NEVEZETES ALGORITMUSOK

GYAKORLÓ FELADATOK C#-BAN

TARTALOM

SOROZATSZÁMÍTÁS

A feladatokban az a közös, hogy **a kiszámítandó értéket** az egész sorozaton értelmezett **függvény adja meg** (pl. átlag, összeg, szorzat, unió, metszet, egymás után írás, stb.).

Segítségével ilyen (és ehhez hasonló) kérdésekre kapunk választ:

- Határozzuk meg a sorozat elemeinek összegét/átlagát/szorzatát!
- Mennyi az első N természetes szám összege?
- Mennyi az N! (N faktoriális) értéke?
- Mennyi az osztály legutóbbi informatika dolgozatainak átlaga?

GYAKORLÓ FELADATOK

A megoldás során készítsünk minden feladathoz külön eljárást, azokat hívjuk meg a főprogramból!

1. ÁTLAGHŐMÉRSÉKLET

Egy héten át feljegyeztük a napi átlaghőmérsékletet. Határozzuk meg a heti átlaghőmérsékletet! A napi átlaghőmérsékletek valós számok, a billentyűzetről írjuk be az értékeket. A számokat nem kell eltárolni.

Megjegyzés: az átlagot kerekítsük két tizedes jegyre. Segítség: az **atlag** nevű változóban lévő érték két tizedesre kerekítése: Math.Round(atlag,2).

2. EGÉSZ SOROZAT

Kérjünk be a billentyűzetről 5 egész számot! A számokat nem kell eltárolni. Adjunk választ az alábbi kérdésekre:

a., Mennyi a számok összege?
b., Mennyi a számok átlaga? (két tizedes jegyre kerekítve írassuk ki)
c., Mennyi a számok szorzata?

3. KOLDUS BEVÉTELE

Egy koldus kapott adományaiból határozzuk meg összbevételét, valamint azt, hogy átlagosan mennyit adtak neki egy-egy alkalommal. A bevételek beolvasása a billentyűzetről történjen addig, amíg a nulla (0) végjelet le nem ütjük. A bevételeket nem kell tárolni, a megoldáshoz hátultesztelő ciklust használjunk!

4. TESTMAGASSÁGOK

Ismerjük egy kosárlabda csapat játékosainak a magasságát (a testmagasságot méterben tároljuk, előfordulhat tört érték is). Ezen értékek ismeretében adjuk meg a játékosok átlagmagasságát! A játékosok száma 10, adataikat egy tömbben tároljuk (nem kérjük be a billentyűzetről). A megoldáshoz számláló ciklust használjunk!

A megoldáshoz egészítsük ki az alábbi kódrészletet:

```
            // Testmagasságok

            // tömb - deklarálás és értékadás
            double[] magassagok; // típus megadása: valós, egydimenziós (=vektor)
            magassagok = new double[10] { 2.10, 2, 1.95, 1.9, 2.12, 2.05, 1.85, 1.95, 1.92, 2 }; //
helyfoglalás a memóriában: 10 elem, értékadás

            // segédváltozók inicializálása

            // testmagasságok összegének meghatározása

            for (int i = 0; i < length; i++)
            {
                // az aktuális tömbelem kiírása
                // magasságok összegének kiszámítása
            }

            // átlag meghatározása
            // átlag kiszámítása
            // formázás 2 tizedesre

            // eredmény kiíratása
```

KIEGÉSZÍTÉS

Ha a program hibátlanul működik, az aktuális tömbelem kiíratását módosítsuk (táblázat-szerűen) formázott kiíratásra. Így nézzen ki a program kimenete:

```
Testmagasságok program

A játékosok testmagasságát egy tömbben tároljuk. Az értékek:
   1. játékos:   2,1 m
   2. játékos:     2 m
   3. játékos:  1,95 m
   4. játékos:   1,9 m
   5. játékos:  2,12 m
   6. játékos:  2,05 m
   7. játékos:  1,85 m
   8. játékos:  1,95 m
   9. játékos:  1,92 m
  10. játékos:     2 m

A testmagasságok átlaga: 1,98 m.
```

5. CSOPORTÁTLAG

Vigyük fel a billentyűzetről egy JEGYEK nevű tömbbe a legutóbbi informatika dolgozat eredményeit (diákonként), majd írassuk ki, mennyi a csoportátlag. A tömb mérete fix legyen, melyet a csoportlétszám határozzon meg!

