Metoder

Upphovsrätt för detta verk

Detta verk är framtaget i anslutning till kursen Inledande programmering med C# vid Linnéuniversitetet.

Du får använda detta verk så här:

Allt innehåll i detta verk av Mats Loock, förutom fotografi samt Linnéuniversitetets logotyp och symbol, är licensierad under:

Creative Commons Erkännande-IckeKommersiell-DelaLika 2.5 Sverige licens.
http://creativecommons.org/licenses/by-nc-sa/2.5/se/

Det betyder att du i icke-kommersiella syften får:

- kopiera hela eller delar av innehållet
- sprida hela eller delar av innehållet
- visa hela eller delar av innehållet offentligt och digitalt
- konvertera innehållet till annat format
- du får även göra om innehållet

Om du förändrar innehållet så ta inte med fotografi samt Linnéuniversitetets logotyp och symbol i din nya version!

Vid all användning måste du ange källan: ”Linnéuniversitetet – Inledande programmering med C#” och en länk till https://coursepress.lnu.se/kurs/inledande-programmering-med-csharp och till Creative Common-licensen här ovan.

Här är du nu

- ✓ Programmen du hittills skrivit har ”bara” bestått av en lista med satser, ungefär på samma sätt som program skrevs under 1970-talet. Men mycket har hänt sedan dess…

Större och mer komplexa program kräver struktur

- Strukturerad programmering, ett steg mot objektorienterad programmering, är ett sätt att programmera där satser som hör ihop grupperas.
- De grupperade satserna bildar en enhet som kallas metod.
  - Det är möjligt att skicka data till en metod.
  - En metod kan returnera data.
  - Satser i en metod exekveras genom att metoden anropas.
- Omstrukturering av kod (”*refactoring*”) i en metod till flera metoder (kan) resulterar i marginellt mer kod men det uppvägs av att det blir enklare att skriva, läsa och underhålla koden.

Vad är en metod?

- ✓ Satsen `System.Console.WriteLine("Hello, world!");` anropar en metod.
  - `System` är namnutrymmet (”*namespace*”) klassen finns i.
  - `Console` är klassen (”*class*”) metoden finns i.
  - `WriteLine` är metoden (”*method*”) där satserna finns som skriver ut datat som skickas med i anropet (”*call*”).
  - `"Hello, world!"` är argumentet (”*argument*”) som skickas med till metodens parameter *(”parameter”).*
- ✓ En metod finns alltid i en klass. (En klass används för att gruppera metoder, och data, som hör ihop.)
- ✓ Den som anropar metoden kan om möjligt skicka med data via argument. Metoden tar emot datat med hjälp av parametrar.
- ✓ En metod kan returnera data via ett returvärde (”*return value*”).
  - I satsen `index = int.Parse("123");` returnerar metoden `Parse` heltalet `123`.

