impera

Objektorienterad programmering med C#

Kursträff 3

Arrayer och samlingar + mer om strängar

Arrayer

- Skapa en array (s. 217):
  - ```
    int[] intArr;
    intArr = new int[10];
    ```
  - ```
    int[] intArr = new int[10];
    ```
  - ```
    var intArr = new int[10];
    ```
- Tilldela värden till en array (s. 219):
  - ```
    intArr[0] = 10;
    ```
- Tilldela och skapa samtidigt (explicit resp. Implicit deklaration):
  - ```
    int[] intArr = new int[3] { 10, 20, 30 };
    ```
  - ```
    var intArr = new [] { 10, 20, 30 };
    ```
- Diskutera:
  - Varför behöver vi inte ange vilken datatyp det är i sista exemplet?

Loopa igenom en array

- For-loop (s. 221):
  - ```
    var intArr = new [] { 10, 20, 30 };
    for (var i = 0; i < intArr.Length; i++) {
       Console.WriteLine(intArr[i]);
    }
    ```
- Foreach-loop (s. 223):
  - ```
    foreach (var num in intArr) {
       Console.WriteLine(num);
    }
    ```

Flerdimensionella arrayer

- 2d-array:
  - ```
    int[,] points = new int[2, 2];
    points[0,0] = 1;
    points[1,0] = 3; // osv…
    ```
  - ```
    var points = new [,] {
       { 1, 0 },
       { 3, 2 }
    };
    ```
- Array av arrayer (s. 228):
  - ```
    string[][] books = {
       new string[] { "Heart of Darkness", "Secret Agent" },
       new string[] { "To Kill a Mockingbird" },
    }
    ```
  - ```
    var books = new [] {
       new [] { "Heart of Darkness", "Secret Agent" },
       new [] { "To Kill a Mockingbird" },
    }
    ```

Sortera en array

- Array.Sort:
  - ```
    var intArr = new [] { 20, 10, 30 };
    Array.Sort(intArr); // intArr: {10, 20, 30}
    ```
  - ```
    var stringArr = new [] { "hej", "Hello", "abc" };
    Array.Sort(stringArr); // { "Hello", "abc", "hej" };
    ```

Kopiera en array

- Arrayer är referenstyper:
  - ```
    int[] points = {10, 20, 30};
    int[] morePoints = points;
    points[0] += 10; // morePoints[0] påverkas också!
    ```
- Array.Copy (s. 235):
  - ```
    int[] points = {10, 20, 30};
    int[] morePoints = new int[points.Length];
    Array.Copy(points, morePoints, points.Length);
    points[0] += 10; // enbart points[0] påverkas!
    ```

Null-conditional

- Ibland måste man kolla så en array eller annat objekt inte är null:
  - ```
    int[] values = null;
    int? numValues = null;
    if (values != null) {
        numValues = values.Length;
    }
    ```
- Det kan förenklas med null-conditionals:
  - ```
    int[] values = null;
    int? numValues = values?.Length; // numValues blir null
    ```
  - ```
    string[] names = null;
    string ucFirst = names?[0]?.ToUpper();
    ```
    ```

Samlingar

- Otypade samlingar:
  - ```
    using System.Collections;
    ```
  - ```
    var list = new ArrayList();
    list.Add(3);
    list.Add("Hej!"); // Ger fel när programmet körs!
    int first = (int) list[0]; // Måste castas!
    ```

- Typad samling:
  - ```
    using System.Collections.Generic;
    ```
  - ```
    var list = new List<int>();
    list.Add(3);
    // list.Add("Hej!"); // går inte att göra!
    int first = list[0]; // Ingen cast behövs!
    ```

Samlingar

- Lägga till element:
  - ```
    using System.Collections.Generic;
    ```
  - ```
    var list = new List<int>();
    list.Add(3);
    list.Add(4);
    ```
- Lägga till flera element:
  - ```
    list.AddRange(new [] {5, 6});
    ```
- Skapa och lägga till samtidigt:
  - ```
    var list = new List<int> { 3, 4, 5, 6 };
    ```

Samlingar

- Kolla om en lista innehåller ett element:
  - ```
    var list = new List<int> { 1, 2, 3 };
    if (list.Contains(3)) {
        // true!
    }
    ```
- Sortera en lista:
  - ```
    var list = new List<int> { 3, 4, 2, 1 };
    list.Sort();
    // list == { 1, 2, 3, 4 }
    ```

Samlingar

- Nyckel/värde-samlingar (ss. 248ff):
  - ```
    var ages = new SortedList<string, int>();
    ages.Add("Alice", 37);
    ages.Add("Bob", 23);
    ```
  - ```
    var ages = new SortedList<string, int> {
        { "Alice", 37 },
        { "Bob", 23 }
    };
    ```
  - ```
    var ageOfAlice = ages["Alice"]; // == 37
    if (ages.ContainsKey("Alice")) {
        // …
    }
    ```
  - ```
    for (var kv in ages) {
        var key = kv.Key; // string
        var value = kv.Value; // int
    }
    ```

Strängar

- Strängmetoder (ss. 268ff):
  - ```
    var str = "Hej, Världen!";

    if (str.StartsWith("Hej")) { // true!
       var firstComma = str.indexOf(","); // == 3
       var firstHash  = str.indexOf("#"); // == -1

       string[] parts = str.Split(new [] { ',' });
       // parts = { "Hej", " Världen!");

       parts[1] = parts[1].Trim(); // "Världen!"

       str = str.Replace("Hej", "Hello"); // Hello, Världen!
    }
    ```
    ```

Parprogrammeringsuppgift

1. Skapa en ny Console App (.NET Framework)
   - Skapa en List<string>
   - Programmet ska ta emot en sträng med Console.ReadLine
     - Om strängen var “q” eller “quit”
       - Skriv ut alla strängar som matats in (utom q eller quit)
         - Listan ska sorteras innan den skrivs ut
       - Vänta på en tangentbordstryckning
     - Annars:
       - Lägg till den inmatade strängen i er List<string>
       - Fråga efter nästa sträng och upprepa

```
Lägg till en rad: Kalle
Lägg till en rad: Hej
Lägg till en rad: Hallå
Lägg till en rad: q
Kalle
Hej
Hallå
```

2. Ändra programmet ovan:
   - Efter att man skrivit “q” eller “quit” ska programmet fråga efter ett filter
   - Sedan ska det istället för att skriva ut alla inmatade rader bara skriva ut dem som börjar med filtret

```
Lägg till en rad: abc
Lägg till en rad: cde
Lägg till en rad: abcdef
Lägg till en rad:  abcd
Lägg till en rad: q
Filter: abc
abc
abcdef
```

3. Extraövning:
   - Ändra så att man kan lägga till flera rader samtidigt genom att använda komma: den inmatade raden ska “split”-as med komma till en array, och varje sträng i arrayen läggas till i listan.