A program egy kimenete lehet az alábbi:

```
Csoportátlag program

A jegyek felvitele következik. Add meg sorban a kért értékeket!
1. jegy: 5
2. jegy: 4
3. jegy: 3
4. jegy: 2
5. jegy: 1
6. jegy: 5
7. jegy: 4

A jegyek átlaga: 2,4
```

KIEGÉSZÍTÉS

Ha a program hibátlanul működik, oldjuk meg, hogy csak egész számokat fogadjon el 1-5-ig. Amennyiben elrontjuk a jegyet, tájékoztasson az elkövetett hiba típusáról (nem egész szám/ nem az intervallumba esik), és addig kérje újra, amíg a kívánt tulajdonságoknak megfelel. Segítség: a megvalósításhoz hátultesztelő ciklust érdemes használni.

A kiegészítés után a program egy lehetséges kimenete:

```
Csoportátlag program

A jegyek felvitele következik. Add meg sorban a kért értékeket!

1. jegy: 5
Ez rendben van.

2. jegy: kettő
Hibás érték, kérlek, egész számot írj be! 2
Ez rendben van.

3. jegy: 9
Ez a szám nem lehet érdemjegy (1 és 5 között írj be értéket)! 4
Ez rendben van.

4. jegy: 2
Ez rendben van.

5. jegy: 5
Ez rendben van.

6. jegy: 4
Ez rendben van.

7. jegy: 5
Ez rendben van.

A jegyek átlaga: 2,7
```

6. VÉLETLEN SZORZAT

Kérjünk be egy egész számot, majd töltsünk fel ennyi véletlen (egész) számmal egy tömböt. A véletlenek 0-10 között lehetnek. Írassuk ki a számokat, s adjuk meg azok összegét, átlagát és szorzatát.

```
Véletlen szorzat program

Mennyi véletlen számot generáljon a program? 3

3 elemű tömb feltöltése véletlen számokkal (1-10 között):
veletlenek[1] = 10
veletlenek[2] = 8
veletlenek[3] = 2

Összeg = 20
Átlag = 6,67
Szorzat = 160
```

7. PÉNZTÁRBLOKK ELLENŐRZÉSE

Tegyük fel, hogy bevásárlás után szeretnénk a pénztárblokk adatait ellenőrizni. A billentyűzetről sorban begépeljük a vásárolt termékek árait, és szeretnénk ellenőrzésként kiszámítani a végösszeget. Az adatokat nem tároljuk, csupán 1x beolvassuk, majd rögtön feldolgozzuk. A blokk tételeinek száma előre nem ismert (nem számoljuk meg, mennyi sor szerepel rajta). Az elemszám begépelés közben derül ki. A 0 leütésekor ér véget a gépelés.

```
Pénztárblokk ellenőrzése program

Kérem a blokkon szereplő összegeket (Enterrel az összeg után)!
Ha "0", akkor vége az adatbevitelnek!

500
10
0

A tételek összege = 510
```

8. BEKÉRÉS ADOTT ÖSSZEG ELÉRÉSÉIG

Készítsünk programot, mely pozitív egész számokat kér be mindaddig, amíg azok összege el nem éri (vagy meg nem haladja) a százat! A beolvasáskor nem ellenőrizzük az adatok megfelelőségét, a számokat nem kell tárolni. A beolvasáshoz használjunk elöltesztelő ciklust.

```
Bekérés adott összeg eléréséig program

Kérem a számokat (Enterrel a végén)!
Ha összegük elérte a százat, vége az adatbevitelnek!

50
25
20
35
Az utolsó számot (35) már nem adjuk az összeghez.

A számok összege (az utolsó 35 nélkül) = 95.
```

KIEGÉSZÍTÉS

Ha készen vagyunk, módosítsuk a fenti programot úgy, hogy a program csak akkor adja hozzá a beírt számot az addigi összeghez, ha az nagyobb, mint az előzőleg hozzáadott szám. Ekkor írja ki az új összeget. Ellenkező esetben tájékoztasson róla, hogy a szám nem nagyobb, mint az előzőleg hozzáadott szám.

