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

Računalstvo

Unos i ispis podataka

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

Sadržaj

- Uvod
- Pretvorba formata podataka
- Razmjena podataka preko konzole
- Unos podataka
- Ispis podataka
- Primjeri

Uvod

- Prilikom izvođenja programa obrađuju se podaci
  - Brojevi, tekst i logičke vrijednosti s dogovorenim značenjem
- Potrebna izmjena podataka između programa i njegove okoline
  - Operater
  - Druga računala
  - Drugi programi ili dijelovi programa
- Potrebno poštivanje formata podataka
  - Računala i ljudi govore različiti jezik

Uvod

- Smjer izmjene podataka
  - Od okoline u program -> **unos**
    - Operater unosi podatke
    - Program čita podatke iz datoteke na disku
    - Program prima podatke od drugog programa
  - Od programa u okolinu -> **ispis**
    - Program prikazuje rezultat na zaslonu
    - Program kreira datoteku na disku
    - Program šalje podatke drugom programu

Pretvorba formata podataka

- Svi podaci u računalu prikazani u binarnom formatu
  - Skup "**0**" i "**1**" udruženih u bytove
  - Radi lakšeg čitanja koristi se heksadecimalni prikaz
    - 4 binarne znamenke -> 1 heksadecimalna znamenka
- Tip podatka definira značenje pojedinog bita unutar byta
  - Npr. $03B1_{16} = 0000\ 0011\ 1011\ 0001_2$
    - Tip **char** daje znak '**α**'
    - Tip **short** daje vrijednost **945**

Pretvorba formata podataka

- Razmjena podataka između programa i računala koristi definirane formate
  - Brojevi se razmjenjuju pomoću numeričkih tipova podataka (**int**, **long**, **float**, **double**, itd.)
  - Tekst se razmjenjuje pomoću niza znakova (**string**)
- Kod razmjene podataka s operaterom koriste se nizovi znakova (**string**)
  - Izravno podržan tekst
  - Numeričke vrijednosti zahtijevaju pretvorbu
    - Tekst (znamenke) u cijeli broj
    - Tekst (znamenke i decimalna točka) u broj s plivajućim zarezom

Pretvorba formata podataka

- Tipovi pretvorbe (konverzije) podataka
  - Implicitne
    - Pretvorba bez intervencije
    - Prevodilac automatski prepoznaje potrebu
  - Eksplicitne
    - Koriste se kada postoji mogućnost gubitka podataka ili pogreške
      - Pretvorba iz formata većeg opsega u format manjeg opsega -> npr. double => float
    - Pretvorba pomoću ukalupljivanja (engl. "cast operator")

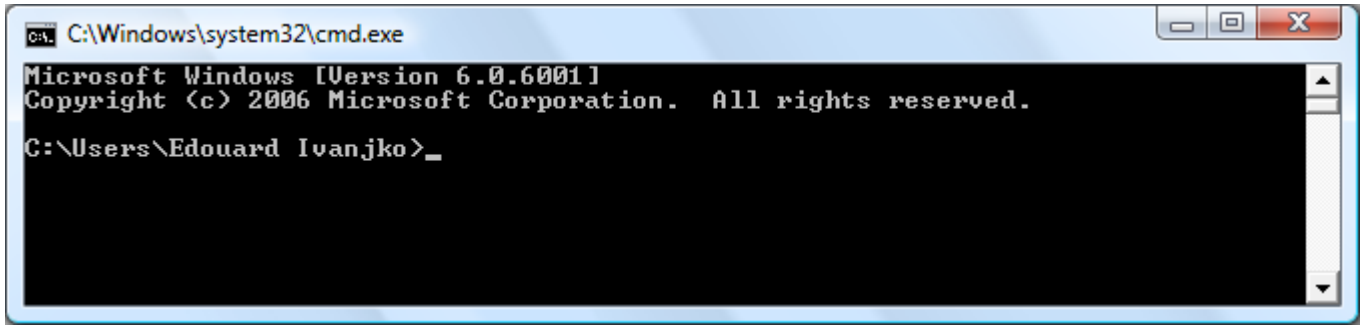
```
double najvecaBrzina = 120.0;
float gornjaBrzina = (float) najvecaBrzina;
```

