

Darbs ar datnēm

Šajā nodaļā:

- Līdzekļi darbam ar datnēm;
- Datņu plūsmas;
- Plūsmas sasaiste ar datni;
- Datu ierakstīšana datnē;
- Datu lasīšana no datnes;
- Datnes ar nezināmu garumu apstrāde;
- Uzdevumi.

Darbs ar datnēm

.NET Framework satur dažādus līdzekļus darbam ar datnēm.

Datne – tā datu kopa, kas ir saglabāta datu nesējā ar izvelēto nosaukumu. Kad datne tiek atvērta lasīšanai vai rakstīšanai, ***tā kļūst par plūsmu***.

Vispārīgi *plūsma* – tā ir baitu virkne, kuras uzvedība var tikt ietekmēta ar dažādu metožu palīdzību.

Kopumā eksistē divas pamata plūsmas: ***ievades plūsma*** un ***izvades plūsma***.

Ievades plūsma tiek izmantota datu lasīšanai no datnes (operācija *read*), izvades plūsma tiek izmantota datu rakstīšanai datnē (operācija *write*).

Darbs ar datnēm

C# valoda atbalsta visas standarta operācijas, kas ir saistītas ar ievades un izvades datņu plūsmām.

Nepieciešamas *klases* (*sk. piezīmi*) ir iespējams izmantot, pieslēdzot atbilstošo *namespace: System.IO*.

Piezīme: Kas ir **Objektorientētā programmēšana (OOP)**:

[http://lv.wikipedia.org/wiki/Objektorient
%C4%93t%C4%81_programm
%C4%93%C5%A1ana](http://lv.wikipedia.org/wiki/Objektorient%C4%93t%C4%81_programm%C4%93%C5%A1ana)

Darbs ar datnēm

Dažas *System.IO* klases:

I/O Klase	Apraksts
<i>BinaryReader</i>	Nolasa datus no bināras plūsmas.
<i>BinaryWriter</i>	Ieraksta datus binārajā formātā.
<i>BufferedStream</i>	Pagaidu objekts baitu plūsmas saglabāšanai .
<i>Directory</i>	Palīdz manipulēt ar folderu struktūru.
<i>DirectoryInfo</i>	Nodrošina informāciju par folderu.
<i>DriveInfo</i>	Satur informāciju par datu nesēju.
<i>File</i>	Palīdz manipulēt ar datņu struktūru.
<i>FileInfo</i>	Nodrošina informāciju par datni.
<i>FileStream</i>	Klase, kas realizē lasīšanas un rakstīšanas darbības ar datnes plūsmu.
<i>MemoryStream</i>	Klase, kas realizē pamata operācijas darbam ar plūsmu atmiņā.
<i>Path</i>	Realizē pamata operācijas darbam ar Path informāciju.
<i>StreamReader</i>	Tiek izmantota lai nolasītu simbolus no baitu plūsmas.
<i>StreamWriter</i>	Tiek izmantota lai rakstītu simbolus baitu plūsmā.
<i>StringReader</i>	Tiek izmantota lai lasītu no rindas(string) plūsmas.
<i>StringWriter</i>	Tiek izmantota lai rakstītu rindas(string) plūsmā.

Darbs ar datnēm

Klase *FileStream*

Klase *FileStream*, kas ir aprakstīta iekš *System.IO* nodrošina iespējas lasīt, rakstīt datnēs, kā arī atvērt un korekti aizvērt datnes.

Lai sāktu darbu ar datni (izveidot vai atvērt), ir nepieciešams izveidot *FileStream* objektu.

