Ohjelmointi 1 C#, kevät 2013, 2. tentti

Tentaattori Antti-Jussi Lakanen

Tässä tentissä saa olla mukana omia muistiinpanoja yhden arkin verran. Tentin valvojalla on oikeus tarvittaessa tarkastaa muistiinpanot (esimerkiksi tilanteessa, jossa tenttijä tekee kerralla enemmän kuin yhden tentin).

Valitse neljä tehtävää ja vastaa niihin (bonuksiin ei ole pakko vastata). Kokeesta voi saada enintään 24 pistettä (+ mahdolliset bonuspisteet). Päästäkseen läpi on saatava 12 pistettä. Demohyvitykset lasketaan tenttipisteiden "päälle", ja sitten katsotaan tuleeko tarvittavat pisteet täyteen. Tenttiaika on 4 tuntia.

Tentin lopussa on vinkkejä ja käyttöesimerkkejä *C#*-kielen metodeista. Voit soveltaa niitä vastauksiisi. Kirjoittamasi luokat ja aliohjelmat pitää dokumentoida, kuten kurssilla on opetettu. *ComTest*-testien kirjoittaminen ei ole pakollista, mutta siitä voi saada lisäpisteitä, mikäli tehtävässä on niin sanottu.

Jos vastauksessa tarvitset jotakin funktiota .NET-APIsta (.NET-kirjasto) etkä muista tarkkaan mikä oli funktion/metodin nimi, niin kirjoita funktion esittely parametreineen ja kommentti siitä, mitä sen pitäisi tehdä. Luonnollisesti et saa itse keksiä funktiota, joita ei ole olemassa (jos itse myös toteuta niitä).

Tehtävä 1 (6 p.)

Tässä tehtävässä on kolme osaa: a, b ja c. Tee alla olevaa dokumentaatiota vastaava ja ComTest-testitapaukset täyttävä funktio (tehtävän c-osa). Funktiota varten kirjoita kaksi apufunktiota (tehtävän a- ja b-osat). Dokumentaatioita ei tarvitse kirjoittaa uudestaan. Kirjoita kuitenkin funktioiden esittelyrivit vastauspaperiin.

```
/// <summary> Antaa uuden kellonajan siten että vanhaan aikaan
/// (aika-parametri) lisätään jokin aika (lisays-parametri). Huomaa, että käytössä
/// on 24 tunnin kello.</summary>
/// <param name="aika">Alkuperäinen kellonaika (muodossa hh:mm)</param>
/// <param name="lisays">Lisäys (muodossa hh:mm)</param>
/// <returns>Uusi kellonaika (muodossa hh:mm)</returns>
/// <example>
/// <pre name="test">
///  Aika.LisaaAikaa("16:01", "00:05") === "16:06";
///  Aika.LisaaAikaa("16:01", "00:00") === "16:01";
///  Aika.LisaaAikaa("16:59", "00:11") === "17:10";
```

```
///  Aika.LisaaAikaa("00:01", "01:01") === "01:02";
///  Aika.LisaaAikaa("12:00", "00:00") === "12:00";
///  Aika.LisaaAikaa("00:00", "00:01") === "00:01";
///  Aika.LisaaAikaa("00:00", "00:00") === "00:00";
/// </pre>
/// </example>
public static String LisaaAikaa(String aika, String lisays)
{
 // Toteuta funktio, tämä on tehtävän c-osa
}
```

(a) Tee funktio `MinuutitAjaksi`, joka muuttaa annetun minuuttimäärän vastaavaa kellonaikaa esittäväksi merkkijonoksi. Alla dokumentaatio ja Comtest-testit. (2 p.)

```
/// <summary>Muuttaa annetut minuutit kellonajaksi.</summary>
/// <param name="minuutit">Aika minuutteina (kokonaisluku).</param>
/// <returns>Kellonaika merkkijonona.</returns>
/// <example>
/// <pre name="test">
///  Tentti.MinuutitAjaksi(1) === "00:01";
///  Tentti.MinuutitAjaksi(240) === "04:00";
///  Tentti.MinuutitAjaksi(780) === "13:00";
///  Tentti.MinuutitAjaksi(1439) === "23:59";
///  Tentti.MinuutitAjaksi(1440) === "00:00"; // Jos kaksi viimeistä testiä
///  Tentti.MinuutitAjaksi(1441) === "00:01"; // menee läpi niin BONUS +1 p.
/// </pre>
/// </example>
public static String MinuutitAjaksi(int minuutit)
{
  // toteuta funktio
}
```

Vinkki: Tarvitset tässä modulo-operaattoria (`%`). Ks. vinkit tentin takaosassa.
Vinkki 2: Kahden `int`-luvun jakolaskun tulos on kokonaisluku.

(b) Tee funktio `AikaMinuuteiksi`, joka muuttaa annetun kellonaikaa esittävän merkkijonon minuuteiksi. Alla dokumentaatio ja Comtest-testit. (2 p.)

```
/// <summary>Muuttaa merkkijonona annetun ajan (hh:mm) minuuteiksi.
/// Käytetään edelleen 24 tunnin kelloa. </summary>
/// <param name="aika">Aika merkkijonona.</param>
/// <returns>Aika minuutteina.</returns>
/// <example>
/// <pre name="test">
///  Tentti.AikaMinuuteiksi("00:01") === 1;
```

```
///  Tentti.AikaMinuuteiksi("04:00") === 240;
///  Tentti.AikaMinuuteiksi("23:59") === 1439;
///  Tentti.AikaMinuuteiksi("00:00") === 0;
/// </pre>
/// </example>
public static int AikaMinuuteiksi(String aika)
{
  // toteuta funktio
}
```

Vinkki: Paloittele merkkijono `Split`-metodilla, ja lue "palojen" luvut `int.Parse`-metodilla.

