SVEUČILIŠTE U ZAGREBU
Fakultet prometnih znanosti
Zavod za inteligentne transportne sustave
Vukelićeva 4, Zagreb, HRVATSKA

Računalstvo

Implementacija petlji i grananja

Doc. dr. sc. Edouard Ivanjko, dipl.ing.

Sadržaj

- Uvod
- Ugniježđene petlje
- Naredba break
- Naredba continue
- Beskonačna petlja
- Korištenje brojača
- Primjeri

Uvod

- Automatizirana obrada zahtjeva izvođenje više istih operacija za jedan obrađivani podatak
  - Koriste se ugniježđene petlje
- Postoji potreba ranijeg završetka petlje
  - Problem s ulaznim podacima ili krivi unos
  - Implementirana beskonačna petlja
    - Uvjet izvođenja je uvijek istina (true)
- Postoji potreba za preskakanjem dijela ili cijele iteracije za neki podatak
  - Dio obrade nije potreban za pripadni podatak
  - Nedostaje dio podataka

Ugniježđene petlje

- Radi se o petlji unutar petlje
  - Unutarnja petlja je ugniježđena petlja
    - Unutarnjoj petlji se može dodati unutarnja petlja
  - Sve iteracije unutarnje petlje se izvedu u samo jednoj iteraciji vanjske petlje
    - Nakon toga se za svaku novu iteraciju vanjske petlje ponovno izvedu sve iteracije unutarnje petlje
    - Vanjska petlja se poziva jednom, a unutarnje petlje više puta ovisno o broju iteracija vanjske petlje
- Moguće kombinacije svih triju vrsti petlji
  - Potrebno osigurati za svaku petlju uvjet završetka da ne dođe do beskonačne petlje

Ugniježđene petlje

- Pseudokôd

```
vanjska petlja
    naredbe vanjske petlje
    unutarnja petlja
        naredbe unutarnje petlje
```

- Ukupan broj iteracija
  - brojIteracijaVanjske * brojIteracijaUnutarnje
  - Naredbe vanjske petlje se izvrše manje puta
    - brojIteracijaVanjske
  - Naredbe unutarnje petlje se izvrše više puta
    - brojIteracijaVanjske * brojIteracijaUnutarnje
- Gniježđenje je moguće i u više razina

Ugniježđene petlje - Primjer while

- Ispisati tablicu množenja od 1 do 10

```
// deklaracija i inicijalizacija varijabli
int vanjski = 1, unutarnji = 1;

// ugnjezdene while petlje
while ( vanjski < 11)
{
    while (unutarnji < 11)
    {
        // ispis umnoska
        Console.Write(vanjski * unutarnji + " ");

        // povecanje drugog mnozitelja
        unutarnji++;
    }

    // povecanje prvog mnozitelja
    vanjski++;

    // ponovna inicijalizacija drugog mnozitelja
    unutarnji = 1;

    // ispis novog reda na kraju desetine
    Console.WriteLine();
}
```

UgnjezdenaWhilePetlja

```
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
```

Ugniježđene petlje - Primjer do - while

- Ispisati tablicu množenja od 1 do 10

```
// deklaracija i inicijalizacija varijabli
int vanjski = 1, unutarnji = 1;

// ugnjezdene while petlje
do
{
    do
    {
        // ispis umnoska
        Console.Write(vanjski * unutarnji + " ");

        // povecanje drugog mnozitelja
        unutarnji++;
    } while (unutarnji < 11);

    // povecanje prvog mnozitelja
    vanjski++;

    // ponovna inicijalizacija drugog mnozitelja
    unutarnji = 1;

    // ispis novog reda na kraju desetine
    Console.WriteLine();

} while (vanjski < 11);
```

UgnjezdenaDoWhilePetlja

```
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
```

Ugniježđene petlje - Primjer for

- Ispisati tablicu množenja od 1 do 10

```
static void Main(string[] args)
{
    for (int i = 1; i < 11; i++)
    {
        for (int j = 1; j < 11; j++)
        {
            Console.Write(i * j + " ");
        }
        Console.WriteLine();
    }
    Console.ReadKey();
}
```

```
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
```