Kiegészítés után a program egy kimenete lehet:

```
Bekérés adott összeg eléréséig program - kiegészített

Kérem a számokat (Enterrel a végén)!
Ha összegük elérte a százat, vége az adatbevitelnek!

10
20
30
25
Ezt a számot: 25 nem adjuk az összeghez, mert nem nagyobb ennél: 30.
35
40
Az utolsó számot: 40 már nem adjuk az összeghez,
mert az összeg meghaladná a 100-at.

A számok összege (a kihagyottak nélkül) = 95.
```

9. KARAKTERFÜZÉR

Egy N elemű karaktersorozat betűit fűzzük össze egyetlen szöveg típusú változóba, majd írassuk a képernyőre. N értékét a felhasználótól kérdezzük meg, a beírt karaktereket beolvasás után egy tömbben tároljuk. Az összefűzött karakterlánc kiíratása után kérdezzük meg a felhasználót, hogy hányadik karaktert szeretné kiíratni, majd írassuk ki a tömbből a kért sorszámú karaktert. Amennyiben rossz számot (nem létező indexet) adott meg, írassuk ki, hogy nincs ilyen elem.

A program egy-egy kimenete:

```
Karakterfüzér program

N = 4

karakterek[0] = t
karakterek[1] = ö
karakterek[2] = m
karakterek[3] = b

A karakterfüzér tartalma: tömb

Hányadik karaktert kéred? 1
karakterek[1] = ö
```

```
Karakterfüzér program

N = 4

karakterek[0] = t
karakterek[1] = ö
karakterek[2] = m
karakterek[3] = b

A karakterfüzér tartalma: tömb

Hányadik karaktert kéred? 5
Nincs 5. indexű elem a tömbben.
```

ELDÖNTÉS

Azt vizsgáljuk, hogy egy sorozatnak **van-e adott tulajdonságú eleme** (pl. páros, nemnegatív, osztható vmivel, hosszabb x karakternél, van benne „e” betű, stb.). A sorozathoz rendelt érték ebben az esetben egy logikai érték (igaz-hamis vagy igen-nem). A sorozat minden eleméről egyértelműen eldönthető, hogy megfelel-e az elvárt tulajdonságnak.

Ez tipikusan az a tétel, amelyben **nem célszerű a sorozat összes elemét vizsgálni**. Elegendő csak addig keresnünk adott tulajdonságú elemet, míg a legelsőt meg nem találjuk, hiszen nem kérdés, hogy hány darab ilyen tulajdonságú elem van. Ha a sorozat legelső eleme adott tulajdonságú, megállhatunk, és rögtön megállapíthatjuk, hogy igen, van ilyen elem.

A legkedvezőtlenebb eset az lehet, ha a sorozat valamennyi elemét megvizsgáltuk, de egyik sem volt adott tulajdonságú. Ezt az jelzi, hogy az **i** ciklusváltozó értéke **N** lesz (tömb, string esetén **N-1**).

A feladatok fentiek szerint két csoportba sorolhatók:

a., Azt kell eldönteni, hogy a sorozatban **létezik-e adott tulajdonságú elem**. (Ilyenkor a sorozat elemeit vizsgáljuk az elejéről, de csak addig, míg megtaláljuk az első megfelelő elemet. Ha túljutunk a sorozat végén, akkor nincs ilyen elem.)

b., Azt kell eldönteni, hogy a sorozatban **minden elem adott tulajdonságú-e**.

GYAKORLÓ FELADATOK

10. KÉRDŐJEL

Döntsük el egy szövegről, hogy van-e benne kérdőjel! A feldolgozáshoz elöltesztelő ciklust használjunk.