(c) Toteuta nyt `LisaaAikaa`-funktio käyttäen a- ja b-kohdissa tekemiäsi funktioita. (2 p.)

Tehtävä 2 (6 p.)

Alla on kolme kohtaa, joista jokaisesta saa enintään kaksi pistettä. Vastaa enintään muutamalla virkkeellä.

1. Kurssilla käyetyt C#:n silmukkarakenteet ja mitä eroa niillä on? Mainitse vähintään kolme erilaista rakennetta.
2. Laske välivaiheineen yhteen luvut 3 ja -3 kun oletetaan että negatiivisille luvuille on käytössä 2-komplementti ja tilaa käytössä 4 bittiä.
3. Esittele vähintään kolme aritmeettista operaattoria ja kerro mitä kukin tekee.

Tehtävä 3 (6 p.)

Tee aliohjelma, joka poistaa annetusta merkkijonolistasta kaikki ne jonot, jotka sisältävät parametrina annetun merkkijonon. Saat käyttää esimerkiksi `List<T>`-tyypin valmista metodia poistamiseen (ks. vinkit). Aliohjelmaa voisi kutsua seuraavasti.

```
List<String> jonot = new List<String>(){"Antti", "Jussi", "Viljo", "Aati", "Henna"};
Tentti.PoistaSisaltavat(jonot, "tti");
// jonot-lista sisältäisi nyt alkiot "Jussi", "Viljo", "Aati", "Henna"
```

Asiaankuuluvat ComTest-testit +1p.

Tehtävä 4 (6 p.)

Tee funktio nimeltä `KaannaTaulukko`, joka palauttaa parametrina annetun kokonaislukutaulukon

päinvastaisessa järjestyksessä. Alkuperäistä taulukkoa ei saa muuttaa, eikä funktio saa tulostaa mitään. Funktiota voidaan kutsua esimerkiksi seuraavasti:

```
public static void Main(String[] args)
{
    int[] luvut = { 5, 9, 3, 7, 8 };
    int[] luvutKaannetty = KaannaTaulukko(luvut);
}
```

Tällöin luvutKaannetty-taulukon sisältö olisi { 8, 7, 3, 9, 5 }. `Array.Reverse()`-metodin käyttö on kielletty. Edelleen tarpeettomasta olioiden luomisesta tulee miinuspisteitä.

Tehtävä 5 (6 p.)

Vastaa sekä a- että b-kohtiin.

(a) Mitä on kommentointi ja mitä eri tapoja kommentoinnille on? Miksi koodia kannattaa kommentoida? (2 p.)

(b) Kirjoita dokumentaatio seuraavaan funktioon. (4 p.) Bonus: Kattavat ComTest-testit + 1p.

```
public static char MuutaKirjain(char kirjain)
{
  String mitka = "ääö";
  String miksi = "aao";
  char muutettuKirjain = Char.ToLower(kirjain);
  int i = mitka.IndexOf(muutettuKirjain);
  if (i >= 0) return miksi[i];
  if (muutettuKirjain < 'a') return '1';
  if (muutettuKirjain > 'z') return '1';
  return muutettuKirjain;
}
```

Vinkit / Tips

String.Contains Method

Returns a value indicating whether the specified String object occurs within this string.

Syntax

```
public bool Contains(
```

```
        string value
)
```

Parameters

value

Type: System.String
The string to seek.

Return value

Type: System.Boolean
true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

Example

```
public class Sample
{
    public static void Main()
    {
    string s1 = "The quick brown fox jumps over the lazy dog";
    string s2 = "fox";
    bool b;
    b = s1.Contains(s2);
    Console.WriteLine("Is the string, s2, in the string, s1?: {0}", b);
    }
}
/* Output:
   Is the string, s2, in the string, s1?: True
*/
```

String.Split(Char[])

`Split`-metodia voidaan käyttää merkkijonojen paloitteluun. `Split`-metodille annetaan parametrina merkki (tai merkit), jonka perusteella erottelu tehdään.

Example

```
String jono = "ab,cd";
String[] osat = jono.Split(','); // nyt: osat[0] == "ab", osat[1] == "cd"
```

int.Parse Method (String)

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

Syntax

```
public static int Parse(
       string s
)
```

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.

List<T>.RemoveAt Method

Removes the element at the specified index of the List<T>.

Syntax

```
public void RemoveAt(
       int index
)
```

Parameters

index

Type: System.Int32
The zero-based index of the element to remove.

% Operator

The % operator computes the remainder after dividing its first operand by its second.

Example

```
public class Tentti
{
    public static void Main(string[] args)
    {
        Console.WriteLine(5 % 2);   // output: 1
        Console.WriteLine(4 % 2);   // output: 0
        Console.WriteLine(11 % 7);  // output: 4
        Console.WriteLine(11 % 11); // output: 0
    }
}
```