# Programiranje

kontrolne naredbe
dodatni zadatci za vježbu
Grupa 01, Zdenko Šimić

## Primjeri logičkih izraza uvjetnog operatora

- a, b, c?

```
int a=1, b=-2, c=-1;
a= (c && a) ? c += a : b == a;
```

- Rješenje:

```
a = 0,  b = -2 ,  c = 0
```

- a, c?

```
int a=1, b=-1, c=1;
a= (b<c)>>1 ? c+1 : c-1;
```

```
a = 0   c = 1
```

- Ispis?

```
int a=5, b=-1, c=0;
c = (a=c&&b) ? a = b : ++c;
printf("a= %d, b= %d, c= %d",
          a, b, c);
```

```
a= 0, b= -1, c= 1
```

## ZA VJEŽBU: Proširiti program Fibonacci

- Izračunati omjer dvaju susjednih Fibonaccijevih brojeva, npr. Fibo(7)/Fibo(6) = 21/13 = 1.615384615:
  - Napraviti program tako da se može odrediti broj nakon kojega se omjer ne mijenja, npr. ispi rezultata:

  Omjer Fibo(10) i Fibo(9) = `89/55 = 1,618181818181820`

  Rezultat je zapravo tzv. zlatni rez = `1.6180339887498948482`

- Algoritam za računanje Fibonaccijevih brojeva:
  Fibo(n) = Fibo(n-1) + Fibo(n-2)
  Fibo(0) = Fibo(1) = 1 -> 1, 1, 2, 3, 5, 8, 13, 21,...



## ZA VJEŽBU: Proširiti program tablice množenja

Program koji ispisuje tablicu množenja cijelih brojeva do 100 varirati tako da:

A. Proširiti program tako da ispisuje prvi red i stupac s brojevima od 1 do 10.

B. Promijeniti program tako da ispisuje samo parne rezultate (paziti na ispis)

C. Promijeniti program tako da ispisuje samo neparne rezultate (paziti na ispis)

## ZA VJEŽBU:
## Kreirati drugačiji program srednje vrijednosti

**Napisati program koji izračunava prosjek unaprijed nepoznatog broja pozitivnih cijelih brojeva (brojevi se učitavaju dok se ne unese 0) korištenjem petlje `for`:**

1. **Bez naredbi `break` i `continue`**
2. **Korištenjem naredbi `break` i `continue`**

5

---

## ZA VJEŽBU:
## Kreirati drugačiji program za prost broj

Moguća poboljšanja algoritma ispitivanja je li zadani broj prost smanjivanjem broja ponavljanja petlje:

1. Dovoljno je s petljom ići do n/2,
2. Dovoljno je ispitivati samo do $\sqrt{n}$ (C funkcija `sqrt(n)`)
3. Može se ispitati djeljivost **n** s 2, te ako nije djeljiv s 2, unutar petlje ispitivati djeljivost s neparnim brojevima većim od 2, uz primjenu 1. ili 2. kriterija
4. Za ispis se može koristiti uvjetni operator `( ? : )`

6

## ZA VJEŽBU:
## Kreirati program provjere ulaznih brojeva

Napisati program koji će učitavati realne brojeve s tipkovnice za tjelesnu temperaturu i postupati prema sljedećim pravilima:

1. Unesena temperatura preko 42 °C ispisati poruku "*Kriticno vosoka temperatura!*" i završiti unos.
2. Unesena temperatura preko 43 °C ispisati poruku "*Krivo unesena temperatura!*" i nastaviti unos.
3. Unesena temperatura ispod 35 °C ispisati poruku "*Kriticno niska temperatura!*" i završiti unos.
4. Unesena temperatura ispod 25 °C ispisati poruku "*Krivo unesena temperatura!*" i nastaviti unos.
5. Sve ostale unesene temperature ispisati npr. "*Unesena temperatura: xx.x stupnjeva C*" i nastaviti unos.

7

---

## Domaće zadaće

- 8 -> **4** x 3% = ukupno **12 %**
- Lozinka za 1. DZ **ppi1dz**
- Prva DZ sastoji se od 2 zadatka – kombinacija petlji i `if`-a
- **Nakon** peuzimanja imate **48 h** za predaju rješenja!
- Zadnji rok za preuzimanje je **26.11.05 u 9 h, subota**
- Zadataka je veliki broj, ali ponavljanja su moguća –jako paziti na (ne)prepisivanje i predaju istovjetnih rješenja!
- Nakon predaje zadatci se automatski provjeravaju na testnim primjerima i automatski ocjenjuju
- Pažljivo pročitati zadatak i temeljito provjeriti rad programa prije predaje (ukloniti pomoćne linije i sl.)
- Pitanja: boris.milasinovic@fer.hr ili ivica.botticki@fer.hr

8

## Kontrolne naredbe:

- Naredba **`if`**:
  - Jednostrana
  - Dvostrana
  - Višestrana
- Programske petlje:
  - Ispitivanje uvjeta na početku: **`while`**
  - Ispitivanje uvjeta na kraju: **`do while`**
  - Poznati broj ponavljanja: **`for`**
- Operatori nad bitovima
- Uvjetni operator ( *`?:`* )
- Naredba *break*
- Naredba *continue*
- Naredba *switch*



## Primjeri logičkih izraza

- **d?**

```
short int a=4, b=2, c=8, d;
d = a < 10 && 2 * b < c;
```

- **Rješenje:**

```
d = (a<10) && ((2*b)<c)
d = 1 && (4<8)
d = 1 && 1
d = 1
```

- **Ispis?**

```
int a=5, b=-1, c=0;
if (c=(a=c&&b)) a=b;
else ++c;
printf("a = %d, ", a);
printf("b = %d, ", b);
printf("c = %d \n", c);
```

```
a = 0, b = -1, c = 1
```