11. MELEGEDÉS

A januári napi középhőmérsékletekről döntsük el, hogy emelkednek-e! A hőmérséklet adatokat Celsius fokban tároljuk egy vektorban (nem kell beolvasni). A teszteléshez érdemes kiíratni a hőmérsékleteket, így látjuk, hol tart a feldolgozás.

Egy lehetséges kimenet:

```
Melegedés program

A januári hőmérsékleteket egy tömbben tároljuk. Az értékek:
Január 1.: -5 Celsius
Január 2.: -4,9 Celsius
Január 3.: -2 Celsius
Január 4.: -1 Celsius
Január 5.: -0,5 Celsius
Január 6.: -1 Celsius

Az adott hónapban nem emelkedett folyamatosan a napi átlaghőmérséklet.
```

12. HÓNAP-E

Döntsük el egy szóról a hónapnevek sorozata alapján, hogy egy hónap neve-e! A hónapneveket egy vektorban tároljuk. Ügyeljünk arra, hogy ne számítson hibának az, ha a felhasználó a kis- és nagybetűket vegyesen használja a begépeléskor.

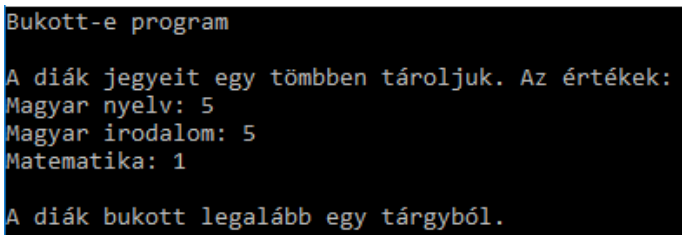
A program lehetséges kimenete:

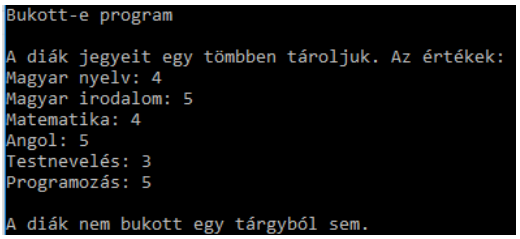
```
Hónap-e program

Kérem a hónapnevet (kis- és nagybetű nem számít): márCiUS

A beírt érték hónapnév.
```

13. BUKOTT-E

Döntsük el egy tanuló év végi osztályzatai alapján, hogy bukott-e valamilyen tantárgyból! Készítsünk két vektort: az egyikben a tantárgyak nevét tároljuk (tantargyak), a másikban jegyeit számként (jegyek) olyan sorrendben, ahogyan a tantárgyakat felvettük (az adatokat a programkódban tároljuk, nem kell beolvasni). Feldolgozás során írjuk ki az aktuálisan feldolgozás alatt álló tárgyat és az osztályzatát (a kiíratás akkor érjen véget, amikor találtunk egy elégtelen osztályzatot).

A program kimenete:

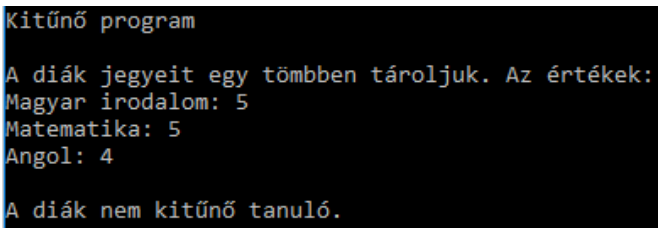


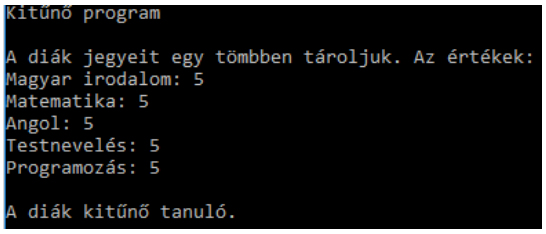


14. KITŰNŐ

Döntsük el egy tanuló év végi osztályzatai alapján, hogy kitűnő tanuló-e! Készítsünk két vektort: az egyikben a tantárgyak nevét tároljuk (tantargyak), a másikban jegyeit számként (jegyek) olyan sorrendben, ahogyan a tantárgyakat felvettük (az adatokat a programkódban tároljuk, nem kell beolvasni). Feldolgozás során írjuk ki az aktuálisan feldolgozás alatt álló tárgyat és az osztályzatát (a kiíratás akkor érjen véget, amikor találtunk egy nem jeles osztályzatot).