Sintakse ir sekojoša:

```
FileStream <object_name> =  
    new FileStream( <file_name>,  
        <FileMode Enumerator>,  
        <FileAccess Enumerator>,  
        <FileShare Enumerator>);
```

Darbs ar datnēm

Piemēram, lai izveidotu *FileStream* objektu *F* priekš lasīšanas no datnes ar nosaukumu „*sample.txt*”, ir nepieciešams:

```
FileStream F =  
    new FileStream("sample.txt",  
        FileMode.Open,  
        FileAccess.Read,  
        FileShare.Read);
```


Darbs ar datnēm

FileStream klases konstruktora parametru apraksts:

Parametrs	Apraksts
FileMode	Enumerators FileMode definē dažādus režīmus darbam ar datni: Append : Atvērt eksistējošo datni (vai izveidot, ja neeksistē) un pārvietot kursora pozīciju datnes beigās; Create : Izveidot datni (ja eksistē, tiks pārrakstīta); Open : Atvērt eksistējošo datni; OpenOrCreate : Atvērt, ja eksistē vai izveidot, ja neeksistē; Truncate : Atvērt eksistējošo datni un iestatīt izmēru 0 (izdzēst saturu).
FileAccess	FileAccess definē trīs piekļuves režīmus: Read , ReadWrite and Write .
FileShare	FileShare enumerators definē: None : Aizliagt citiem procesiem piekļuvi datnei; Read : Atļaut lasīšanu citiem; Write : Atļaut rakstīšanu citiem; ReadWrite : Atļaut lasīšanu un rakstīšanu citiem.

Darbs ar datnēm

Datu rakstīšana/lasīšana

Piemērs:

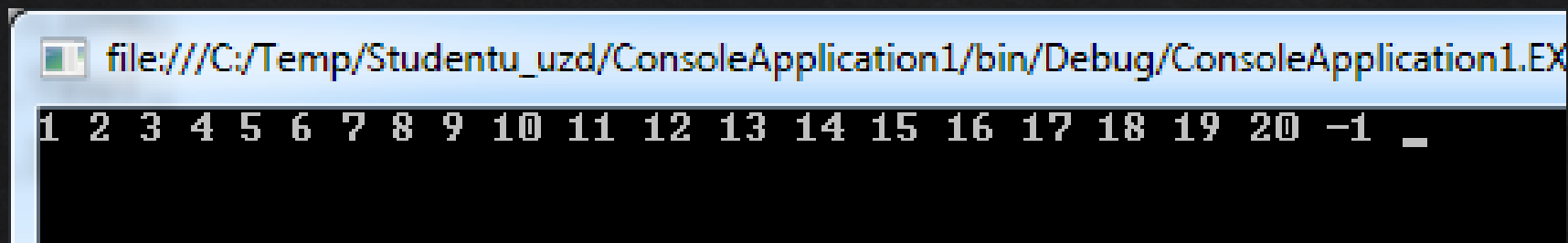
```
using System;
using System.IO; //darbam ar datnēm!
...
static void Main(string[] args) {
    FileStream F = new FileStream("test.dat",
        FileMode.Create, FileAccess.ReadWrite);
    for (int i = 1; i <= 20; i++) {
        F.WriteByte((byte)i); }
    F.Position = 0;
    for (int i = 0; i <= 20; i++) {
        Console.Write(F.ReadByte() + " "); }
    F.Close();
    Console.ReadKey(); //pauze beigās
}
```


Darbs ar datnēm

Datu rakstīšana/lasīšana datnē

Programmas piemērā ar klases *FileStream* objekta *F* palīdzību tiek izveidota datne "test.dat".

Pirmajā *for* ciklā datnē tiek ierakstīti 20 baiti (intervāls no 1 līdz 20). Pēc tam, pateicoties koda rindai *F.Position = 0*; tekošā kursora lasīšanas / rakstīšanas pozīcija tiek pārvietota datnes sākumā. Otrajā *for* ciklā ar *F.ReadByte()* metodes palīdzību tiek nolasīts kārtējais simbols (*byte*) un uzreiz arī tiek izvadīts uz ekrāna. Pēdējais "-1" izvadē parāda to, ka ir sasniegtas datnes beigas, t.i. tālāka nolasīšana nav iespējama / bezjēdzīga.



```
file:///C:/Temp/Studentu_uzd/ConsoleApplication1/bin/Debug/ConsoleApplication1.EXE
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 -1 _
```

