se uvjet.
- Rezultat uvjeta je podatak tipa bool (0 ili 1).
- Blok naredbi se izvršava sve dok je vrijednost uvjeta jednaka logičkoj istini (1).
- Kad vrijednost uvjeta postane neistina (0) petlja se prekida.

for petlja


kontrolna_varijabla=početno stanje
uvjet nije ispunjen, false (0)
uvjet
uvjet ispunjen, true (1)
blok naredbi
promjena vrijednosti kontrolne varijable za vrijednost prirasta

Primjer 4

- Potrebno je napisati program kojim se ispisuje brojevi od 1 do 20.

```
početak
        za brojac = 1 do 20 činiti
                ispiši (brojac);
kraj
```

Primjer 5

- Potrebno je napisati program kojim se ispisuje 20 učitanih brojeva (unosi ih korisnik).

```
početak
        za brojac = 1 do 20 činiti
                učitaj (x);
                ispiši (x);
kraj
```

Primjer 6

- Potrebno je napisati program kojim se ispisuje zbroj brojeva od 1 do 20.

```
početak
        zbroj = 0;
        za brojac = 1 do 20 činiti
                zbroj = zbroj + brojac;
        ispiši (zbroj);
kraj
```

Primjer 7

- Potrebno je napisati program kojim se ispisuje zbroj 20 učitanih brojeva.

```
početak
        zbroj = 0;
        za brojac = 1 do 20 činiti
                učitaj (x);
                zbroj = zbroj + x;
        ispiši (zbroj);
kraj
```