A program egy kimenete lehet:





15. PÁRATLAN

Egy 10 elemű egész számokból álló sorozatról (vektorban tároljuk) állapítsuk meg a következőket:

a., Van-e közöttük páratlan szám?
b., Minden szám páratlan-e?
c., Számtani sorozat? (Segítség: a számtani sorozat egymást követő elemeinek különbsége állandó.)

A megoldás előtt írassuk ki a sorozat összes elemét egy sorba vesszővel elválasztva. Minden részfeladathoz külön ciklust használjunk, és írassuk ki, melyik feladat következik!

A program egy lehetséges kimenete:

```
Páratlan program

A számokat egy tömbben tároljuk. Az értékek:
-10, -7, -4, -1, 2, 5, 8, 11, 14, 17,

Van-e a sorozatban páratlan szám?
-10, -7,
A sorozatnak van páratlan eleme.

Minden szám páratlan?
-10,
A sorozatnak nem minden eleme páratlan.

Számtani sorozat?
-4, -1, 2, 5, 8, 11, 14, 17,
Az elemek számtani sorozatot alkotnak.
```

KIVÁLASZTÁS

Egy adott sorozatban meg kell határozni egy adott tulajdonságú elem **első előfordulását** úgy, hogy tudjuk, hogy **biztosan van ilyen elem** a sorozatban.

Ezekben a feladatokban az a közös, hogy ha az eldöntés tételét alkalmaznánk a feladat megoldása során, akkor mindig igaz választ kapnánk. Itt azonban ez nem elég: meg kell adnunk egy megfelelő elemet, vagy annak sorszámát.

Megjegyzés: Általában érdemesebb a sorszámot meghatározni, mert abból az elem már egyértelműen meghatározható (fordítva nem mindig igaz).

GYAKORLÓ FELADATOK

16. ELSŐ PÁROS

Adjuk meg egy természetes számokat tartalmazó sorozat páros elemét, ha tudjuk, hogy biztosan van ilyen!

A program kimenete:

```
Első páros program

A számokat egy tömbben tároljuk. Az értékek:
-11, -7, 5, -1, 2, 5, 8, 11, 14, 17, -11, -7, 5, -1,

Az első páros szám: 2
```

17. SZÓKÖZ HELYE

Adjuk meg, hogy egy szövegben hányadik karakter a szóköz, ha tudjuk, hogy a szöveg biztosan tartalmaz szóközt!

18. HÓNAP SORSZÁMA

Kérjük be egy hónap nevét. A hónapnevek sorozata alapján mondjuk meg a hónap sorszámát úgy, hogy a program akkor is működjön, ha begépeléskor a kis- és nagybetűket vegyesen használtuk.

LINEÁRIS KERESÉS

Határozzuk meg, hogy **van-e** a sorozatban **adott tulajdonságú elem**, és **ha igen**, akkor **adjuk meg** azt. Vagy magát az elemet adjuk meg, vagy a sorszámát (ez utóbbi javasolt). A „lineáris” kifejezés onnan ered, hogy a sorozatot az első elemétől kezdve vizsgáljuk sorban mindaddig, amíg a kívánt elemet meg nem találjuk (vagy túlmentünk a sorozat hosszán).

Megoldás: Az eldöntés és kiválasztás tétel együttes alkalmazása.

GYAKORLÓ FELADATOK

19. HALLGATÓ KERESZTNEVE

Ismert egy kurzus hallgatóinak névsora (8 elemű vektorban tároljuk). Határozzuk meg az első Péter keresztnevű sorszámát és írassuk ki a teljes nevét (ha van ilyen)! Segítség: annak lekérdezése, hogy a teljesNev változó Péter-re végződik-e: teljesNev.EndsWith(" Peter").

