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

Računalstvo

Prikaz podataka

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

Sadržaj

- Uvod
- Varijable
- Konstante
- Vrste podataka
- Pohrana podataka

Uvod

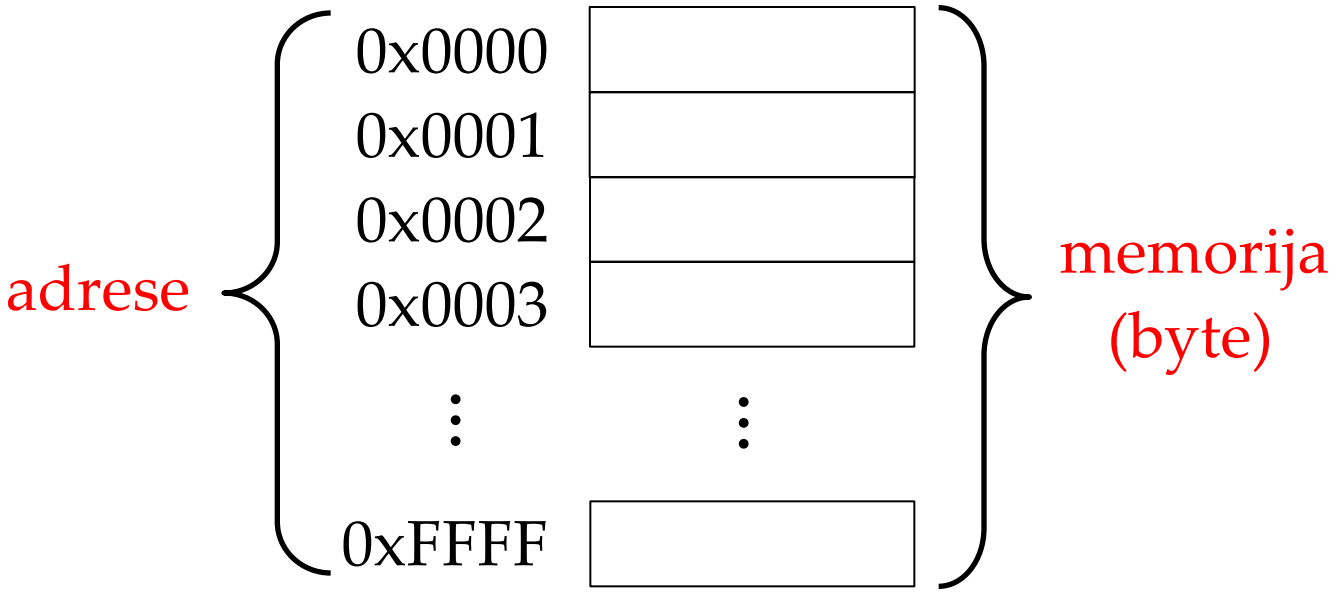
- Podaci - fizički opisi pojmova, ideja, apstrakcija
- Podaci služe
  - Prijenosu informacija
  - Pohrani informacija za buduće upotrebe
  - Izvođenju novih informacija tijekom obrade podataka
- Informacije - značenja pridružena podacima
  - Služe kao podrška u procesima odlučivanja i upravljanja

Uvod – Organizacija memorije

- Najmanji dio memorije se naziva **bit**
  - Može spremiti jednu binarnu znamenku
    - **0** ili **1**
- Skupina od 8 bitova se naziva **byte**
  - Najmanji adresabilni dio memorije
- Byte se koristi u organizaciji memorije
  - Može prikazati numerički raspon od **0** do **255**
- Svi podaci kôdiraju se binarnim brojevima
  - Baza prikaza je **2** i vrijedi (**n** je broj bitova)
    - Pozitivni brojevi prikaz od **0** do $\mathbf{2^n-1}$
    - Negativni i pozitivni brojevi prikaz od $\mathbf{-2^{n-1}}$ do $\mathbf{2^{n-1}-1}$