Programmas izvade

Darbs ar datnēm

Datu rakstīšana datnē

Piemērs:

```
static void Main(string[] args) {  
    double d = 10.0;  
    FileStream fs = new FileStream("dati.txt",  
        FileMode.Create, FileAccess.ReadWrite);  
    StreamWriter writer = new StreamWriter(fs);  
    writer.WriteLine("Teksta piemērs");  
    for (int i = 1; i <= 10; i++) {  
        writer.Write(i + " "); }  
    writer.WriteLine();  
    for (int i = 1; i <= 10; i++) {  
        writer.Write(i / d + " "); }  
    writer.Close();  
    Console.ReadKey(); //pauze beigās  
}
```


Darbs ar datnēm

Datu rakstīšana datnē

Programmas darbības rezultātā tiek izveidota datne "dati.txt" mapē, kur atrodas programmas izpildāmā datnes, un šo datni var atvērt, izmantojot kādu standarta teksta apstrādes lietotni.

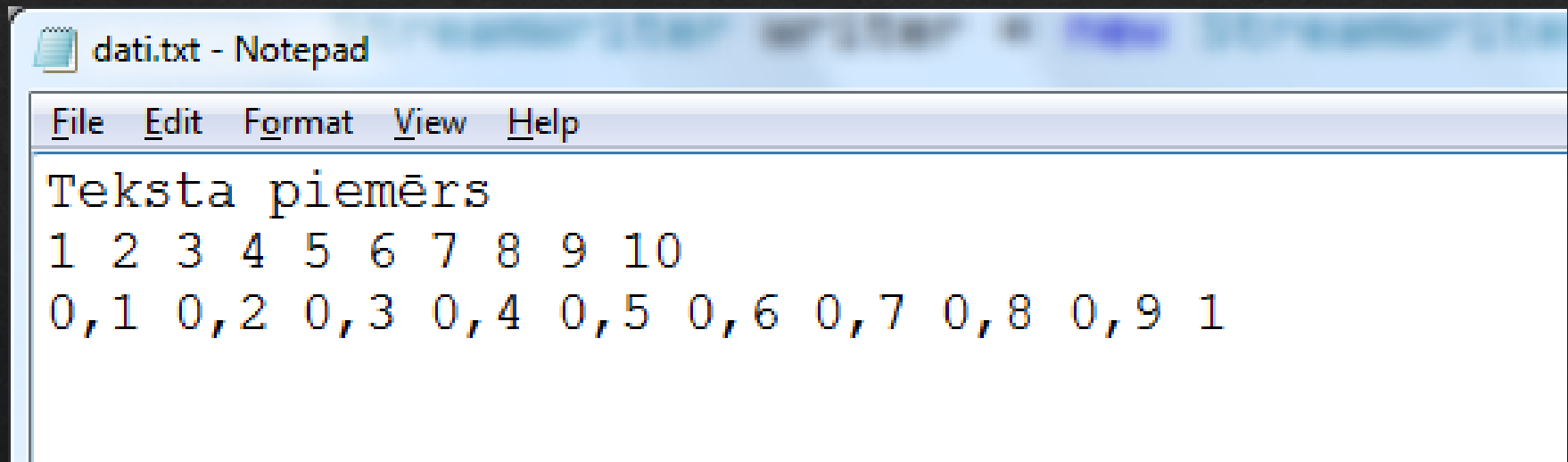
Izmantojot izvades plūsmas iespējas (***StreamWriter*** klases objektu), pietiekami elementāri var nodrošināt informācijas izvadi datnē. Dotajā piemērā datnē tiek ierakstīta teksta rinda, tad, izmantojot *for* cikla konstrukciju, veseliem skaitļiem no 1 līdz 10, kā arī decimālskaitļiem no 0,1 līdz 1 (pēdējā *for* ciklā).

Jāņem vērā, ka rakstīšanas procesa beigās ir jāaizver datne, lai sistēma būtu spējīga korekti pabeigt datu ierakstīšanas procesu. To nodrošina speciāla metode *Close*.