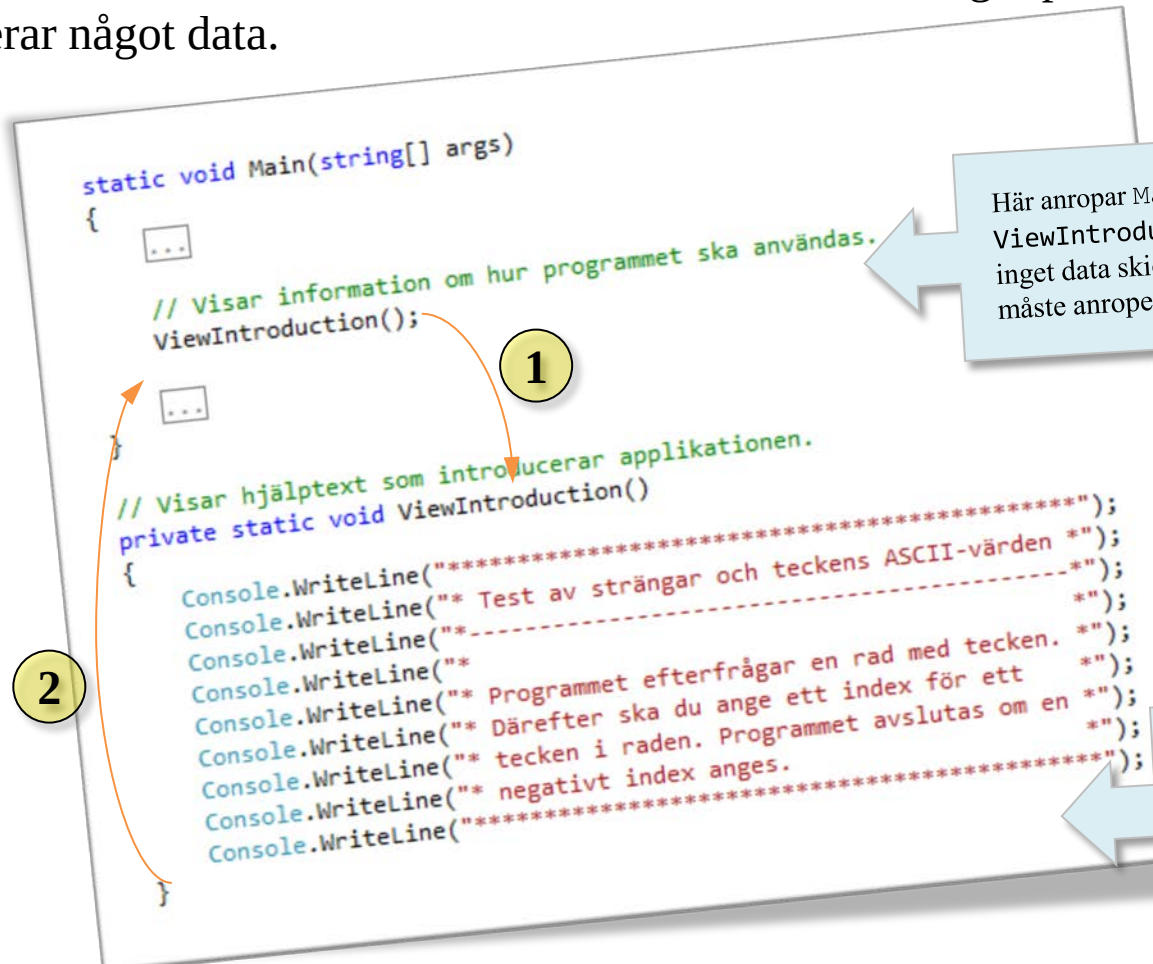
Efter omstrukturering av koden

- Koden i `Main`-metoden har delvis omstrukturerats till fyra nya metoder. Programmet har nu en bättre struktur.
- `Main`-metoden är nu enklare att läsa och kod som läser in en bokstavs index är inte längre dubblerad utan har placerats i en egen metod, `ReadCharacterIndex`, som returnerar ett värde.
- De fyra nya metoderna har olika signaturer, d.v.s. namn och parameterlistor.

```
using System;
namespace WhatAscii
{
    class Program
    {
        static void Main(string[] args)
        {
            // Deklarerar och intierar variabler.
            string line = null;
            int index = 0;

            // Visar information om hur programmet ska användas.
            ViewIntroduction();

            // Läser in en rad med text.
            line = ReadTextRow();

            // Så länge som ett "giltigt" index anges ska tecknets ASCII-värde presenteras
            // och ett nytt index ska läsas in.
            index = ReadCharacterIndex();
            while (index >= 0)
            {
                ViewCharInfo(line, index);
                index = ReadCharacterIndex();
            }
        }
        // Visar hjälptext som introducerar applikationen.
        private static void ViewIntroduction()
        {
            Console.WriteLine("**********************************************");
            Console.WriteLine("* Test av strängar och teckens ASCII-värden *");
            Console.WriteLine("*                                            *");
            Console.WriteLine("* Programmet efterfrågar en rad med tecken.  *");
            Console.WriteLine("* Därefter ska du ange ett index för ett     *");
            Console.WriteLine("* tecken i raden. Programmet avslutas om ett *");
            Console.WriteLine("* negativt index anges.                      *");
            Console.WriteLine("**********************************************");
        }
        // Läser in ett teckens index.
        private static int ReadCharacterIndex()...
        // Läser in en textrad.
        private static string ReadTextRow()...
        // Presenterar specifierat teckens ASCII-värde.
        private static void ViewCharInfo(string text, int charIndex)...
    }
}
```

Anrop av metoden ViewIntroduction

- Metoden `ViewIntroduction` är en metod som inte har några parametrar eller returnerar något data.

