Datoteke I

Osnove: Razredi Directory, File in Path

Datoteke - osnove

- Glede na tip zapisa jih delimo na:
  - **tekstovne** datoteke in
  - **binarne** datoteke

- Obravnavali bomo le **Tekstovne** datoteke
  - vsebujejo nize znakov oziroma zlogov, ki so ljudem berljivi – odpremo jih lahko s katerimkoli urejevalnikom (Beležnica, WordPad, Word…)
  - v tekstovnih datotekah so zapisani tudi določeni t.i. kontrolni znaki. Najpomembnejša tovrstna znaka sta znaka za konec vrstice (***end of line***) in konec datoteke (***end of file***).
  - tekstovne datoteke razdeljene na **vrstice**.

Datoteke

- Za delo z datotekami v jeziku C# so zadolžene metode iz razredov v imenskem prostoru *System.IO*. Zato na začetku vsake datoteke, v kateri je program, ki dela z datotekami, napišemo

```
using System.IO;
```

- *Razredi za delo z imeniki, datotekami in potmi do datotek.*
  - ***Directory:*** *Uporablja se za kreiranje, urejanje, brisanje ali pridobivanje informacij o imenikih.*
  - ***File:*** *Uporablja se za kreiranje, urejanje, brisanje ali za pridobivanje informacij o datotekah.*
  - ***Path:*** *Uporablja se za pridobivanje informacij o poteh do datotek.*

Datoteke – razred Directory

- Najpomembnejše metode razreda *Directory*:
  - **Exists(path)** Vrne logično vrednost, ki ponazarja ali nek imenik obstaja ali ne.
  - **CreateDirectory(path)** Kreira imenike v navedeni poti.
  - **Delete(path)** Brisanje imenika in njegove vsebine.
  - **GetFiles(path)** Pridobivanje imen datotek navedene poti
  - **GetDirectories(pot)** Pridobivanje imen map navedene poti
  - **GetCreationTime(pot)** Pridobivanje datuma kreiranja mape
  - **GetDirectoryRoot(pot)** Pridobivanje imena logične enote, na kateri se nahaja določena mapa
  - **GetCurrentDirectory()** Pridobivanje poti do tekoče mape

Razred Directory - zgledi

```
string dir = "C:\\Vaje C#";  //Lahko tudi string dir = @"C:\Vaje C#";
if (!Directory.Exists(dir)) //Preverimo, če imenik C:\Vaje C# obstaja
    Directory.CreateDirectory(dir);
string[] datoteke = Directory.GetFiles(dir);//Pridobimo imena vseg datotek v mapi
Console.WriteLine("Seznam datotek imenika Vaje C# na disku C:");
foreach (string imedatoteke in datoteke) /*izpišimo imena vseh datotek mape*/
    Console.WriteLine(imedatoteke);
try
{   Directory.Delete(dir); //Skušamo pobrisati mapo C:\Vaje C#
    Console.WriteLine("\nMapa c:Vaje C # uspešno pobrisana!");
}
catch { Console.WriteLine("\nBrisanje neuspešno, ker mapa ni prazna"); }
DateTime datum = Directory.GetCreationTime(@"c:\Windows");
Console.WriteLine(@"Mapa C:\Windows je bila kreirana " +
    datum.ToShortDateString());
tring[] mape = Directory.GetDirectories(@"C:\Program Files");//Ugotavljanje podmap
foreach (string ime in mape)//izpišimo imena vseh datotek mape C:\Vaje C#
    Console.WriteLine(ime);
string mapa = @"Program Files";
Console.WriteLine("Mapa '" + mapa + "' se nahaja na disku " +
    Directory.GetDirectoryRoot(@"c:\Program Files"));
string tekocamapa = Directory.GetCurrentDirectory(); //ugotovimo celotno pot do
    mape
Console.WriteLine(tekocamapa);//izpis tekoče mape
string dir1 = "C: \\C# 2010\\Datoteke\\"; //ali @"C: \C# 2005\Datoteke\"
if (!Directory.Exists(dir1))  //če imenik še ne obstaja, ga skreiramo
       Directory.CreateDirectory(dir1);
Console.ReadKey();
```

Datoteke – razred Path

- Najpomembnejše metode razreda *Path*:
  - **GetFullPath(pot)** Pridobivanje celotne poti do datoteke

```
string ime = "Datoteka.txt";
/*celotno pot do datoteke dobimo s pomočjo metode
   GetFullPath razreda Path*/
string celotnaPot = Path.GetFullPath(ime);
Console.WriteLine(celotnaPot);/*izpis celotne poti do
   datoteke*/
```

Datoteke – Razred File

- Najpomembnejše metode razreda *File*:
  - **Exists(path)**Vrne logično vrednost, ki ponazarja ali neka datoteka obstaja ali ne.
  - **Delete(path)**Brisanje datoteke.
  - **Copy(source, dest)**Kopiranje datoteke iz izvorne poti (*source*) do končne poti (*dest*).
  - **Move(source, dest)**Premik datoteke iz izvorne poti (*source*) do končne poti (*dest*).
  - **CreateText(pot)**Kreiranje ali odpiranje tekstovne datoteke za pisanje. Parameter *pot* vsebuje pot do datoteke in njeno ime
  - **OpenText(pot)**Odpiranje obstoječe tekstovne datoteke za branje. Parameter *pot* vsebuje pot do datoteke in njeno ime
  - **OpenRead(pot)**Odpiranje obstoječe datoteke za branje. Parameter *pot* vsebuje pot do datoteke in njeno ime
  - **OpenWrite(pot)**Odpiranje obstoječe datoteke za pisanje. Parameter *pot* vsebuje pot do datoteke in njeno ime