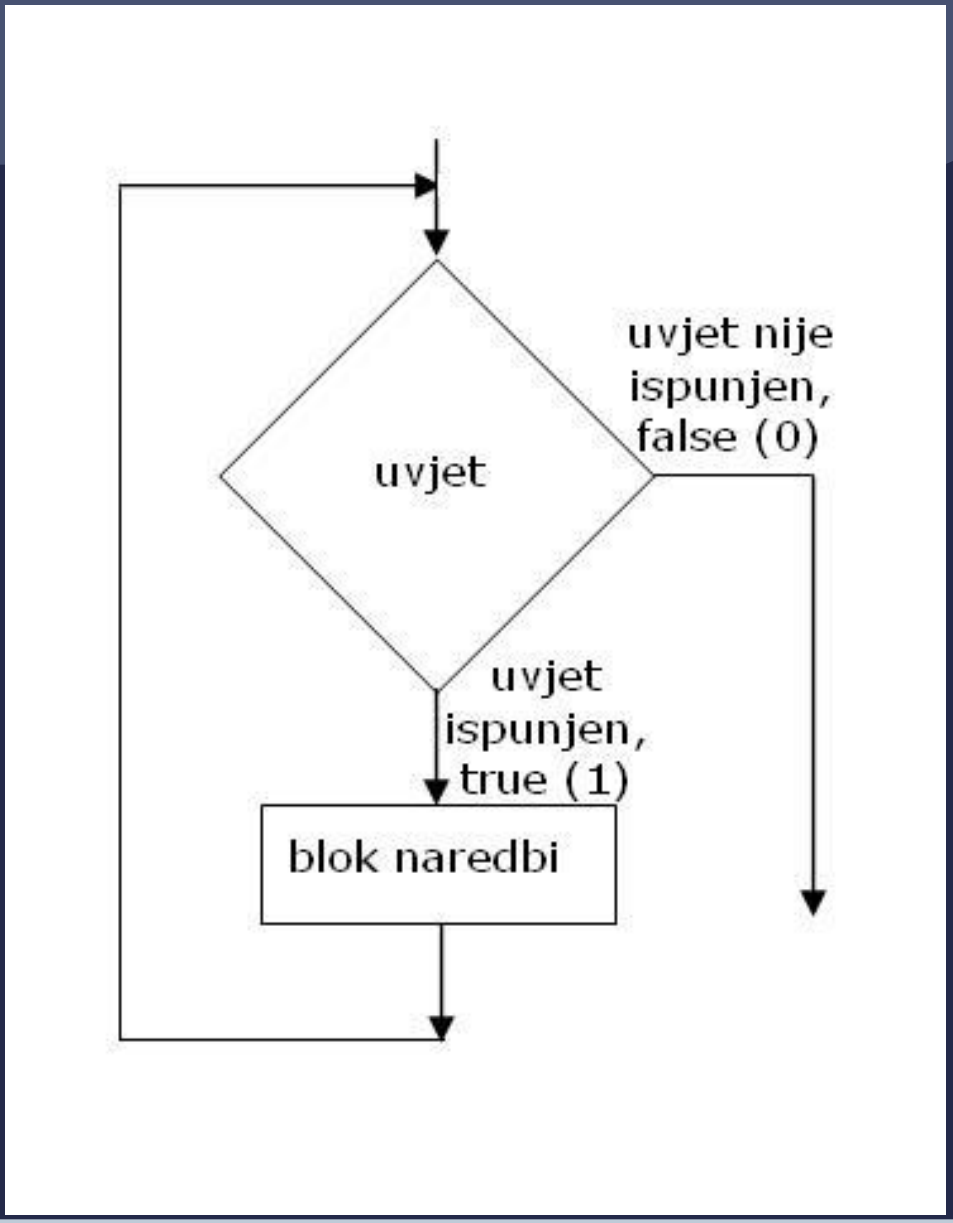
Zadatak 2

- Potrebno je napisati program koji će od 20 unesenih brojeva (unosi ih korisnik) ispisati samo one koji su parni brojevi (djeljivi s 2!).
- Npr.:
  - Za uneseni niz brojeva:
    - 5, 7, 3, 5, 4, 9, 8, 8, 6, 13, 11, 12, 14, 20, 20, 2, 3, 13, 19, 10
  - Ispisati će se sljedeći brojevi:
    - 4, 8, 8, 6, 12, 14, 20, 20, 2, 10

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 čiji rezultat mora biti tipa *bool*.
- Kada rezultat postane logička neistina (false, 0), preskače se blok naredbi petlje i program se nastavlja od prve naredbe nakon bloka naredbi petlje.

while petlja



while petlja

Ključne riječi pseudokoda	Primjer
dok je uvjet **činiti** … *(Napomena: instukcije u petlji izvršavaju se dok je god uvjet istinit)* *UVJET ISPITUJEMO NA POČETKU PETLJE!*	**dok je** (broj < a) **činiti** **ispiši** (broj); broj = broj + 1;

Primjer 8

- Potrebno je napisati program kojim se ispisuje brojevi od 1 do 20.

```
početak
        brojac = 1;
        dok je (brojac < 20) činiti
                ispiši (brojac);
                brojac = brojac + 1;
kraj
```

Primjer 9

- Potrebno je napisati program kojim se ispisuje 20 učitanih brojeva (unosi ih korisnik).

```
početak
        brojac = 1;
        dok je (brojac < 20) činiti
                učitaj (x);
                ispiši (x);
                brojac = brojac + 1;
kraj
```

Primjer 10

- Potrebno je napisati program kojim se ispisuje zbroj brojeva od 1 do 20.

```
početak
        zbroj = 0;
        brojac = 1;
        dok je (brojac < 20) činiti
                zbroj = zbroj + brojac;
                brojac = brojac + 1;
        ispiši (zbroj);
kraj
```

Primjer 11

- Potrebno je napisati program kojim se ispisuje zbroj 20 učitanih brojeva.

```
početak
        zbroj = 0;
        brojac = 1;
        dok je (brojac < 20) činiti
                učitaj (x);
                zbroj = zbroj + x;
                brojac = brojac + 1;
        ispiši (zbroj);
kraj
```

