**Megoldás sablon C. Beadandó Házi feladat.**

Név: Golyha Gergő

Neptunkód: A7MMZ1

A képen szöveg, képernyőkép, Betűtípus, dokumentum látható

Automatikusan generált leírásFeldat: Gazdaságos üzemeltetés

1. Részteljesítés max: 50 pont

|  |  |
| --- | --- |
| Specifikáció | max: 20 pont |
| [Link](https://progalap.elte.hu/specifikacio/?data=H4sIAAAAAAAACt1QwU7CQBT8lcmeqGlJWwTNJpBoIheBC3KxcFhhMZusxbDbhki48yt%2Bh3%2Fil%2FheayIQondPO5N5897sbIV71XOzNHPlzSoXUtxqCbV4%2BtzvRyGK%2BinrR1lLYOKVy5Jmk4ZmIZihi4HGBn1tQ0Zd0DARBtP1NL83Es%2FqbUHmAfOxknC8kkm%2FlChXufKrXGIU9fhcTRubgDaMHyfD4U3DdOnkpsqQmVlzqW30Da0OeM%2FdUiJiMCHgToyUNUp%2BFpsgwMe7q0LRpMMFCvTQqMYCYqUIhdfOOyGzrVgor6gZUiU6fKKQSOKYUVmhClIcyS8QwVKJMWoGUFgaax%2BKyYnain%2BxttJD8fL6WD1ypifW9OpI7ZyxcgkSD%2BtCM3OUrZPS93P1ounThNbaFdYLmezC8120%2F1UTfWXdX1XEu9nuC2EFh307AwAA)  Be: adb∈N, u∈N, v∈N, all∈Utas[1..adb], Utas = Le x Fel, Le = N, Fel = N  Ki: gazd∈L  Sa: s∈N  Fv: vonaton: N->N, vonaton(x) = SZUMMA(i=1..x, all[i].fel-all[i].le)  Ef: -  Uf: s = SZUMMA(i=1..adb-1, vonaton(i)) és gazd = s \* u > (adb-1) \* v | Itt megalkotva, vagy másolva más környezetből vagy kézzel írt megoldás fényképe |
| Sablon + forrásmegjelölés | max: 5 pont |
| Összegzés (mind a kettő)  programozási minták.pdf / 4. oldal ([canvas](https://canvas.elte.hu/courses/38943/files/2451951?module_item_id=587641)) | Lehet képernyőkép vagy másolat az előadás anyagából.  Forrásmegjelölés: a forrásfájl neve (publikálás útvonala) és oldalszám/dia sorszáma. |
| Visszavezetési táblázat (megfeleltetés) | max: 5 pont |
| Összegzés (fő) Összegzés (vonaton)  e..u ~ 1..adb-1 e..u ~ 1..x  f(i) ~ vonaton(i) f(i) ~ all[i].fel-all[i].le | A specifikáció és a minta közötti kapcsolat leírása |
| Algoritmus | max: 20 pont |
|  | Itt megalkotva vagy képernyőkép vagy kézzel írt megoldás fényképe |

1. Részteljesítés max 50 pont

Módosítások max 10 pont

Itt kell dokumentálni, ha a beadott tervet a visszajelzés vagy a kódolás és tesztelés során kiderülő részletek miatt módosítani kell. A dokumentáció tartalmazza

|  |  |
| --- | --- |
| Uf: s = SZUMMA(i=1..adb-1, vonaton(i)) és gazd = s \* u > adb-1 \* v | A biro-ban 100-at kaptam, ha adb-ot nem csökkentettem, de találtam olyan esetet, ahol helytelen az eredmény.  Mivel adb az állomások száma, a vonat adb-1 állomásnyi utat tett meg, ezért a csökkentett értékkel kell számolni.  A biro-ba lehet érdemes egy olyan tesztesetet beletenni, mint pl nálam a harmadik (a biro elfogadta a kódot (-1) nélkül is de itt nem működne). |

(A módosítások jellegétől függően, lehet szövegdoboz nyilakkal vagy lábjegyzet is az indoklás.)