Pretvorba formata podataka

- Dozvoljene implicitne pretvorbe podataka
  - **char** -> ushort, int, uint, long, ulong, float, double, decimal
  - **short** -> int, long, float, double, decimal
  - **ushort** -> int, uint, long, ulong, float, double, decimal
  - **int** -> long, float, double, decimal
  - **uint** -> long, ulong, float, double, decimal
  - **long** -> float, double decimal
  - **ulong** -> float, double, decimal
  - **float** -> double
  - Osnovna značajka da nema opasnosti od gubitka podataka
    - Format manjeg opsega se sprema u format većeg opsega

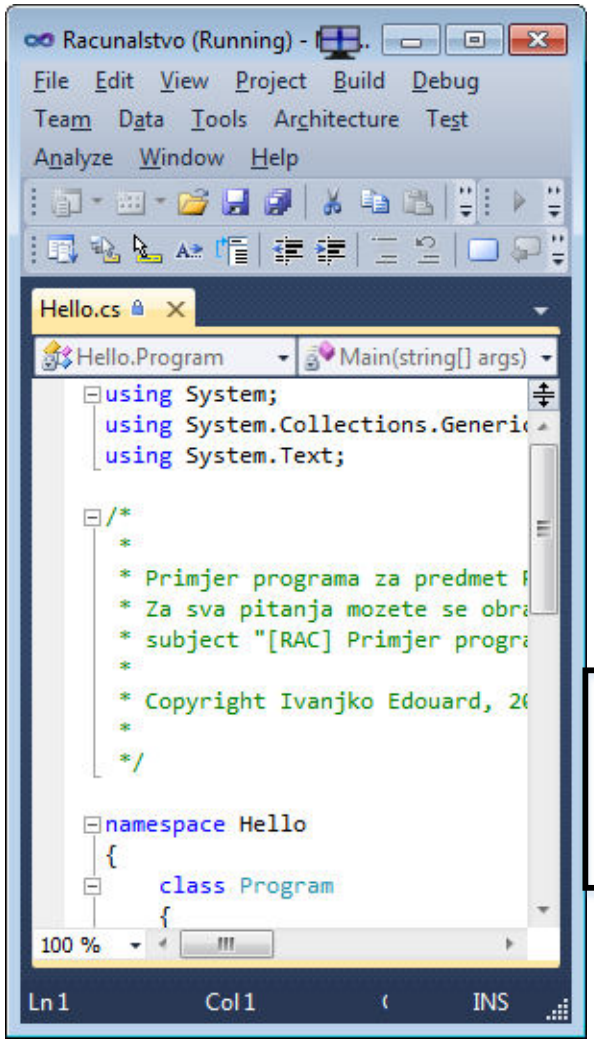
Razmjena podataka preko konzole

- Konzola je tekstualno sučelje programa
- Razmjena podataka bez mogućnosti korištenja grafičkih objekata
  - Samo znakovi (znamenke, slova i interpunkcijski znakovi)
- Dvosmjerna razmjena podataka
  - Omogućen unos i ispis podataka