Datoteke – Razred File

- **AppendText(pot)**Odpiranje obstoječe tekstovne datoteke za pisanje in dodajanje teksta v to datoteko. Če datoteka še ne obstaja, se le-ta najprej ustvari. Parameter *pot* vsebuje pot do datoteke in njeno ime
- **ReadAllText(pot)**Odpiranje obstoječe tekstovne datoteke, branje vseh vrstic iz te datoteke, nato pa še zapiranje datoteke. Parameter *pot* vsebuje pot do datoteke in njeno ime
- **WriteAllText(pot,stavek)**Kreiranje nove datoteke, zapis stavka v datoteko in zapiranje datoteke. Če datoteka s tem imenom že obstaja, bo prepisana.
- **WriteAllLines(pot,stavki)**Kreiranje nove datoteke, zapis stavkov (npr. iz tabele stavkov) v datoteko in zapiranje datoteke. Če datoteka s tem imenom že obstaja, bo prepisana.
- **AppendAllText(pot,stavek)**Zapis stavka v datoteko. Če datoteka še ne obstaja, bo skreirana nova.

Datoteke – Metodi Exists in CreateText

- Napišimo program, ki nas vpraša po imenu tekstovne datoteke

```
Console.Write("Ime datoteke: ");
string ime = Console.ReadLine();
if (File.Exists(ime))
{
    Console.WriteLine("Datoteka " + ime + " že
obstaja!");
}
else
{
    StreamWriter pisi=  File.CreateText("Vaja.txt");
    pisi.Close();
    Console.WriteLine("Datoteko " + ime + " smo
naredili!");
}
```

Datoteka – metode

- Metoda ***WriteAllLines***: V datoteko zapišimo celotno tabelo imen

```
string[] Imena = {"Janko","Metka","Anja","Petra" };
File.WriteAllLines("Imena.txt", Imena);
```

- Metoda ***WriteAllText*** in ***AppendAllText***: V datoteko zapišimo poljuben niz in dodajmo še enega

```
string stavek = "Ta stavek bomo zapisali v datoteko!";
File.WriteAllText("Vaja.txt", stavek);
string stavek1 = "Tale stavek bomo dodali v datoteko";
File.AppendAllText("Vaja.txt", stavek1);
```

Datoteke – metode

- Metoda delete

```
string imeDatoteke = "Vaja.txt";
if (File.Exists(imeDatoteke))
    File.Delete(imeDatoteke);
```

- Metoda Copy

```
string imeDatoteke = "Vaja.txt";
if (File.Exists(imeDatoteke)) /*če obstaja datoteka s tem
   imenom*/
    File.Copy(imeDatoteke,"Rezerva.txt");//naredimo
   kopijo
```

Datoteka – pisanje v datoteko

- Ustvari novo datoteko in vanjo zapiši stavek

```
string imeDatoteke = “Vaja.txt”;
StreamWriter pisi = File.CreateText(imeDatoteke);
pisi.WriteLine(“Ta stavek gre v datoteko!");
pisi.Close();
```

Datoteke - branje

- Metoda ***ReadAllText***

```
//vsebino datoteke v celoti preberemo v niz
string vsebina = File.ReadAllText("Vaja.txt");
Console.WriteLine(vsebina);//izpis niza
```

- Metoda ***ReadAllLines***

```
//vsebino datoteke zapišemo v tabelo
string[] stavki = File.ReadAllLines("Vaja.txt");
```

Vaje

- V tekstovno datoteko zapiši svoje osebne podatke, vsak podatek v svojo vrstico!
- V tekstovno datoteko zapiši prvih 100 naravnih števil, vsakega v svojo vrstico!
- V tekstovno datoteko zapisuj stavke, ki jih uporabnik vnaša preko tipkovnice. Znak za konec zapisovanja je prazen stavek!

Datoteke - branje

- V tekstovni datoteki so sam cela števila, vsako število v svoji vrsti. Koliko števil je v datoteki, in kolikšna je njihova povprečna vrednost?

```
string ime = "Stevila.txt";//ime datoteke
int vsota = 0;//začetna vsota števil v datoteki
int stVrstic=0;//začetno število vrstic v datoteki
StreamReader beri = File.OpenText("ime");//odpiranje za branje
string vrstica = beri.ReadLine();//skušao brati prvo vrstico
while (vrstica != null)//če je bilo branje uspešno
{
    stVrstic++;
    int stevilo = Convert.ToInt32(vrstica);//ptevorba v število
    vsota = vsota + stevilo;
    vrstica = beri.ReadLine();//skušamo brati naslednjo vrstico

}
beri.Close();//Zapremo datoteko
Console.WriteLine("V datoteki je " + stVrstic + " vrstic");
Console.WriteLine("POvprečje vseh števil: " + (double)vsota /
   stVrstic);
Console.ReadKey();
```

Vaje

- V tekstovni datoteki Kraji.txt so zapisana imena krajev, vsak kraj v svoji vrstici.
  - Koliko je vseh krajev v datoteki
  - Ugotovi in izpiši kraj z največjim številom znakov
  - Kolikšno je skupno število vseh znakov v datoteki
  - Kolikokrat se v datoteki nahaja kraj Kranj
  - Vsebino datoteke prepiši v novo datoteko Pošte.txt, pri čemer vsakemu kraji dodaš še poštno številko – poštne številke vnaša uporabnik programa!

- V tekstovni datoteki Zneski.txt je zapisano neznano število števil tipa double. Vsebino datoteke prenesi v tabelo, zneske tam uredi po velikosti in jih nato zapiši nazaj v datoteko z istim imenom.