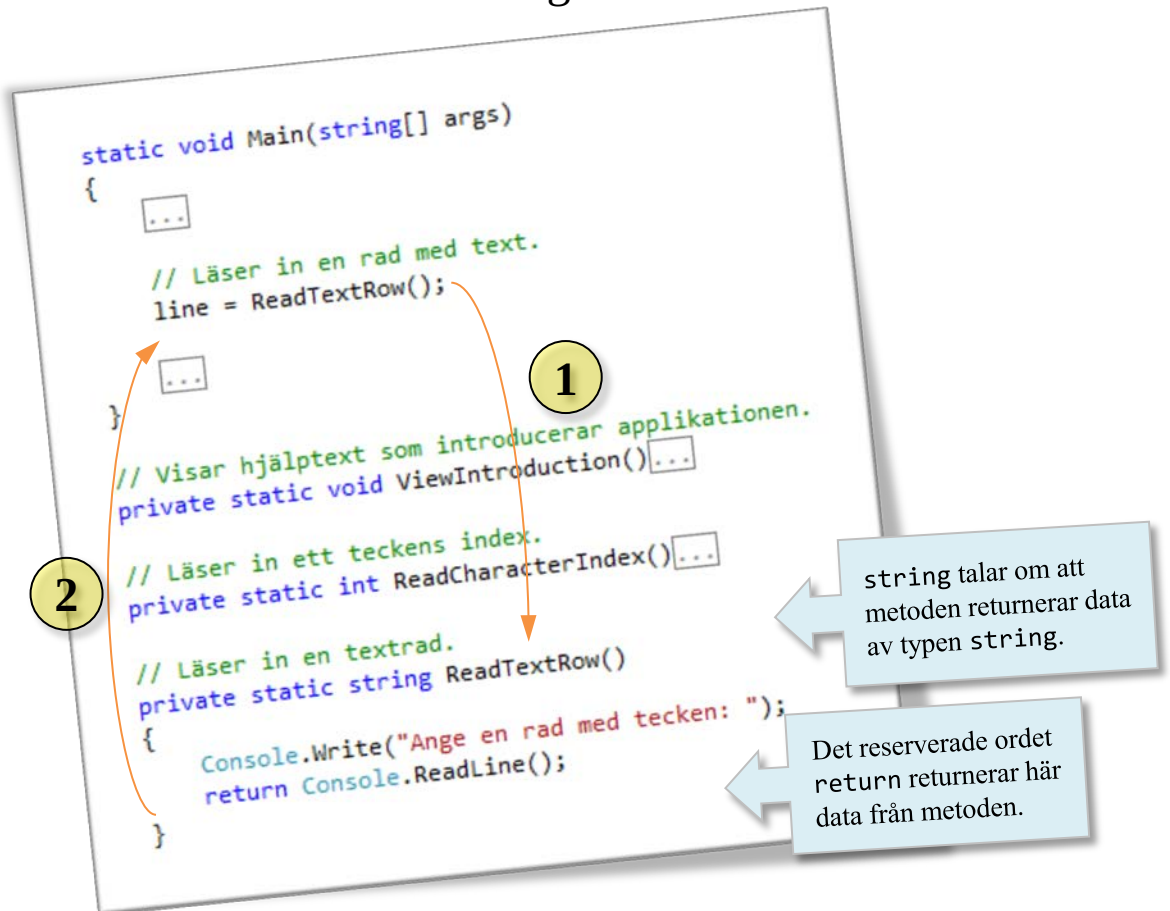


Här anropar `Main` metoden `ViewIntroduction`. Trots att inget data skickas till metoden måste anropet avslutas med `()`.

Då `ViewIntroduction` är klar återvänder programmet till `Main` för att fortsätta exekveringen av koden i `Main`.