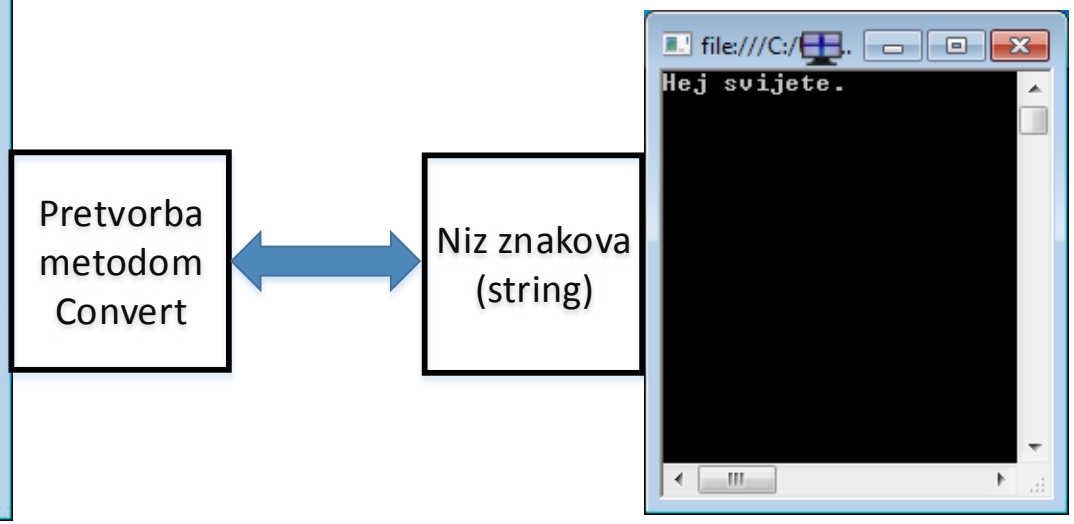
Razmjena podataka preko konzole

C# program



- Konzola kao tekstualno sučelje prihvaća samo nizove znakova
  - I od operatera i od programa
  - Tip podatka **string**

Konzola



Razmjena podataka preko konzole

- Pristup konzoli pomoću metode "**Console**" iz imeničkog prostora "**System**"
  - Pristup pojedinoj metodi ide pomoću operatora točke
  - Bez uključenog imeničkog prostora System

    ```
    System.Console.ImeMetode();
    ```
  - Uz uključen imenički prostor System

    ```
    Console.ImeMetode();
    ```
- Razmjena podataka omogućuje kreiranje interaktivnog programa
  - Izvršavanje ovisno o unosu operatera

Razmjena podataka preko konzole

- Metoda za pretvorbu podataka "**Convert**"
  - Bez uključenog imeničkog prostora System

    ```
    rezultat = System.Convert.ImeMetodePretvorbe(podatak);
    ```

  - Uz uključen imenički prostor System

    ```
    rezultat = Convert.ImeMetodePretvorbe(podatak);
    ```

- Podatak za pretvorbu može biti bilo koji drugog tipa
  - Cijeli broj, niz znakova, …
- Koristi se kod unosa podataka ili kao zamjena za ukalupljivanje

Razmjena podataka preko konzole

- Metoda "**Convert**" podržava pretvorbe
  - **ToBoolean** => u logičku varijablu (**bool**)
  - **ToInt16** => u cijeli broj duljine 2 byte (**short**)
  - **ToInt32** => u cijeli broj duljine 4 byte (**int**)
  - **ToInt64** => u cijeli broj duljine 8 byte (**long**)
  - **ToDouble** => u broj s plivajućim zarezom (**double**)
  - **ToDecimal** => u broj s plivajućim zarezom većeg opsega (**decimal**)
  - **ToChar** => u znak (**char**)
  - **ToString** => u niz znakova (**string**)
- Ulazni tip podatka se automatski prepoznaje

Razmjena podataka preko konzole

- Primjer pretvorbe podataka

```
// deklacija i inicijalizacija varijabli
bool logicka;
int cijeliBroj = 100;
long dugiCijeliBroj;
float plivajuciZarez;
double dugiPlivajuciZarez = 1000.0;
char znak;
string nizZnakova;

// implicitna pretvorba
dugiCijeliBroj = cijeliBroj;
dugiCijeliBroj = Convert.ToInt64(cijeliBroj);

// eksplicitna pretvorba
// slovo 'd' prema Unicode tablici
znak = (char) cijeliBroj;
// slovo 'd' prema Unicode tablici
znak = Convert.ToChar(cijeliBroj);
// niz znakova "100"
nizZnakova = Convert.ToString(cijeliBroj);
// true za broj > 0 i < 0, false kada broj == 0
logicka = Convert.ToBoolean(cijeliBroj);
// vrijednost 1000.0 uz manje zauzece memorije
plivajuciZarez = (float) dugiPlivajuciZarez;
```

PretvorbaPodataka

Unos podataka

- Smjer podataka od operatera u program
- Unos podataka preko konzole
- Imenički prostor "**System**"
- Koriste se metode
  - Read
    - Čita znak po znak dok operater ne pritisne Enter
    - Znak završetka (Enter) će također biti pročitan
  - ReadLine
    - Čeka da operater otipka ulazne podatke i pritisne Enter
    - Metoda generira znak za novi redak da se kursor pomakne
    - Vraća niz znakova osim zadnjeg znaka za novi redak (Enter)