Maximális pontszámot ér az összes módosítás dokumentálása. Ha nem volt módosítás és a kód a korábban beadott tervet képezi le, akkor megkapható a maximális pontszám (nem a mennyiség, hanem a minőség számít.) Ebben az esetben is szerepeljen itt az első részteljesítés.

Kód forrása max 11 pont

|  |  |
| --- | --- |
| biro.elte.hu téma: **Tételek összeépítése** feladat: **Gazdaságos üzemeltetés** beadás (feltöltés) sorszáma: **4** pontszám: **100** | Break, exit vagy try-catch használata esetén a 11 pontból legfeljebb 1 pont adható. |
| Ha volt: hibás tesztesetek sorszáma és a mellettük megjelenő hibaüzenetek | A hibaüzenetek és a kód függvényében, akár 80%-os eredmény is lehet maximális pontszámú |

Kód max 26 pont

2-2-2 pont: megjegyzésbe írt információk, külön választott deklarálás és kiírás.

8 pont a külön választott beolvasás/adatok eltárolása Ezen belül break, exit vagy try-catch használata esetén a 8 pontból 0 pont adható.

12 pont az algoritmus szabványos kódolása. Ezen belül break, exit vagy try-catch használata esetén a 26 pontból 0 pont adható.

using System;

// Golyha Gergő

// A7MMZ1

// golhyagergo@gmail.com

// Tételek összeépíése: Gazdaságos üzemeltetés

namespace beadando2

{

internal class Program

{

struct Utas

{

public int le;

public int fel;

}

static void Main(string[] args)

{

(int u, int v, Utas[] all) = beolvasas();

bool gazd = feldolgozas(u, v, all);

kiiras(gazd);

}

static (int u, int v, Utas[] all) beolvasas()

{

int u, v;

int n;

Utas[] all;

Console.Error.Write("n = ");

// Ef: 1 <= n <= 100

int.TryParse(Console.ReadLine(), out n);

all = new Utas[n];

Console.Error.Write("u = , v = ");

string[] be1 = Console.ReadLine().Split();

int.TryParse(be1[0], out u);

int.TryParse(be1[1], out v);

// Ef: 0 <= u <= 100

// 0 <= v <= 100000

for (int i = 0; i < n; i++)

{

Console.Error.Write("{0}. állomáson le = , fel = ", i + 1);

string[] be2 = Console.ReadLine().Split();

int.TryParse(be2[0], out all[i].le);

int.TryParse(be2[1], out all[i].fel);

// Ef: 0 <= le <= 800

// 0 <= fel <= 800

}

return (u, v, all);

}

static bool feldolgozas(int u, int v, Utas[] all)

{

int vonaton(int x)

{

int s = 0;

for (int i = 0; i <= x; i++)

{

s += all[i].fel - all[i].le;

}

return s;

}

bool gazd;

int s = 0;

for (int i = 0; i < all.Length - 1; i++)

{

s += vonaton(i);

}

return s \* u > (all.Length - 1) \* v;

}

static void kiiras(bool gazd)

{

Console.Error.WriteLine("Az üzemeltetés {0} gazdaságos", gazd ? "" : "nem");

Console.WriteLine(gazd ? 1 : 0);

}

}

}

Teszt max 3 pont

A feladathoz adott 2 teszten túl még 3 teszt.

(A tervmódosítást alátámasztó tesztek, speciális esetek tesztjei.)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Saját 1. | Saját 2. | Saját 3 |
| Bemenet: | 2  100 100  0 1  1 0 | 5  50 1000  0 100  10 50  31 0  15 7  0 101 | 3  100 99  0 1  0 0  1 0 |
| Kimenet: | 0 | 1 | 1 |