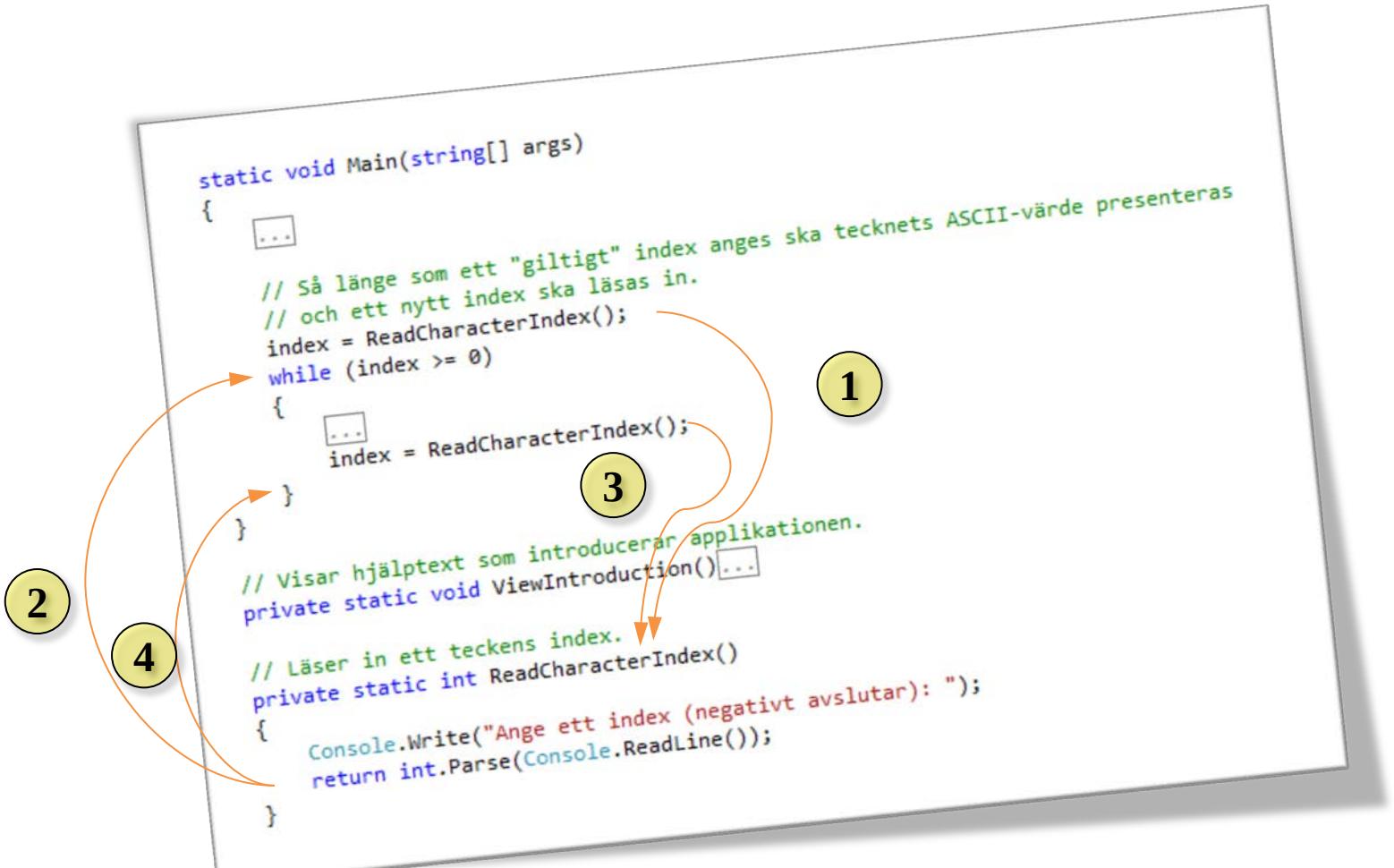
Anrop av metod som returnerar en sträng

- Metoden `ReadTextRow` skriver ut en hjälptext och returnerar den sträng användaren matar in. Variabeln `line` refererar till strängen `ReadTextRow` returnerar.