Darbs ar datnēm

Datu rakstīšana datnē

Programmas darbības rezultātā tiek izveidota datne "dati.txt" mapē, kur atrodas programmas izpildāmā datnes, un šo datni var atvērt, izmantojot kādu standarta teksta apstrādes lietotni.



The screenshot shows a Notepad window titled "dati.txt - Notepad". The menu bar includes File, Edit, Format, View, and Help. The text content is as follows:

```
Teksta piemērs  
1 2 3 4 5 6 7 8 9 10  
0,1 0,2 0,3 0,4 0,5 0,6 0,7 0,8 0,9 1
```

Izveidotā datne ar ierakstīto informāciju

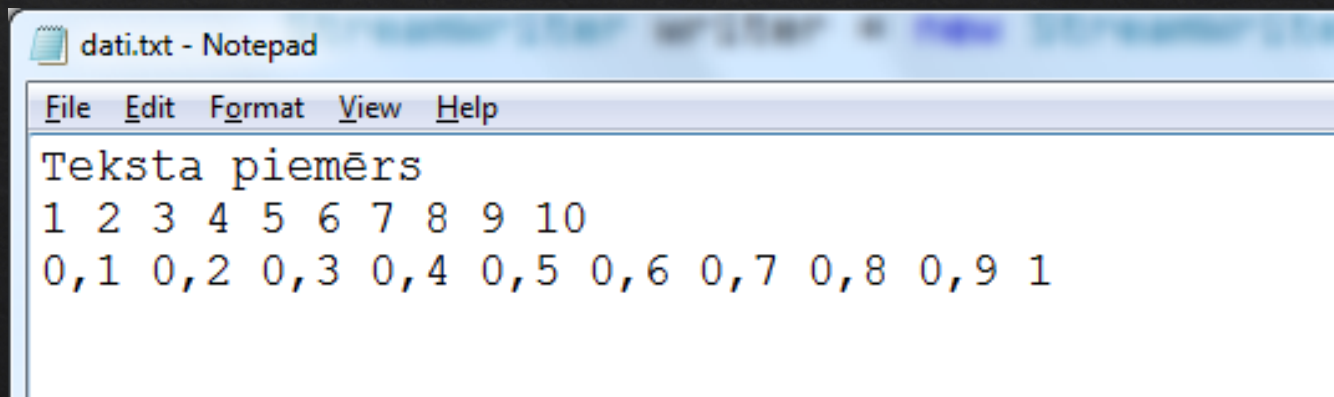
Darbs ar datnēm

Datu nolasīšana no datnes

Datu nolasīšana no datnes notiek, izmantojot *FileStream* un *StreamReader* klašu objektus.

Ja kāda datne jau eksistē un iepriekš ir zināma datnes struktūra, tad šī struktūra ir jāievēro datu lasīšanas laikā.

Piemēram, ja datnes struktūra ir tāda, kāda tā ir redzama attēlā, tad ir jāparedz, ka vispirms ir jānolasa teksta rinda, tad desmit veseli skaitļi un beigās desmit decimālskaitļi.



```
dati.txt - Notepad
File Edit Format View Help
Teksta piemērs
1 2 3 4 5 6 7 8 9 10
0,1 0,2 0,3 0,4 0,5 0,6 0,7 0,8 0,9 1
```

Darbs ar datnēm

Datu nolasīšana no datnes (1)

Piemērs:

```
static void Main(string[] args) {  
    string teksta_rinda = "";  
    string line = "";  
    int[] intSkaitli = new int[10];  
    double[] doubleSkaitli = new double[10];  
    FileStream fs = new FileStream("dati.txt",  
        FileMode.Open, FileAccess.Read);  
    StreamReader reader = new StreamReader(fs);
```

Programmas sākumā tiek definēti mainīgie, kuros ir paredzēts nolasīt datus no datnes "dati.txt", kā arī tiek atvērta datne lasīšanai. Klases *StreamReader* objekts *reader*, realizē dažādas lasīšanas palīgmetodes (sk. tālāk).