A program kimenete lehet:

```
Hallgató keresztneve program

Van Péter keresztnevű hallgató. Sorszáma: 5. Teljes neve: Kovách Péter
```

20. NEGATÍV ÉRTÉK

Adjuk meg egy egészekből álló 5 elemű sorozatnak egy negatív elemét (ha van ilyen).

21. KÉRDŐJEL

Adjuk meg, hogy egy szövegben hányadik karakter a(z első) kérdőjel (ha van ilyen).

MEGSZÁMLÁLÁS

Meg kell számolnunk, hogy a sorozatban **hány darab adott tulajdonságú elem van**.

Megjegyzés: Ha egyetlen olyan elem sincs a sorozatban, mely adott tulajdonságú, akkor a DB értéke nulla.

GYAKORLÓ FELADATOK

22. FELKIÁLTÓJELEK SZÁMA

Számoljuk meg, hogy hány felkiáltójelet tartalmaz a bekért szöveg.

23. LÉTMINIMUM ALATT

Családok létszáma, illetve jövedelme alapján állapítsuk meg, hogy hány család él a létminimum alatt. A létszám és a jövedelem tárolására 2 különböző vektort tárolunk úgy, hogy az összetartozó adatok ugyanazon indexen találhatók mindkét vektorban.

24. MAGÁNHANGZÓK SZÁMA

Adjuk meg egy szöveg magánhangzóinak számát! A magánhangzókat egy char típusú vektorban tároljuk (14 magánhangzó van). A program ne vegye figyelembe a kis- és nagybetű eltéréseket. A megoldáshoz két egymásba ágyazott ciklust használjunk: a külső (for) ciklus léptet végig a szöveg karakterein, a belső (while) ciklus döntse el, hogy szerepel-e az aktuális karakter a magánhangzók vektorban.

A program kimenete:

```
Magánhangzók száma program

A szöveg = Ez itt a szöveg.
A szövegben 5 magánhangzó van.
```

25. OLIMPIAI KVALIFIKÁCIÓ

Egy futóverseny időeredményei alapján határozzuk meg, hogy a versenyzők hány százaléka teljesítette az olimpiai induláshoz szükséges szintet! Az adatokat egy vektorban tároljuk, melynek méretét és a kvalifikáláshoz szükséges szintet az időeredmények beolvasása után a felhasználótól kérdezzük meg. (Megjegyzés: egy adott futó eredménye annál jobb, minél alacsonyabb az időeredménye.)

A program egy kimenete lehet:

```
Olimpiai kvalifikáció program

Versenyzők száma = 5
Az olimpiai csapatba jutás határa = 2

A versenyzők eredményeinek beolvasása.
eredmenyek[1] = 1
eredmenyek[2] = 2,4
eredmenyek[3] = 3
eredmenyek[4] = 4
eredmenyek[5] = 1,2

Induló versenyzők száma: 5 fő, ebből 3 fő jutott be az olimpiai csapatba,
ami 60%-ot jelent.
```

MAXIMUMKIVÁLASZTÁS (SZÉLSŐÉRTÉK KIVÁLASZTÁSA)

Egy sorozatból **ki kell választani a legnagyobb elemet**, vagy annak sorszámát. Az ilyen típusú feladatoknak csak akkor van értelmük, ha a sorozat hossza legalább 1. Bizonyos feladatokban a legkisebb elemet keressük.

GYAKORLÓ FELADATOK

26. LÁZ

Egy kórházban megmérték minden beteg lázát (10 fő), amit egy vektorban rögzítettek (lazlap), a betegek nevét pedig a betegek nevű vektorban. Adjuk meg, hogy ki a leglázasabb, és hogy mennyi a láza!

27. BEVÉTELEK

Adott egy üzlet napokra lebontott heti bevétele. Adjuk meg a legnagyobb és a legkisebb bevétel napját és értékét!

28. LEGROSSZABB JEGY

Ismerjük egy kurzus hallgatóinak jegyeit! Melyik a legrosszabb jegy? Adjuk meg az értékét és a sorszámát is!

KIEGÉSZÍTÉS:

Határozzuk meg az összes legrosszabb jegy helyét.