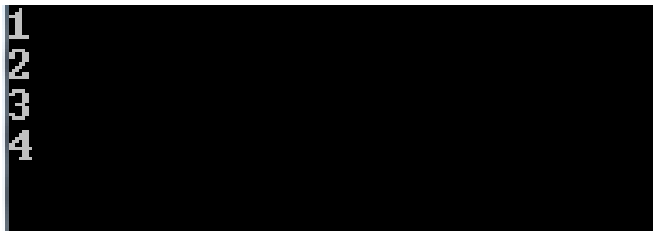
UgnjezdenaForPetlja

Naredba break

- Kontrolna naredba za upravljanje tijekom izvođenja petlje
- Prekida izvođenje petlje i vrši trenutni izlazak iz petlje
  - Izvodi se prva naredba / linija kôda iza programskog bloka petlje
- U slučaju ugniježđene petlje prekida izvođenje najbliže vanjske programske petlje
  - Vanjska petlja u čijem se tijelu nalazi naredba break
- Za ugniježđene petlje vrijedi isto pravilo kao i za naredbu **continue**

Naredba break - Primjer

```
static void Main(string[] args)
{
    for (int i = 1; i <= 100; i++)
    {
        if (i == 5)
        {
            break;
        }
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```



NaredbaBreak

Naredba continue

- Kontrolna naredba za upravljanje tijekom izvođenja petlje
- Prekida izvođenje trenutne iteracije petlje i generira novu iteraciju
  - Trenutna iteracija petlje se neće izvesti
  - Petlja se dalje nastavlja izvoditi u novoj iteraciji
  - U **while** i **do – while** petlji uzrokuje ponovno ispitivanje uvjeta
  - U **for** petlji se prvo izvrši inkrement/dekrement kontrolne varijable pa ispitivanje uvjeta

Naredba continue - Primjer

```
static void Main(string[] args)
{
    for (int i = 1; i <= 10; i++)
    {
        if (i < 9)
            continue;
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```

NaredbaContinue

Beskonačna petlja

- Uvjet za izvođenje petlje uvijek istina (true)
- Tijelo petlje se izvodi beskonačno puta
  - Prekid uzrokuje naredba u tijelu petlje
    - Naredba break
    - Naredba return
      - Vraća rezultat u nadređenu metodu
    - Naredba goto
      - Preusmjerava izvođenje programa na postavljenu labelu
  - Bez naredbe za prekid program ne može nikad završiti s radom
    - Opasnost pojave "plavog ekrana smrti"

Beskonačna petlja

- Prekid izvršavanja programa u slučaju nužde
  - Kombinacija tipki "**CTRL**" + "**C**"
- Beskonačne petlje pogodne za programe u radu 24/7
  - Nikada ne prestaju s radom
    - Mjerenje mase na autocesti
    - Naplata cestarine
    - Upravljanje semaforima
    - Nadgledanje prometnog/industrijskog procesa
    - …

Beskonačna petlja – Primjer while

- Napisati program koji će konstantno ispitivati dozvoljenu masu teretnog vozila na autocesti

🗁 **ProvjeraMaseBeskonacno**

```
// deklaracija varijabli
double najvecaDozvoljenaMasa, masaVozila;

// unos i obrada podataka
Console.Write("Unesite najvecu dopustenu masu teretnog vozila [kg] > ");
najvecaDozvoljenaMasa = Convert.ToDouble(Console.ReadLine());
while (true)
{
    Console.Write("\nUnesite masu teretnog vozila [kg] > ");
    masaVozila = Convert.ToDouble(Console.ReadLine());

    if (masaVozila > najvecaDozvoljenaMasa)
        Console.WriteLine("UPOZORENJE! Masa teretnog vozila veca od dopustene!");
    else
        Console.WriteLine("Masa teretnog vozila u granici dopustene mase.");
}
```

Beskonačna petlja – Primjer do - while

- Napisati program koji će konstantno ispitivati dozvoljenu masu teretnog vozila na autocesti

📂 **ProvjeraMaseBeskonacnoDoWhile**

```
// deklaracija varijabli
double najvecaDozvoljenaMasa, masaVozila;

// unos i obrada podataka
Console.Write("Unesite najvecu dopustenu masu teretnog vozila [kg] > ");
najvecaDozvoljenaMasa = Convert.ToDouble(Console.ReadLine());
do
{
    Console.Write("\nUnesite masu teretnog vozila [kg] > ");
    masaVozila = Convert.ToDouble(Console.ReadLine());

    if (masaVozila > najvecaDozvoljenaMasa)
        Console.WriteLine("UPOZORENJE! Masa teretnog vozila veca od dopustene!");
    else
        Console.WriteLine("Masa teretnog vozila u granici dopustene mase.");
} while (true);
```

Beskonačna petlja – Primjer for

- Napisati program koji će konstantno ispitivati dozvoljenu masu teretnog vozila na autocesti