Uvod – Organizacija memorije

- Svaki byte u memoriji ima svoju adresu
  - Kao adresa podatka uzima se adresa prvog byte kako ih je potrebno više od jednog za spremanje podatka
  - Adrese su u heksadecimalnom formatu



Varijable

- Veličine koja poprimaju vrijednosti iz skupa dopuštenih vrijednosti
- Vrijednost varijable tijekom izvršavanja programa određuje algoritam
- Deklaracija i inicijalizacija varijable u C#

tip imeVarijable;

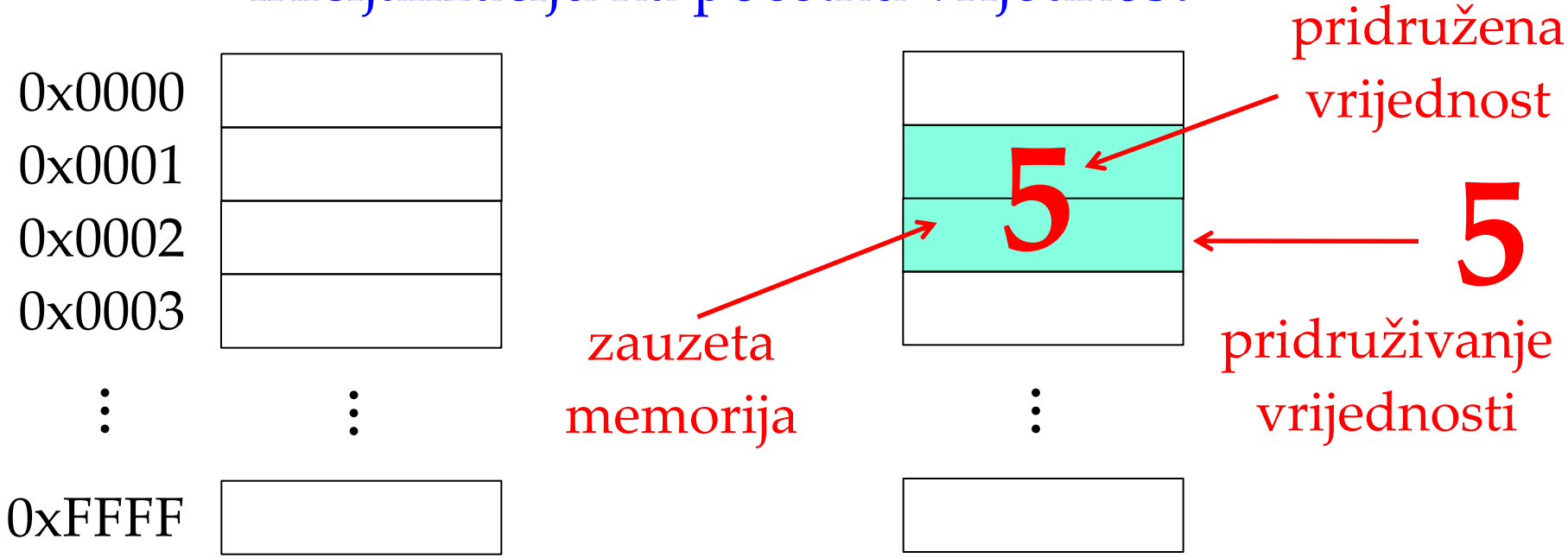
imeVarijable = vrijednost;

  - Vrijednost može biti numerička, logička, znak ili niz znakova
  - U istom retku moguće deklarirati i inicijalizirati varijablu

tip imeVarijable = vrijednost;