Unos podataka – Metoda Read

- Vraća vrijednost tipa **int**
  - Potrebna pretvorba u tip **char**
  - Koristi se metoda **Convert**

```
char c;
c = Convert.ToChar(Console.Read());
```

- Znakovi kôdirani Unicode tablicom
- S konzole čita samo jedan znak
- Čitanje počinje nakon završetka unosa
- Za čitanje više znakova potrebno je ponavljati operaciju

Unos podataka – Metoda ReadLine

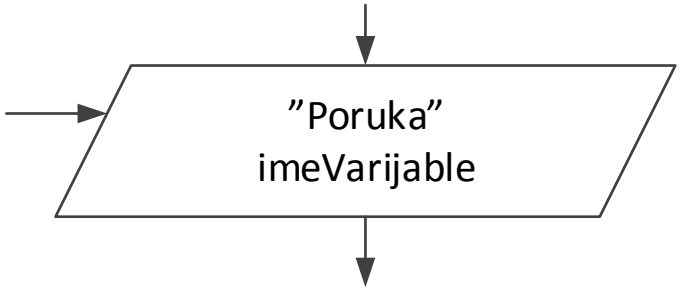
- Vraća vrijednost tipa **string**
  - Potrebna pretvorba ako je željeni tip podatka različit od niza znakova
    - Numerička ili logička vrijednost
  - Koristi se metoda **Convert**
    - Potrebno poštivati željeni tip podatka kod unosa

```
string unos;
int cijeliBroj;
unos = Console.ReadLine();
cijeliBroj = Convert.ToInt32(unos);
```

- Niz znakova ne sadrži znak za kraj unosa
  - Enter ili Return

Unos podataka – Poveznica s dijagramom toka

- Unos uz poruku u dijagramu toka



- Pripadni C# kôd

```
int imeVarijable;
string unos;
Console.Write(”Poruka”);
unos = Console.ReadLine();
imeVarijable = Convert.ToInt32(unos);
```

  - Ili skraćeno

```
imeVarijable = Convert.ToInt32(Console.ReadLine());
```

Unos podataka – Primjer

```
// koristenje metode Read
// deklaracija varijabli
char c, d, e;
int i;
// citanje znaka
i = Console.Read();
c = Convert.ToChar(i);
// citanje znaka za kraj
i = Console.Read();
d = Convert.ToChar(i);
// citanje znaka za novi redak
i = Console.Read();
e = Convert.ToChar(i);

// koristenje metode ReadLine
// deklaracija varijabli
string unos;
int cijeliBroj;
long dugiCijeliBroj;
double plivajuciZarez;
// citanje niza znakova
unos = Console.ReadLine();
// pretvorba u zeljeni tip podatka
cijeliBroj = Convert.ToInt32(unos);
dugiCijeliBroj = Convert.ToInt64(unos);
plivajuciZarez = Convert.ToDouble(unos);
```

- Mogućnost povezivanja pojedinih naredbi

```
c = Convert.ToChar(Console.Read());
```

```
cijeliBroj =
Convert.ToInt32(Console.ReadLine());
```

📂 **UnosPodataka**

Ispis podataka

- Smjer podataka od programa prema operateru
- Ispis podataka na konzolu
- Imenički prostor "**System**"
- Koriste se metode
  - Write
    - Ispisuje proslijeđeni niz znakova bez odlaska u novi redak
  - WriteLine
    - Ispisuje proslijeđeni niz znakova uz odlazak u novi redak

Ispis podataka – Metoda Write

- Ispis niza znakova bez automatskog ispisa znaka za novi redak
- Potrebna pretvorba numeričke i logičke vrijednosti u niz znakova
  - Svaka varijabla ima pomoćnu metodu **ToString** za potrebnu pretvorbu

```
int cijeliBroj = 10;
string nizZnakova;
nizZnakova = cijeliBroj.ToString();
```