Darbs ar datnēm

Datu nolasīšana no datnes (2)

Piemērs:

```
teksta_rinda = reader.ReadLine();  
  
line = reader.ReadLine();  
string[] arrayInt = line.Split(' ');  
line = reader.ReadLine();  
string[] arrayDouble = line.Split(' ');  
reader.Close();
```

Programmas fragmentā ar objekta *reader* metodes *ReadLine()* palīdzību no datnes tiek nolasīta visa informācijas (trīs rindas). Lai korekti apstrādātu 2. un 3. rindu, tiek izsaukta *String* klases metode *Split()*, kas sadala teksta rindu pa fragmentiem, ņemot vērā norādīto atdalītāju (šeit tukšumzīme). Rindas fragmenti (skaitļi) tiek ierakstīti *string* tipa viendimensijas masīvā.

Darbs ar datnēm

Datu nolasīšana no datnes (3)

Piemērs:

```
for (int i = 0; i < 10; i++) {  
    intSkaitli[i] = int.Parse(arrayInt[i]);  
    doubleSkaitli[i] = double.Parse(arrayDouble[i]);  
}
```

Programmas fragmentā skaitļi, kas šobrīd ir ierakstīti *string* masīvos kā teksta rindas, tiek konvertēti (metode *Parse*) skaitļos un tiek ierakstīti atbilstošajās pozīcijās *int* un *double* tipa masīvos.

Darbs ar datnēm

Datu nolasīšana no datnes (4)

Piemērs:

```
Console.WriteLine(teksta_rinda);  
for (int i = 0; i < 10; i++) {  
    Console.Write(intSkaitli[i] + " ");  
}  
Console.WriteLine();  
for (int i = 0; i < 10; i++) {  
    Console.Write(doubleSkaitli[i] + " ");  
}  
Console.ReadKey(); //pauze beigās  
}
```

Programmas beigās pārbaudes nolūkiem visi dati tiek izvadīti uz ekrāna.

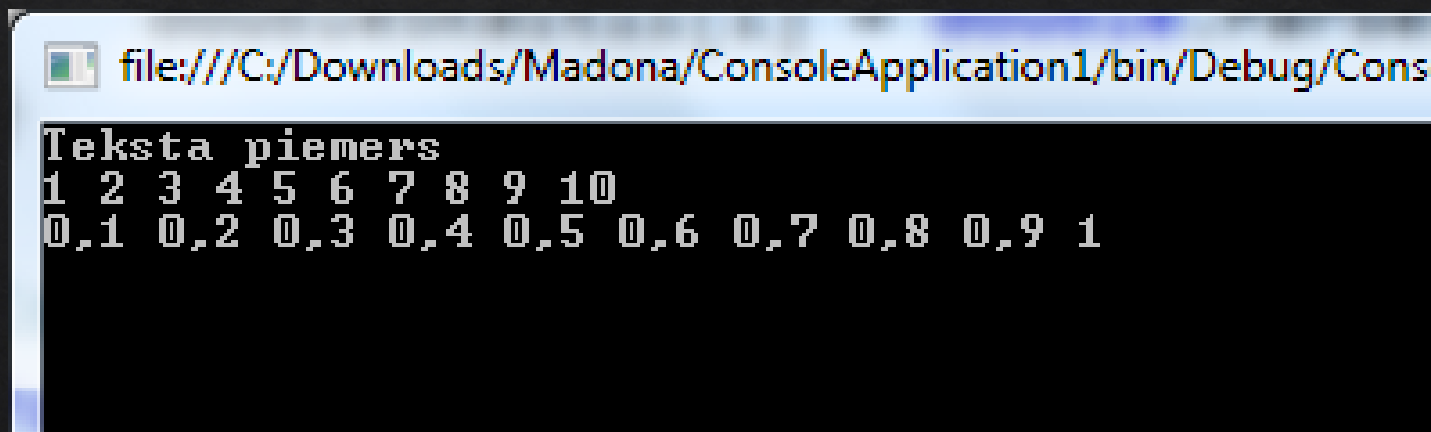
Darbs ar datnēm

Datu nolasīšana no datnes (5)

Programmas darbības rezultāts, kad dati no datnes tiek ierakstīti programmas iekšējās datu struktūrās.