Varijable

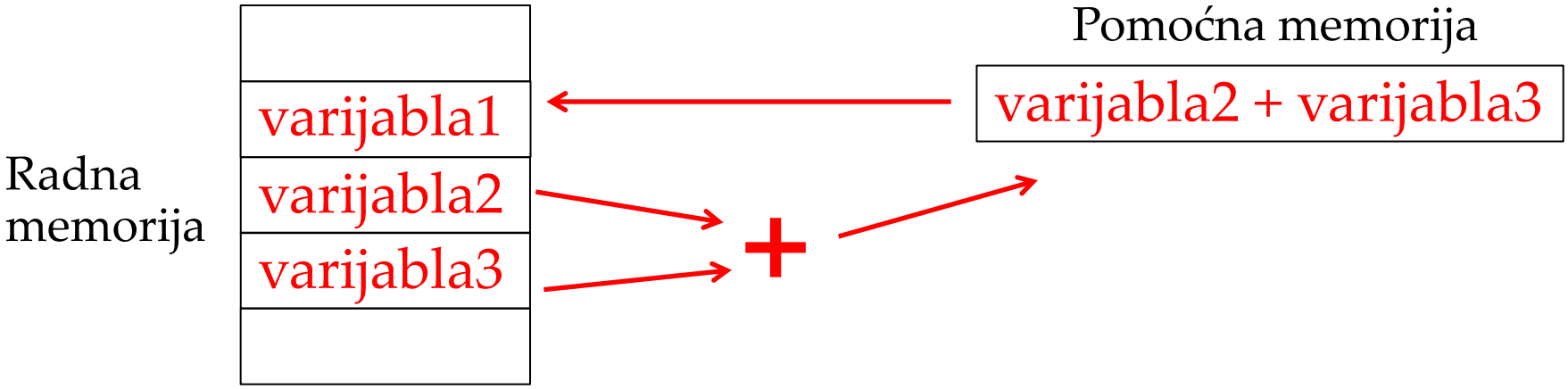
- Deklaracija varijable zauzima potreban prostor u memoriji
- Početna vrijednost 0 ili nepoznata
  - Uputno je prije korištenja varijable napraviti inicijalizaciju na početnu vrijednost



Varijable – Operator "="

- Operator pridruživanja
- Varijabli s lijeve strane znaka pridruživanja pridružuje se vrijednost izraza s desne strane znaka pridruživanja
  - Radi se o spremanju vrijednosti u memoriju pridruženoj varijabli s lijeve strane izraza

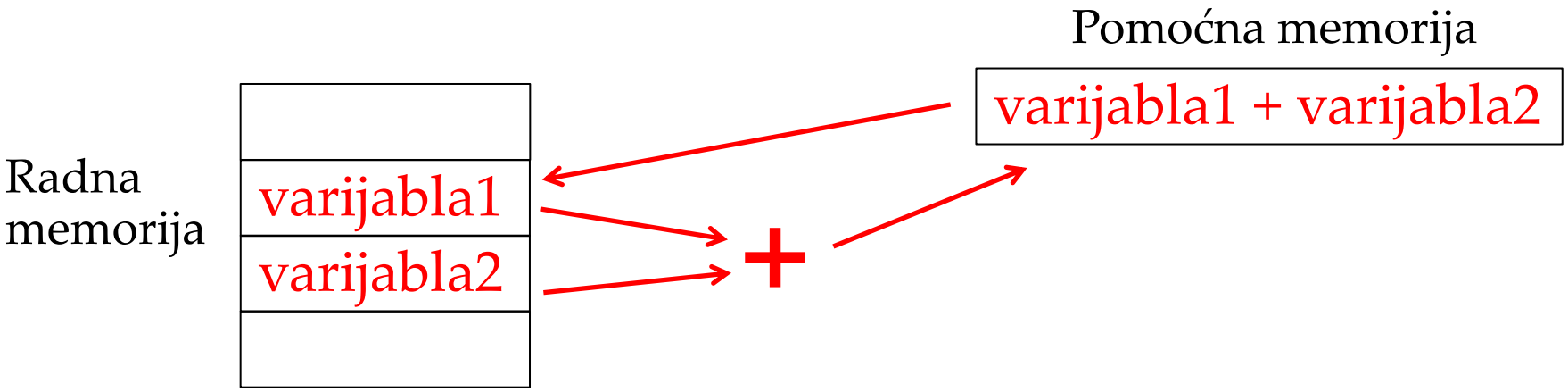
```
varijabla1 = varijabla2 + varijabla3;
```