- Niz znakova (poruka) za ispis se stavlja u dvostruke navodnike

```
Console.Write("Niz znakova za ispis!");
```

Ispis podataka – Metoda Write

- Za ispis poruke i vrijednosti varijabli koristi se operator "+"
  - Spaja više znakovnih nizova u jedan

```
int cijeliBroj = 10;
Console.Write("Vrijednost iznosi " + cijeliBroj.ToString());
```

    - Ispis na konzoli: **Vrijednost iznosi 10_**
  - Današnja inačica C# automatski radi pretvorbu numeričke i logičke vrijednosti u niz znakova

```
int cijeliBroj = 10;
Console.Write("Vrijednost iznosi " + cijeliBroj);
```

    - Ispis na konzoli: **Vrijednost iznosi 10_**

Ispis podataka – Metoda Write

- Moguće i definiranje formata ispisa
  - Broja decimala, razmaka, postotka, novčane jedinice i sl.
  - Specijalni znakovi za ispis
    - "**\r**" -> Return ili Enter
    - "**\n**" -> novi redak
    - "**\t**" -> tab
    - "\#" -> znak #
    - "\"" -> znak "
    - "\\" -> znak \

Ispis podataka – Metoda Write

- Detalji o formatu ispisa se ubacuju unutar vitičastih zagrada
  - Format je

    {redniBroj,pomak:brojCijelihMjesta.brojDecimalnihMjesta}

    - "**redniBroj**" označava varijablu čija se vrijednost želi ispisati
    - "**pomak**" označava broj znakova od zadnje ispisanog znaka u smislu pomaka početka
      - Može biti pozitivan (desno poravnanje) ili negativan (lijevo poravnanje)
      - Zanemaruje se ako je pomak veći od duljine teksta za ispis
    - "**brojCijelihMjesta**" označava broj znakova namijenjen ispisu znamenki lijevo od decimalnog zareza (točka u C#)
    - "**brojDecimalnihMjesta**" označava broj znakova namijenjen ispisu znamenki desno od decimalnog zareza (točka u C#)

Ispis podataka – Metoda Write

- Parametri **brojCijelihMjesta** i **brojDecimalnihMjesta** zadaju se simbolima
  - „**0**" označava jednu znamenku
    - Ako znamenke nema u podatku ispiše se "**0**"
  - „**#**" označava jednu znamenku
    - Ako znamenke nema u podatku ništa se ne ispisuje
  - Npr. ispis broja PI na tri decimale

    Console.WriteLine("Broj PI: {0,0:00.000}", Math.PI);
    - Rezultira ispisom **03.142**

    Console.WriteLine("Broj PI: {0,0:##.###}", Math.PI);
    - Rezultira ispisom **3.142**

Ispis podataka – Metoda Write

- Parametri **brojCijelihMjesta** i **brojDecimalnihMjesta** mogu se zamijeniti već predefiniranim oznakama
  - **P** -> ispis postotka
  - **C** -> ispis oznake novčane jedinice
  - **X** -> heksadecimalni format
  - **D** -> ispis cijelog broja s predznakom
  - **E** -> ispis broja s potencijom uz 6 decimala
  - **N** -> općeniti ispis broja s grupiranjem tisućica i drugih viših potencija te decimala
  - **F** -> ispis datuma i vremena

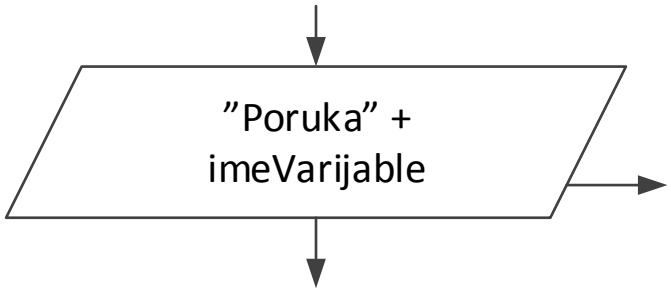
📂 **RacunPDV**

Ispis podataka – Metoda WriteLine

- Ispis niza znakova uz automatski ispis znaka za novi redak
- Po ostalim pravilima analogna metodi "**Write**"
  - Potrebna pretvorba numeričke i logičke vrijednosti u niz znakova
  - Niz znakova (poruka) za ispis se stavlja u dvostruke navodnike
  - Mogućnost podešavanja formata ispisa