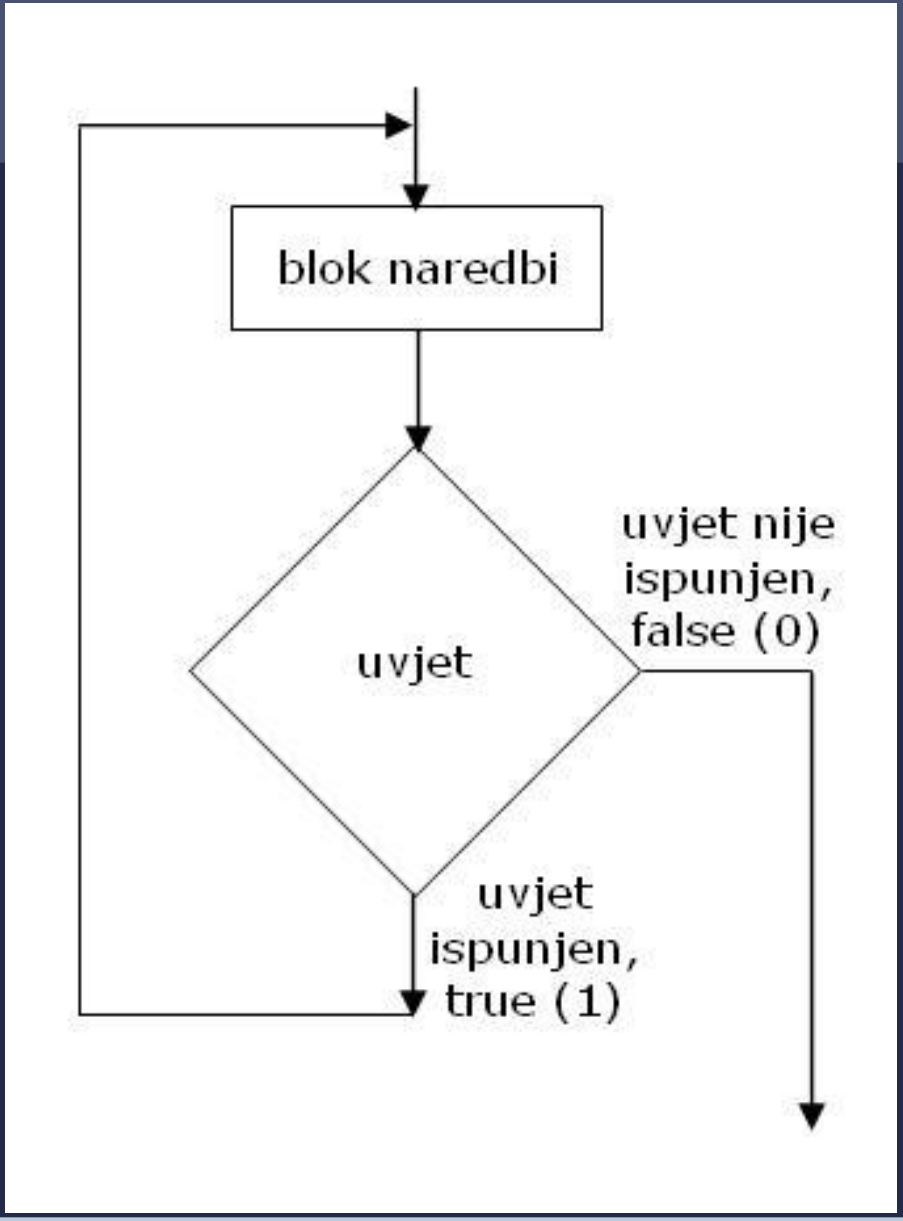
Zadatak 3

- Napisati program koji će ispisati sve višekratnike broja 5 manje od učitanog broja a te prebrojiti koliko ih ima i ispisati ukupan broj višekratnika.
- Npr.:
  - Za uneseni broj 20 program bi trebao ispisati:
    - 5, 10, 15
  - Za uneseni broj 23 program bi trebao ispisati:
    - 5, 10, 15, 20

do-while petlja

- *do-while* petlja se također koristi ako broj ponavljanja bloka naredbi u petlji nije unaprijed poznat.
- Za razliku od do sad opisanih petlji kod *do-while* petlje se uvjet ispituje nakon izvođenja bloka naredbi u petlji.

do-while petlja



do-while petlja

Ključne riječi pseudokoda	Primjer
ponavljati **…** **dok vrijedi** uvjet *(Napomena: instukcije u petlji izvršavaju se dok je god uvjet istinit)* *UVJET ISPITUJEMO NA POČETKU PETLJE!*	**učitaj** (broj); **ponavljati** **ispiši** (broj); broj = broj – 1; **dok vrijedi** (broj > 0)

do-while petlja

- Blok naredbi u petlji *do-while* izvodi se bar jednom bez obzira je li uvjet ispunjen ili ne.
- U tome je 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.

Primjer 8

- Potrebno je napisati program kojim se ispisuje brojevi od 1 do 20.

```
početak
        brojac = 1;
        ponavljaj
                ispiši (brojac);
                brojac = brojac + 1;
        dok vrijedi (brojac < 20)
kraj
```

Primjer 9

- Potrebno je napisati program kojim se ispisuje 20 učitanih brojeva (unosi ih korisnik).

```
početak
        brojac = 1;
        ponavljaj
                učitaj (x);
                ispiši (x);
                brojac = brojac + 1;
        dok vrijedi (brojac < 20)
kraj
```

Primjer 10

- Potrebno je napisati program kojim se ispisuje zbroj brojeva od 1 do 20.

```
početak
        zbroj = 0;
        brojac = 1;
        ponavljaj
                zbroj = zbroj + brojac;
                brojac = brojac + 1;
        dok vrijedi (brojac < 20)
        ispiši (zbroj);
kraj
```

Primjer 11

- Potrebno je napisati program kojim se ispisuje zbroj 20 učitanih brojeva.

```
početak
        zbroj = 0;
        brojac = 1;
        ponavljaj
                učitaj (x);
                zbroj = zbroj + x;
                brojac = brojac + 1;
        dok vrijedi (brojac < 20)
        ispiši (zbroj);
kraj
```

Zadatak 4

- Napisati program koji će se prekinuti kod unosa broja 5. Potrebno je ispisati zbroj unesenih brojeva:
  a) Broj 5 ulazi u zbroj brojeva
  b) Broj 5 ne ulazi u zbroj brojeva