Varijable – Operator "="

- Vrijednost varijable se mijenja pomoću operatora pridruživanja
  - Bez pridruživanja nove vrijednosti nema promjene

    varijabla1 = varijabla1 + varijabla2;
    - Vrijednost varijable **varijabla1** se mijenja
    - Prvo se uzima trenutna vrijednost varijable **varijabla1** i koristi se na desnoj strani da bi se rezultat izraza s desne strane spremio u varijablu **varijabla1**



Varijable – Vrste

- Podjela prema dostupnosti varijabli
  - Globalne varijable
  - Lokalne varijable
- Globalne varijable
  - Varijable dostupne iz bilo kojeg dijela programa
    - Glavne i pomoćnih metoda
  - Nedostaci
    - Mnoštvo globalnih varijabli usložnjava program
    - Složeno ispravljanje programa
    - Otežano razumijevanje programa
    - Kôd nije izravno prenosiv

Varijable – Vrste

- Lokalne varijable
  - Moguće im je pristupiti unutar metode u kojoj su definirane
    - Granice definiraju vitičaste zagrade s kôdom metode
  - Izvan metode lokalne varijable ne postoje
- Varijable koje imaju isto područje vidljivosti moraju imati različita imena
  - Područje vidljivost ograničava se primjenom imeničkih prostora
    - Direktiva "namespace"

Konstante

- Konstanta je vrsta "varijable" koja se nakon deklaracije samo jednom inicijalizira
- Tijekom programa više ne mijenja svoju vrijednost
  - Izmjena vrijednosti nije dozvoljena
- Deklaracija/inicijalizacija konstante

```
const float pi = 3.14;
```

  - U istoj naredbi se konstanta deklarira i pridijeli joj se vrijednost

Konstante – Predefinirane konstante

- Osnovne konstante već predefinirane u C#
- Sadržane u osnovnom imeničkom prostoru "System"
  - Konstanta **pi** (Arhimedov ili Ludolfov broj -> 3,14)

    System.Math.PI
  - Konstanta **e** (Eulerov broj ili Napierova konstanta -> 2,71)

    System.Math.E
  - Primjer

```
static float radius = 2.0;
static float opseg;
opseg = 2.0 * radius * System.Math.PI;
```

Vrste podataka

- Vrste podataka
  - Numerički (brojevi)
  - Logički podaci (Boole-ova algebra)
  - Tekst (slova, znamenke, interpunkcijski znakovi)
    - Prikaz samo jednog znaka
    - Prikaz niza znakova
- Svaka vrsta podatka ima svoj pripadni tip u C# za spremanje u memoriju
- Miješanje nije dozvoljeno
  - Znak nije moguće spremiti u tip podatka za spremanje brojeva i obrnuto
  - Bitna razrada koncepta programa radi definiranja vrste podataka

Pohrana podataka – Osnovni tipovi

- Pohrana brojeva u memoriji računala
  - Cijeli brojevi
  - Brojevi s plivajućim zarezom

Tip	Pohrana	Najmanja vrijednost	Najveća vrijednost
sbyte	8 bita	-128	127
byte	8 bita	0	255
short	16 bita	-32,768	32,767
ushort	16 bita	0	65,535
int	32 bita	-2,147,483,648	2,147,483,647
uint	32 bita	0	4,294,967,295
long	64 bita	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
ulong	64 bita	0	18,446,744,073,709,551,615
float	32 bita	$1.5 \cdot 10^{-45}$	$3.4 \cdot 10^{38}$
double	64 bita	$5.0 \cdot 10^{-324}$	$1.7 \cdot 10^{308}$
decimal	128 bita	$1.0 \cdot 10^{-28}$	$7.9 \cdot 10^{28}$

