Ohjelmointi 1 C#, spring 2013, 19 April 2013

Exam (univ. students) / **2nd "välikoe"** (high school students). **Examiner** Antti-Jussi Lakanen

One sheet of own notes is allowed in this exam.

University students
Answer all of the questions (you do not have to answer the bonus questions). You can get 24 points max (+ possible bonus points). To pass the exam you have to get 12 points.

High school students
Choose **two** questions of your choice and solve them. You can get 12 points max (+ possible bonus points). To pass the exam you need 6 points.

For all
Demo points are counted "on top" of the exam points. You have 4 hours time to finish this exam.

Please find tips and examples of use of methods of C# in the back of this exam. You can apply the information to your answers. Your classes and methods have to be documented as taught in the course. Writing *ComTest*-tests is not obligatory, but writing tests gives bonus points, if that is mentioned in the assignment.

If you need some function/method from .NET-API but you are unable to recall the name of the function/method, write the function declaration and a comment about what it should do. Naturally, you must not make up functions that do not exist, unless you also implement those.

Question 1 (6 p.)

Implement a function documented below. The function should pass the documented *ComTest* test cases. Rewriting the documentation is not needed. However, the function declaration should be written in the answer paper.

```
/// <summary>Function discovers, is it possible to form the given string using given
/// sequence of numbers (which are in a string). The phenomenon is familiar from mobile
/// phones and T9 text input. For the sake of simplicity we use here only capital letters
/// A-Z and only a small subset of other characters.</summary>
/// <param name="merkit">Array that indicates which characters belong "together"</param>
/// <param name="numerot">Sequence containing numbers from 0 to 9.</param>
/// <param name="jono">String</param>
/// <returns>Is it possible to form the given string (jono) from the given number
```

```
/// sequence using the given set of characters.</returns>
/// <example>
/// <pre name="test">
///  String[] merkit = {" ", ".-!?", "ABC", "DEF", "GHI", "JKL", "MNO",
///                     "PQRS", "TUV", "WXYZ"};
///  Merkit.T9(merkit, "26884", "ANTTI")                  === true;
///  Merkit.T9(merkit, "26884", "BOTTI")                  === true;
///  Merkit.T9(merkit, "26884", "DOTTI")                  === false;
///  Merkit.T9(merkit, "268844", "BOTTI")                 === false;
///  Merkit.T9(merkit, "26884", "BOTTII")                 === false;
///  Merkit.T9(merkit, "222", "ABC")                      === true;
///  Merkit.T9(merkit, "26884158774", "ANTTI-JUSSI")      === true;
///  Merkit.T9(merkit, "567520735282260737272555621",
               "KOSKA PELATAAN PESAPALLOA?")             === true;
///  Merkit.T9(merkit, "", "")                            === true;
///  Merkit.T9(merkit, "", "A")                           === false;
/// // (BONUS: 1 p.: if not a number, return false)
///  Merkit.T9(merkit, "a", "A")                          === false;
/// </pre>
/// </example>
public static bool T9(String[] merkit, String numerot, String jono)
{
  // implement the function
}
```

Question 2 (6 p.)

Answer each of the questions below. Write shortly, no more than a few sentences. Each question is worth one point.

1. What are the four different ways to increment a `int` variable with one in C#?
2. What is the difference between the operators `==` and `=` ?
3. What is the value of the variable `n` after the statement
   ```
   double n = (7 % 3) * (4 + 6) / 3;
   ```
   What if the expression was without brackets?
4. What is the `!` operator? Give an example how to use it.
5. What is the `return` clause and what it's used for?
6. Let
   ```
   int a = 3;
   int b = 5;
   ```
   What is the value of the expression:
   ```
   b % a < b - a
   ```

Question 3 (6 p.)

(a) Define *recursion* (1 p.)
(b) Fibonacci sequence is defined as follows:

1. First two numbers in the Fibonacci sequence are 0 and 1
2. Each subsequent number is the sum of the previous two

In other words

$F_0 = 0,$
$F_1 = 1,$
$F_n = F_{n-1} + F_{n-2}$

Write both a recursive function *and* an iterative function (using loops) for counting the n:th number in the Fibonacci sequence. The functions must not print anything. It is assumed that the function is not defined with negative numbers. In addition, we can assume here that the range of `int` type is sufficient. You do not have to write a class -- functions (and documentations) will do. (5 p.)

Question 4 (6 p.)

(a) What this program prints and why. Give reasons. (3 p.)

```
using System;
using System.Text;

public class Tulostus
{
   public static void Main(string[] args)
   {
       StringBuilder sb = new StringBuilder("Antti");
       LisaaLoppuun(sb, "-Jussi");
       sb.Append(" Lakanen");
       Console.WriteLine(sb.ToString());

       String s = "Antti";
       LisaaLoppuun(s, "-Jussi");
       s += " Lakanen";
       Console.WriteLine(s);
```

```
    }

    public static void LisaaLoppuun(String jono, String lisattava)
    {
        jono += lisattava;
    }

    public static void LisaaLoppuun(StringBuilder jono, String lisattava)
    {
        jono.Append(lisattava);
    }
}
```

(b) Implement a function that gives the length of the longest decreasing sequence of the numbers of an `int` array.

Example: The longest decreasing sequence of an array of the integers 5, 3, 4, 4, 2, 0, 1, 2, 3, 3, 2 is (equality is sufficient) 4, 4, 2, 0, and its length is 4. Notice that the function has to return the *length* of the sequence. (3 p.)

Bonus: Proper ComTest tests. (+ 1 p.)

Vinkit / Tips

String.Length Property

Gets the number of characters in the current String object.

Property Value

Type: System.Int32
The number of characters in the current string.

Example

```
// Sample for String.Length
using System;

class Sample
{
    public static void Main()
    {
    string str = "abcdefg";
    Console.WriteLine("1) The length of '{0}' is {1}", str, str.Length);
    Console.WriteLine("2) The length of '{0}' is {1}", "xyz", "xyz".Length);
    }
}
/*
This example produces the following results:
1) The length of 'abcdefg' is 7
2) The length of 'xyz' is 3
*/
```

int.Parse Method (String)

Converts the string representation of a number to its 32-bit signed integer equivalent.

Parameters

s

Type: System.String
A string containing a number to convert.

Return Value

Type: System.Int32
A 32-bit signed integer equivalent to the number contained in s.

String.IndexOf Method (Char)

Reports the zero-based index of the first occurrence of the specified Unicode character in this string.

Parameters

c

Type: System.Char
A Unicode character to seek.

Return Value

Type: System.Int32
The zero-based index position of value if that character is found, or -1 if it is not.

Char.IsDigit Method (Char)

Indicates whether a Unicode character is categorized as a decimal digit.

Parameters

c

Type: System.Char
The Unicode character to evaluate.

Return Value

Type: System.Boolean
`true` if a decimal digit; otherwise, `false`.

Example

```
using System;

public class IsDigitSample {
    public static void Main() {
        char ch = '8';
        Console.WriteLine(Char.IsDigit(ch));                    // Output: "True"
        Console.WriteLine(Char.IsDigit("sample string", 7));    // Output: "False"
    }
}
```