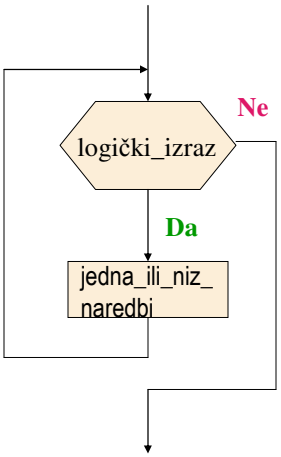
# Programska petlja s ispitivanjem uvjeta ponavljanja na početku

- U C-u

```
while(logički_izraz) {
      jedna_ili_niz_naredbi
}
```

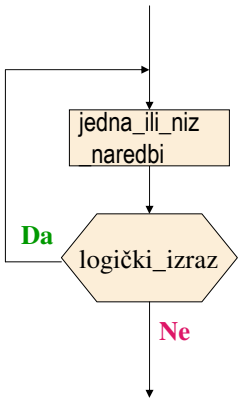
- Ovisno o početnim uvjetima može se dogoditi da se tijelo petlje uopće ne izvršava
- Primjer:

```
int x=-1;
while(1 + x&&1) {
  x++;
  printf("%d", x);
}
```






# Programska petlja s ispitivanjem uvjeta ponavljanja na kraju




- U C-u

```
do {
      jedna_ili_niz_naredbi
 } while (logički_izraz)
```

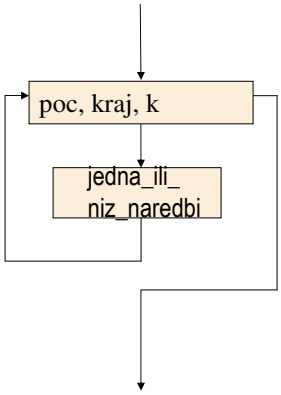
- Tijelo petlje se obavezno izvršava barem jednom
- Primjer:

```
char x=-1;
do {
  x++;
} while(1 + x&&1);
```

# Programska petlja s poznatim brojem ponavljanja

- U C-u

```
for (i = poc; i <= kraj; i = i + k)
   {
      jedna_ili_niz_naredbi
    }
```




- Broj ponavljanja može se unaprijed izračunati neovisno o izvršavanju tijela petlje
- Primjer

```
int i;
for (i = 0; i <= 99; i++)  {
      printf("%d", i);
   }
```



# Programska petlja s poznatim brojem ponavljanja

- Sintaksa:

```
for (izraz1; izraz2; izraz3)
   {
      .      . .
    }
```

- **izraz1**
  - Izvršava se samo jednom prije prvog ponavljanja.
  - Najčešće se koristi za inicijalizaciju brojača.
- **izraz2**
  - Logički izraz riješen prije svakog ponavljanja
  - Ponavljanje se izvršava ako je vrijednost istinita
- **izraz3**
  - Obavlja se nakon svakog ponavljanja.
  - Najčešće se koristi za promjenu vrijednosti brojača.

- Bilo koji od izraza (**izraz1, izraz2 i izraz3**) može se izostaviti.
- Kada je izostavljen **izraz2** to rezultira beskonačnim brojem ponavljanja.
- Ako je potrebno napisati više naredbi u izrazima **1 i 3**, one se odvajaju zarezom.
- Primjer

```
for (i=0, j=60; i>j; i++, j--)
{
    . . .
}
```

```
printf ("%.16e %.16f %.16g \n",y,y,y);
```

```
5.0000000000000000e-001  0.5000000000000000  0.5
2.5000000000000000e-001  0.2500000000000000  0.25
1.2500000000000000e-001  0.1250000000000000  0.125
6.2500000000000000e-002  0.0625000000000000  0.0625
3.1250000000000000e-002  0.0312500000000000  0.03125
1.5625000000000000e-002  0.0156250000000000  0.015625
7.8125000000000000e-003  0.0078125000000000  0.0078125
3.9062500000000000e-003  0.0039062500000000  0.00390625
1.9531250000000000e-003  0.0019531250000000  0.001953125
9.7656250000000000e-004  0.0009765625000000  0.0009765625
4.8828125000000000e-004  0.0004882812500000  0.00048828125
2.4414062500000000e-004  0.0002441406250000  0.000244140625
1.2207031250000000e-004  0.0001220703125000  0.0001220703125
6.1035156250000000e-005  0.0000610351562500  6.103515625e-005
3.0517578125000000e-005  0.0000305175781250  3.0517578125e-005
1.5258789062500000e-005  0.0000152587890625  1.52587890625e-005
```



## Prioritet do sada upoznatih operatora

| | OPERATORI | Pridruživanje |
|---|---|---|
| ↑ Viši prioritet | ( ) | L → D |
| | ! ~ ++ -- sizeof & * unarni + - | D → L |
| | (cast) | D → L |
| | * / % | L → D |
| | + - | L → D |
| | << >> | L → D |
| | < <= > >= | L → D |
| | == != | L → D |
| | & | L → D |
| | ^ | L → D |
| | \| | L → D |
| | && | L → D |
| | \|\| | L → D |
| | ? : | D → L |
| | = *= /= %= += -= &= ^= \|= <<= >>= | D → L |
| Niži prioritet → | , | L → D |