Pohrana podataka – Osnovni tipovi

- Primjer deklaracije i inicijalizacije varijabli za spremanje brojeva
  - Cijeli brojevi

```
static int cijeliBroj;
cijeliBroj = 10;
static long veciCijeliBroj = 4000000000000;
```

    - Navodi se cijeli broj bez decimalne točke
  - Brojevi s plivajućim zarezom (C# koristi točku)

```
static float plivajuciBroj;
plivajuciBroj = 0.00;
static double veciPlivajuciBroj = 1.00;
```

    - Bitno je **uvijek** navesti decimalnu točku

Pohrana podataka – Osnovni tipovi

- Prikaz logičkih vrijednosti
  - Tip podatka "**bool**"
  - Moguće dvije vrijednosti
    - Istina ili logička jedinica ili "**true**"
    - Neistina ili logička nula ili "**false**"
  - Primjer

```
bool rezultat;
int a;
a = 4;
rezultat = (a > 64 && a < 123);
Console.WriteLine(rezultat);
```

Pohrana podataka – Osnovni tipovi

- Pohrana znakova
  - Tip podatka "**char**"
  - Zauzima **16** bita ili **2** byte
  - Prikaz jednog znaka UNICODE tablice
    - Svaki znak kôdiran pripadnim cijelim brojem
  - Moguća pretvorbe formata znaka u broj
    - char -> int, char -> long, char -> double, …
  - Pretvorba formata broja u znak nije moguća
  - Primjer deklaracije/inicijalizacije char varijable

```
char znak;
znak = 'Z';
```

    - Vrijednost se upisuje unutar **jednostrukih navodnika**

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Riječi, rečenice, poruke
  - Ključna riječ "**string**"

    static **string** nizZnakova;
  - Veličina ovisi o broju pohranjenih znakova
  - UNICODE -> 2 byte po jednom znaku
  - Posebna skupina metoda za obradu varijabli ovog tipa
    - Pretprocesorska direktiva: "**using System.Text;**"
  - Primjer deklaracije/inicijalizacije string varijable

    string znakovi;

    znakovi = "Niz znakova!";

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Operacijski sustav automatski zauzima memoriju prema duljini znakovnog niza
  - Atribut "**Length**" vraća duljinu niza znakova

```
string s = "12345abc";
int duljina;
duljina = s.Length;
```

    - Inicijalizacija varijable tipa **string** radi se pomoću upisa vrijednosti unutar **dvostrukih navodnika**
    - Vrijednost varijable **duljina** biti će **8**
    - U memoriji zauzeto dovoljno mjesta za spremanje 8 podataka tipa char -> 16 byte

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Znakovi su indeksirani prema rednom mjestu spremanja u memoriju
    - Prvi znak je na **poziciji 0**
  - Dohvaćanje pojedinog znaka

```
static void Main(string[] args)
{
    string s = "Programiranje";
    char c = s[3];
    Console.WriteLine(c); // ispisuje g
    Console.ReadKey();
}
```

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Metode koje olakšavaju rad s varijablama tipa **string**

- `Contains`
- `StartsWith`
- `EndsWith`
- `Trim`
- `Remove`
- `Replace`
- `ToLower`
- `ToUpper`
- `IndexOf`

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - **Metoda "Contains"**
    - Vraća logičku vrijednost "**true**" ukoliko znakovni niz sadrži zadani podniz

```
static void Main(string[] args)
{
    string s = "Programiranje";
    bool a = s.Contains("mir"); // true
    bool b = s.Contains("aaa"); // false
    bool c = s.Contains("MIR"); // false
    Console.ReadKey();
}
```

    - C# razlikuje znakove velikih i malih slova

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Metode **"StartsWith"** i **"EndsWith"**
    - Vraćaju logičku vrijednost **"true"** ukoliko znakovni niz počinje / završava sa zadanim podnizom

```
static void Main(string[] args)
{
    string s = "Programiranje";
    bool a = s.StartsWith("Prog"); // true
    bool b = s.StartsWith("abcd"); // false
    bool c = s.EndsWith("e");      // true
    bool d = s.EndsWith("Prog");   // false
    Console.ReadKey();
}
```

