IPAR 4.0 .NET C# ALAPOKON

1. forduló

SEMILAB

A kategória támogatója: Semilab Zrt.

Ismertető a feladathoz

Kérjük, hogy a feladatlap indítása előtt mindenképp olvasd el az alábbi útmutatót:

- MINDEN kérdésre **van helyes válasz**.
- Olyan kérdés **NINCS**, amire az összes válasz helyes, ha mégis az összes választ bejelölöd, arra a feladatra automatikusan 0 pont jár.
- A **radio button-os kérdésekre egy helyes válasz van**.
- **Ha lejár a feladatlap ideje, a rendszer AUTOMATIKUSAN** beküldi azt az addig megjelölt válaszokkal.
- Azokat a feladatlapokat, amelyekhez **csatolmány** tartozik, javasoljuk **NEM mobilon** elindítani, erre az érintett feladatlapok előtt külön felhívjuk a figyelmet.
- Az **adatbekérős feladatokra NEM jár részpontszám**, csak a feleletválasztósakra.
- **Helyezéseket a 4. forduló után mutatunk**, százalékos formában: adott kategóriában a TOP 20-40-60%-hoz tartozol.
- **Badge-ket** szintén a 4.forduló után kapsz majd először.
- Ha egyszerre több böngészőből, több ablakban vagy több eszközről megnyitod ugyanazt a feladatlapot, **nem tudjuk vállalni** az adatmentéssel kapcsolatban esetlegesen felmerülő anomáliákért a felelősséget!
- A hét forduló során az egyes kategóriákban (de nem feltétlenül mindegyikben) **könnyű-közepes-nehéz kérdésekkel** egyaránt találkozhatsz majd.

Jó versenyzést kívánunk!

1.forduló

A feladatlap több csatolmányt is tartalmaz, ezért a megoldását asztali gépen javasoljuk!

Fontos!

Fordulónként javasoljuk az összes részfeladat végigolvasását a kidolgozás megkezdése előtt, mivel a feladatok sokszor egymásra épülnek. Előfordul, hogy egy részfeladat nehézségét az input mérete adja, így érdemes hatékony megoldásokra törekedni.

Felhasznált idő: 00:00/40:00 **Elért pontszám:** 0/12

Indítás előtti csatolmányok Indítás utáni csatolmányok

1. feladat 0/2 pont

Mit ír ki a következő kódrészlet a képernyőre?

```
static void Main(string[] args)
{
    var numbers = new List<int> { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
    var filtered = numbers.Where(i => i % 2 == 0);
    for (int i = 0; i < numbers.Count; i++)
    {
        if (numbers[i] % 2 == 1)
        {
            numbers[i]++;
        }
    }
    Console.WriteLine(filtered.Max());
}
```

Válasz

A helyes válasz:

10

Magyarázat

A filtered változón a where csak akkor fog kiértékelődni, amikor a filtered értékeire hivatkozunk. Éppen ezért mindegy, hogy a where előtt, vagy után változtatjuk meg a hivatkozott adatokat, mivel a where kiértékelése csak a Max() hívásakor fog megtörténni.

2. feladat 0/4 pont

Mérőgépeink sztring azonosítókat rendelnek az egyes mintákhoz. Helytakarékossági okokból szükségünk van ezen azonosítók egy lerövidített változatára, melyet úgy állítunk elő, hogy a lehető legtöbb karaktert töröljük az azonosító végéről úgy, hogy az még egyedi maradjon. Akkor egyedi egy rövidített azonosító, ha egyetlen másik minta eredeti azonosítójának sem kezdő karaktersorozata. A tömbben nincs olyan azonosító, amelyet teljes egészében tartalmazna egy másik tömbben található azonosító.

Adjuk meg a rövidített sztringekből álló tömb összkarakterszámát!

Példa

Input:

RMN_1350_ST

RMN_1360_ST

RMN_1370_BRIGHT

RMN_1370_DARK

Rövidített alak:

RMN_135

RMN_136

RMN_1370_B

RMN_1370_D

Kimenet:

34

(A rövidített sztringek összkarakterszáma: 7 + 7 + 10 + 10 = 34)

A 1_2_teszt.txt eredménye: 193346

Adjuk meg a 1_2.txt input eredményét.

Megjegyzés:

Ha a megoldásához sztring összehasonlító függvényeket használunk, figyeljünk oda a megfelelő CultureInfo paraméter választására!

Válasz

A helyes válasz:

188789

Magyarázat

Kicsi input esetén egy brute force algoritmus megadása elegendő.

3. feladat 0/6 pont

A második feladathoz hasonlóan adjuk meg a rövidített sztringekből álló tömb összkarakterszámát!

Adjuk meg a 1_3.txt input eredményét!

Válasz

A helyes válasz:

19247224

Magyarázat

A megoldásunk rekurzióból és GroupBy hívásokból áll. A GetLength rekurzív fv visszatérési értéke a paraméterként átadott lista megoldása.

A GetLength második paramétere az aktuálisan vizsgált karakterek indexe, amely nullától indul, és folyamatosan növekszik 1-el.

A GetLength belsejében grouppolunk az aktuális indexű karakterekre, majd az így keletkezett csoportokra rekurzívan meghívjuk a GetLength-et eggyel növelve az indexet, majd a kapott eredményeket szummázzuk.

Amikor a kapott lista már csak 1 elemű, az azt jelenti, hogy az index-edik karakterrel bezárólag egyedi volt a sztringünk, ezért hossznak visszadjuk az indexet.

```
public static int Solve(string input)
{
    var strings = File.ReadAllLines(input);
    var sbStrings = strings.Select(s => new StringBuilder(s)).ToList();
```

```
    var sum = GetLength(sbStrings, 0);
    return sum;
}


static int GetLength(List<StringBuilder> sbStrings, int index)
{
    if (sbStrings.Count == 1)
    {
        return index;
    }
    var groups = sbStrings.GroupBy(sb => sb[index]);
    var sum = 0;
    foreach (var group in groups)
    {
        sum += GetLength(group.ToList(), index + 1);
    }
    return sum;
}
```