Šajā gadījumā tās ir *string* tipa mainīgais un divi masīvi:

- *string* tipa mainīgais *teksta_rinda* ir paredzēts, lai glabātu teksta informāciju;
- *int* tipa masīvs *intSkaitli* saturēs veselus skaitļus;
- *double* tipa masīvs *doubleSkaitli* saturēs decimālus skaitļus.



The screenshot shows a Windows console window with a title bar indicating the file path: `file:///C:/Downloads/Madona/ConsoleApplication1/bin/Debug/Cons`. The console output displays the text "Teksta piemers" followed by two lines of data. The first line contains integers from 1 to 10, and the second line contains decimal values from 0.1 to 1.0.

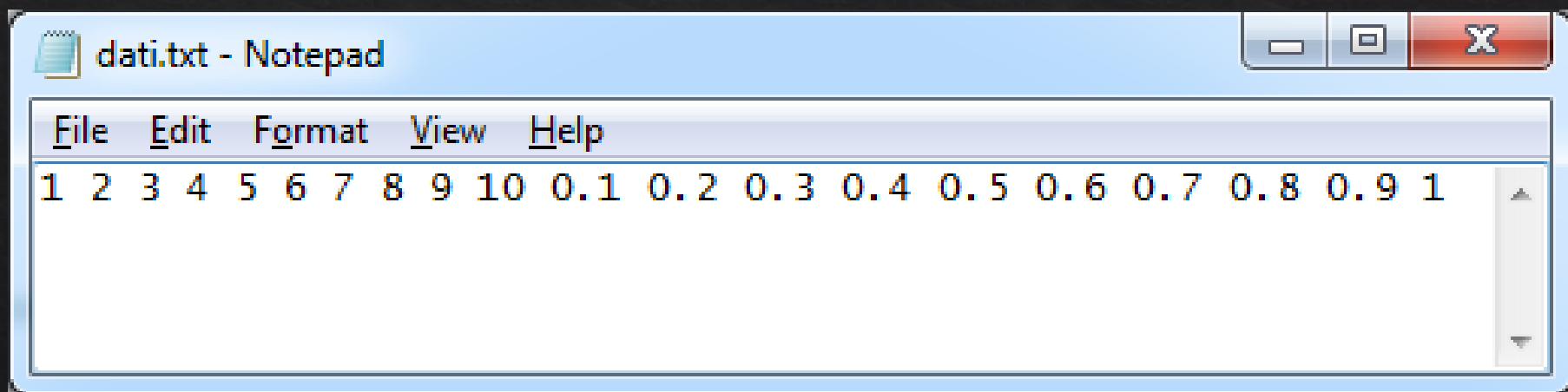
```
Teksta piemers
1 2 3 4 5 6 7 8 9 10
0,1 0,2 0,3 0,4 0,5 0,6 0,7 0,8 0,9 1
```


Darbs ar datnēm

Datu nolasīšana no datnes

Dažreiz datnes struktūra līdz galam nav zināma vai ir zināma daļēji. C# valodā piedāvātie rīki ļauj apstrādāt arī šādas situācijas.

Piemēram, eksistē kāda datne, kurā ir ierakstīti skaitļi. Ir dots uzdevums uzrakstīt programmu, kura būtu spējīga nolasīt un apstrādāt datus, kas atrodas datnē (piemēram, atrast skaitļu summu).