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Metoda "**`Trim`**"
    - Uklanja razmake s početka i kraja znakovnog niza ukoliko oni postoje

```
static void Main(string[] args)
{
    string s = "  Programiranje  ";
    s = s.Trim(); // uklonjeni razmaci
    Console.ReadKey();
}
```

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Metoda "**Remove**"
    - Uklanja određen broj znakova počevši od zadane pozicije

```
static void Main(string[] args)
{
    string s = "Programiranje";
    s = s.Remove(3, 4);
    Console.WriteLine(s); // ispisuje Proiranje
    Console.ReadKey();
}
```

    - Prvi argument označava poziciju od koje se znakovi uklanjaju, a drugi argument broj znakova koje treba ukloniti
    - Metodu je moguće pozvati bez drugog argumenta čime se od zadane pozicije uklanjaju svi znakovi do kraja stringa

```
Console.WriteLine(s.Remove(3)); // ispisuje Pro
```

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Metoda "**`Replace`**"
    - Zamjenjuje sve znakove ili podnizove u **`string`**-u sa zadanim znakom ili podnizom

```
static void Main(string[] args)
{
    string s = "Programiranje";
    string s1 = s.Replace('a', 'A');  // ProgrAmirAnje
    string s2 = s.Replace("ra", "XX");// ProgXXmiXXnje
    Console.ReadKey();
}
```

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Metoda "**ToLower**"
    - Pretvara velika slova u mala slova
  - Metoda "**ToUpper**"
    - Pretvara mala slova u velika slova

```
static void Main(string[] args)
{
    string s = "Programiranje";
    Console.WriteLine(s.ToLower()); // programiranje
    Console.WriteLine(s.ToUpper()); // PROGRAMIRANJE
    Console.ReadKey();
}
```

Pohrana podataka – Složeni tipovi

- Pohrana niza znakova
  - Metoda "**`IndexOf`**"
    - Vraća poziciju prvog pojavljivanja zadanog znaka ili podniza u znakovnom nizu

```
static void Main(string[] args)
{
    string s = "Programiranje";
    int a = s.IndexOf("g");   // rezultat 3
    int b = s.IndexOf("mir"); // rezultat 6
    int c = s.IndexOf("x");   // rezultat -1
    Console.ReadKey();
}
```

    - Postoji više varijanti pozivanja, npr.

```
// prvo pojavljivanje gledano od pozicije 2
int a = s.IndexOf("r", 2); // rezultat 4
```

Pohrana podataka – Složeni tipovi

- Grupiranje podataka različitog tipa
  - Koriste se strukture
  - Ključna riječ "**struct**"
  - Veličina ovisi o grupiranim varijablama
  - Primjer

```
public struct Vozilo {
        public    int    snagaMotora;
        public    bool   motorUkljucen;

        public    void   PodesiSnagu(int  p1){
                  snagaMotora = p1;
        }
        public    void   UkljuciMotor(bool  p2){
                   motorUkljucen = p2;
        }
}
```

Pohrana podataka – Složeni tipovi

- Grupiranje podataka različitog tipa
  - Korištenje strukture Vozilo

```
// glavna metoda
static void Main(string[] args)
{
    // deklaracija strukture
    Vozilo vozilo = new Vozilo();

    // inicijalizacija vrijednosti strukture
    vozilo.PodesiSnagu(100);
    vozilo.UkljuciMotor(true);

    // dohvat vrijednosti iz strukture
    System.Console.WriteLine("Snaga motora je " + vozilo.snagaMotora + " kW.");
    System.Console.WriteLine("Motor je ukljucen: " + vozilo.motorUkljucen );

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

}
```

📂 **StrukturaVozilo**

  - Dohvat podataka ide pomoću operatora točka