Anrop av metod som returnerar ett heltal

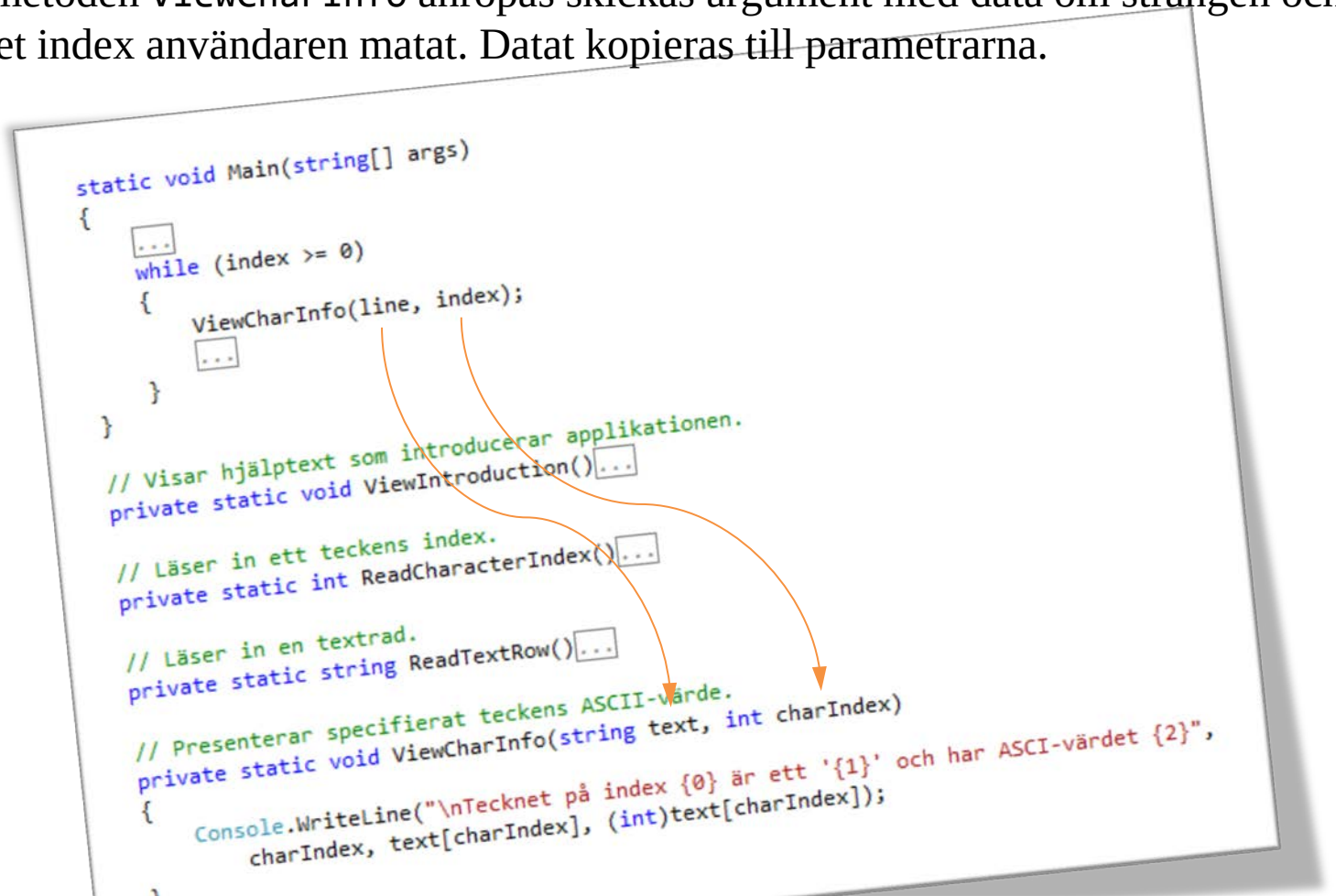
- Metoden `ReadCharacterIndex` skriver ut en ledtext och returnerar ett heltal.




```
static void Main(string[] args)
{
    ...

    // Så länge som ett "giltigt" index anges ska tecknets ASCII-värde presenteras
    // och ett nytt index ska läsas in.
    index = ReadCharacterIndex();
    while (index >= 0)
    {
        ...
        index = ReadCharacterIndex();
    }
}

// Visar hjälptext som introducerar applikationen.
private static void ViewIntroduction()...

// Läser in ett teckens index.
private static int ReadCharacterIndex()
{
    Console.Write("Ange ett index (negativt avslutar): ");
    return int.Parse(Console.ReadLine());
}
```

Skicka data till en metod

- ✓ Då metoden `ViewCharInfo` anropas skickas argument med data om strängen och vilket index användaren matat. Datat kopieras till parametrarna.



```
static void Main(string[] args)
{
    ...
    while (index >= 0)
    {
        ViewCharInfo(line, index);
        ...
    }
}

// Visar hjälptext som introducerar applikationen.
private static void ViewIntroduction()...

// Läser in ett teckens index.
private static int ReadCharacterIndex()...

// Läser in en textrad.
private static string ReadTextRow()...

// Presenterar specifierat teckens ASCII-värde.
private static void ViewCharInfo(string text, int charIndex)
{
    Console.WriteLine("\nTecknet på index {0} är ett '{1}' och har ASCI-värdet {2}",
        charIndex, text[charIndex], (int)text[charIndex]);
}
```