Ispis podataka – Poveznica s dijagramom toka

- Ispis uz poruku u dijagramu toka



- Pripadni C# kôd

```
int imeVarijable;
imeVarijable = 1;
Console.Write(”Poruka” + imeVarijable);
```

  - Ili uz odlazak kursora u novi redak

```
Console.WriteLine(”Poruka” + imeVarijable);
```

Primjeri – Unos teksta i ispis

- Varijable **unosPrvi** i **unosDrugi** su tipa **string** pa nije potrebna pretvorba
- Novi redak se može ispisati specijalnim znakom "**\n**" ili metodom **WriteLine**

```
// deklaracija varijabli
string unosPrvi, unosDrugi;

// unos teksta
Console.Write("Unesite prvi zeljeni tekst > ");
unosPrvi = Console.ReadLine();
Console.Write("Unesite drugi zeljeni tekst > ");
unosDrugi = Console.ReadLine();

// ispis dva nova retka radi preglednosti
Console.Write("\n");
Console.WriteLine();

// ispis pomocu metode Write
Console.Write("Metoda Write daje sljedeci ispis:\n");
Console.Write(unosPrvi);
Console.Write(unosDrugi);
Console.Write("\n");

// ispis dva nova retka radi preglednosti
Console.Write("\n");
Console.WriteLine();

// ispis pomocu metode WriteLine
Console.WriteLine("Metoda WriteLine daje sljedeci ispis:");
Console.WriteLine(unosPrvi);
Console.WriteLine(unosDrugi);
```

📂 **IspisPodataka**

Primjeri – Formati ispisa

```
// deklaracija varijabli
double cijenaPrva, cijenaDruga, postotak, dobit;

// unos podataka
Console.Write("Unesite kupovnu cijenu dionice  > ");
cijenaPrva = Convert.ToDouble( Console.ReadLine() );
Console.Write("Unesite prodajnu cijenu dionice > ");
cijenaDruga = Convert.ToDouble(Console.ReadLine());

// izracun podataka
dobit = cijenaDruga - cijenaPrva;
postotak = dobit / cijenaPrva * 100.0;

// ispis podataka bez poravnanja
Console.WriteLine("\n");
Console.WriteLine("Kupovna cijena je bila " + cijenaPrva + " HRK");
Console.WriteLine("Prodajna cijena je bila " + cijenaDruga + " HRK");
Console.WriteLine("Dobit iznosi " + dobit + " HRK");
Console.WriteLine("U postocima je dobit " + postotak + " %");

// ispis podataka uz poravnanje
Console.WriteLine("\n");
Console.WriteLine("Kupovna cijena je bila\t{0,15:c}", cijenaPrva);
Console.WriteLine("Prodajna cijena je bila\t{0,15:c}", cijenaDruga);
Console.WriteLine("Dobit iznosi\t\t{0,15:c} odnosno {1,8:p2}", dobit, postotak/100);

// da se vidi ispis konzole
Console.ReadLine();
```

- Format ispisa “p2” označava ispis postotka na dvije decimale

📁 **CijeneDionica**

Primjeri – Formati ispisa

- Ispis bez formata ispisuje potreban broj decimala
  - 0 ako je broj cijeli
- Ispis u heksadecimalnom formatu je za cijele brojeve

PrikazUdaljenosti

```
// deklaracija varijabli
double udaljenost;

// unos podataka
Console.Write("Unesite udaljenost u [km] > ");
udaljenost = Convert.ToDouble(Console.ReadLine());

// ispis podataka
Console.WriteLine("Udaljenost u metrima iznosi " + udaljenost * 1000);
Console.WriteLine("Udaljenost u metrima iznosi {0,0:e}", udaljenost * 1000);
Console.WriteLine("Udaljenost u metrima iznosi {0,0:n}", udaljenost * 1000);
Console.WriteLine("Udaljenost u metrima iznosi {0,0:X}", (long) udaljenost * 1000);

// da se vidi ispis konzole
Console.ReadLine();
```