ProvjeraMaseBeskonacnoFor

```
// deklaracija varijabli
double najvecaDozvoljenaMasa, masaVozila;

// unos i obrada podataka
Console.Write("Unesite najvecu dopustenu masu teretnog vozila [kg] > ");
najvecaDozvoljenaMasa = Convert.ToDouble(Console.ReadLine());
for(;true;)
{
    Console.Write("\nUnesite masu teretnog vozila [kg] > ");
    masaVozila = Convert.ToDouble(Console.ReadLine());

    if (masaVozila > najvecaDozvoljenaMasa)
        Console.WriteLine("UPOZORENJE! Masa teretnog vozila veca od dopustene!");
    else
        Console.WriteLine("Masa teretnog vozila u granici dopustene mase.");
}
```

Korištenje brojača

- Postoji potreba za praćenjem broja nekog podatka
  - Napravljene iteracije petlje
  - Broj prikupljenih unosa
  - Broj obrađenih podataka
- U tu svrhu se koriste brojači
  - Varijable koje se inkrementiraju ili dekrementiraju kada je stavka obrađena
    - Kod **for** petlje je inkrement/dekrement kontrolne varijable ugrađen
      - Broje se odrađene iteracije radi uvjeta završetka petlje

Korištenje brojača - Primjer

- Izraditi program koji izračunava prosjek unesenih ocjena, prekid unosa izvodi se unosom znamenke 0 umjesto ocjene

📂 **ProsjekOcjena**

```
static void Main(string[] args)
{
    Console.WriteLine("IZRACUN PROSJEKA OCJENA: Unesi ocjenu (0 za kraj)");

    double unos = Convert.ToDouble(Console.ReadLine());
    double brojac = 1, suma = unos;

    while (unos != 0)
    {
        Console.WriteLine("Unesi ocjenu ili 0 za kraj");
        unos = Convert.ToDouble(Console.ReadLine());

        suma = suma + unos;
        brojac = brojac + 1;
    }

    brojac = brojac - 1;

    Console.WriteLine("Prosjecna ocjena je: " + suma / brojac);
    Console.ReadKey();
}
```

Korištenje brojača - Primjer

- Izračunati aritmetičku sredinu brojeva koji se čitaju s tipkovnice dok njihova suma ne premaši zadanu gornju granicu

📂 **AritmetickaSredinaGornjaGranica**

```
// deklaracija varijabli
int gornjaGranica;
int suma, brojPodataka;
double aritmetickaSredina;

// inicijalizacija varijabli
suma = 0;
brojPodataka = 0;

// ucitavanje podataka
System.Console.Write("Unesite gornju granicu > ");
gornjaGranica = Convert.ToInt32(Console.ReadLine());
System.Console.Write("\n");
do
{
    System.Console.Write("Unesite broj > ");
    suma = suma + Convert.ToInt32(Console.ReadLine());
    brojPodataka++;
} while (suma < gornjaGranica);

// izracun aritmeticke sredine
aritmetickaSredina = (double) suma / brojPodataka;
Console.WriteLine("Aritmeticka sredina ucitanih brojeva je " + aritmetickaSredina);
```

Korištenje brojača - Primjer

📂 **BrojanjeKlaseVozila**

```
// deklaracija varijabli
double masaVozila;
int neispravnaMasa, motocikl, kombiniraniAutomobil, teretnoVozilo;

// inicijalizacija varijabli
neispravnaMasa = 0;
motocikl = 0;
kombiniraniAutomobil = 0;
teretnoVozilo = 0;

// unos i obrada podataka
do {
    Console.Write("\nUnesite masu vozila u [kg] (0 za kraj) > ");
    masaVozila = Convert.ToDouble(Console.ReadLine());
    // kraj petlje
    if (masaVozila == 0.0)
        break;
    // neispravna masa
    else if (masaVozila < 0.0)
        neispravnaMasa++;
    else if (masaVozila > 0.0 && masaVozila <= 400.0)
        motocikl++;
    else if (masaVozila > 400.0 && masaVozila <= 3500.0)
        kombiniraniAutomobil++;
    else
        teretnoVozilo++;
} while (true);

// ispis rezultata
Console.WriteLine("\nBroj motocikala je " + motocikl);
Console.WriteLine("Broj kombiniranih automobila je " + kombiniraniAutomobil);
Console.WriteLine("Broj teretnih vozila je " + teretnoVozilo);
Console.WriteLine("Broj neispravno unesenih masa je " + neispravnaMasa);
```

- Napisati program koji učitava masu vozila i broji njihov broj po klasama
  - Učitavanje prestaje uz unos mase od 0 kg, a potrebno je brojati i krivo zadane mase (< 0 kg)