Datne ar skaitļiem

Darbs ar datnēm

Datu nolasīšana no datnes

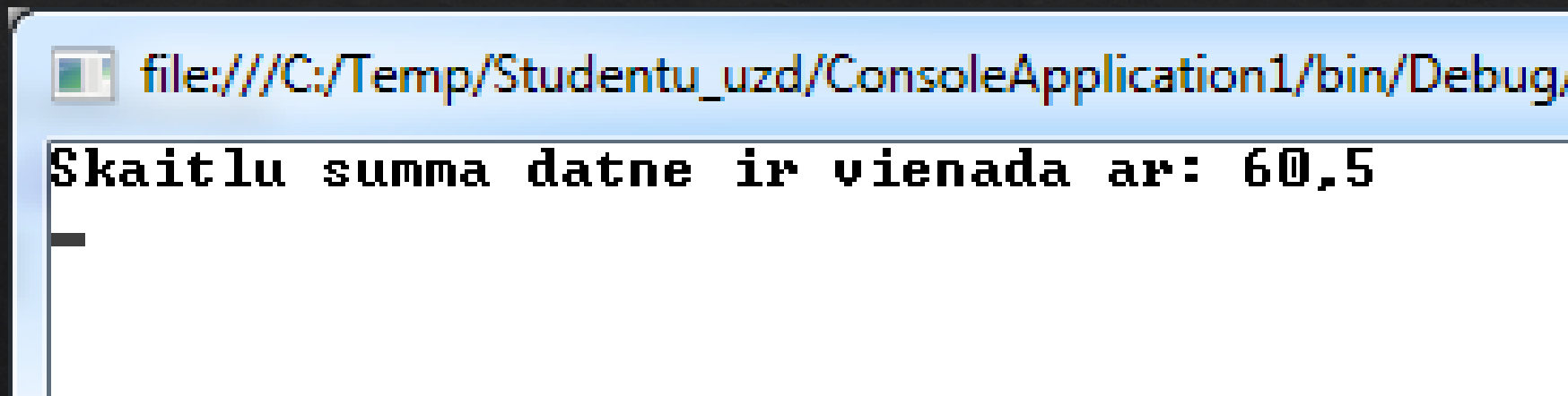
Piemērs

```
static void Main(string[] args) {  
    double sum = 0.0, tmp;  
    using (StreamReader reader = new StreamReader("dati.txt")) {  
        string line;  
        string[] ar = null;  
        while ((line = reader.ReadLine()) != null) {  
            ar = line.Split(' ');  
            for (int i = 0; i < ar.Length; i++) {  
                if (double.TryParse(ar[i], out tmp)) {  
                    sum += tmp;  
                }  
                else {  
                    Console.WriteLine("Kļūda. Nav iespējams konvertēt {0}",  
ar[i]);  
                }  
            }  
        }  
        reader.Close();  
        Console.WriteLine("Skaitļu summa datnē ir vienāda ar: {0}", sum);  
    }  
    Console.ReadKey(); //pauze beigās  
}
```


Darbs ar datnēm

Datu nolasīšana no datnes

Piedāvātais piemērs demonstrē iespējamo uzdevuma risināšanas gaitu, kad iepriekš nevar strikti pateikt, kāds ir datnes garums. Mēģinot nolasīt datnes saturu pa rindām, vienlaikus tiek kontrolēts vai nav sasniegtas datnes beigas. Konvertācija notiek ar pārbaudi (*tryparse*), lai nepieciešamības gadījumā izvadītu paziņojumu.



```
file:///C:/Temp/Studentu_uzd/ConsoleApplication1/bin/Debug/
Skaitļu summa datne ir vienāda ar: 60,5
-
```

Nezināma garuma datnes apstrāde

Uzdevumi

1. Ir dots datnes nosaukums un divi vesēlie skaitļi N un K . Uzrakstīt programmu, kas izveido datni un ieraksta N rindas, katra no kurām sastāv no K '*' simboliem.
2. Uzrakstīt programmu, kas ieraksta datnē tekstu "Reizinājuma tabula $N \times M$ " un reizinājuma tabulu $N \times M$, kur N un M ievada lietotājs.
3. Uzrakstīt programmu kas piedāvā lietotājam ievadīt datnes nosaukumu un izvada uz ekrāna datnes saturu.
4. Uzrakstīt programmu, kas prasa lietotājam ievadīt datnes nosaukumu un kādu simbolu. Programmai jāizvada ievadīto simbolu skaitu dotajā datnē.

Paldies par uzmanību

Kursa materiāli Internētā:

<http://ekursi.rta.lv/>