



OKTATÁSI HIVATAL

**A 2021/2022. tanévi  
Országos Középiskolai Tanulmányi Verseny  
első forduló**

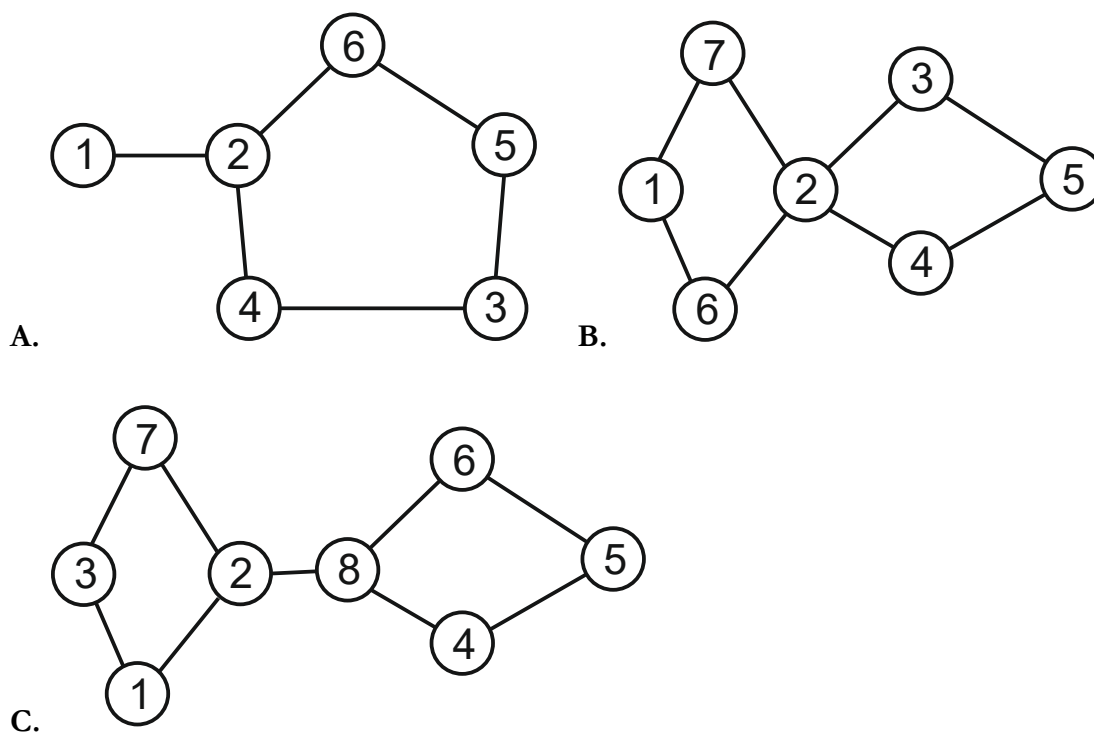
**INFORMATIKA II. (PROGRAMOZÁS) KATEGÓRIA  
FELADATLAP**

**Munkaidő: 180 perc**

**Elérhető pontszám: 400 pont**

**1. feladat: Utak (44 pont)**

Az alábbi ábrákon sorszámozott települések és közöttük vezető utak láthatók. Egyes településekből indulva el lehet jutni az összes településre úgy, hogy közben egy települést sem érintünk kétszer. Add meg, melyek ezek a települések!



Az Országos Középiskolai Tanulmányi versenyek megvalósulását az NTP-TMV-M-21-A0002 projekt támogatja



## 2. feladat: Mit csinál? (58 pont)

Az alábbi algoritmus bementeként kapja a  $K$  ( $K > 1$ ) és  $N$  értékeket, valamint  $N$  darab egész számot az  $X[1] \dots X[N]$  tömbelemekben.

```

D:=0; i:=1; M:=0
Ciklus amíg  $i \leq N$  és  $D < K$ 
    Ha  $X[i]$  páros akkor  $D:=D+1$ 
                                Ha  $D=1$  akkor  $j:=i$ 
    Elágazások vége
    Ha  $D < K$  akkor  $i:=i+1$ 
Ciklus vége
Ha  $D=K$  akkor
     $M:=i-j+1$ 
     $A:=j$ 
     $B:=i$ 
    Ciklus amíg  $i \leq N$ 
         $i:=i+1$ 
        Ciklus amíg  $i \leq N$  és  $X[i]$  nem páros
             $i:=i+1$ 
        Ciklus vége
         $j:=j+1$ 
        Ciklus amíg  $X[j]$  nem páros
             $j:=j+1$ 
        Ciklus vége
        Ha  $i-j+1 > M$  és  $i \leq N$  akkor  $M:=i-j+1$ ;  $A:=j$ ;  $B:=i$ 
    Ciklus vége
Elágazás vége
    
```

- A. Mi kerül az  $M$ ,  $A$ ,  $B$  változóba, ha  $K=2$ ,  $N=8$ ,  $X=[3, 2, 5, 4, 3, 3, 2, 1]$ ?
- B. Mi kerül az  $M$ ,  $A$ ,  $B$  változóba, ha  $K=3$ ,  $N=8$ ,  $X=[3, 2, 5, 4, 3, 3, 2, 2]$ ?
- C. Mi a feltétele annak, hogy a végrehajtás után  $M$  értéke 0 maradjon?
- D. Fogalmazd meg általánosan, hogyan függ  $M$ ,  $A$ ,  $B$  értéke a bemenettől?
- E. Az első ciklus után az  $i$  és a  $j$  változó értéke hogyan függ a bemenettől?
- F. Mi a szerepe az algoritmusban az  $i$  és a  $j$  változónak?

## 3. feladat: Törpék (70 pont)

A hét törpe elhatározta, hogy számítógépet fognak játszani, de nem ért minden törpe mindenhez. Morgó és Hapci fájlból tud olvasni (a fájlok neve egy  $i$  k, illetve más  $i$  k), Tudor hasonlítani tud, Szundi és Kuka pedig csak fájlba tud írni. Hapci, Morgó és Tudor folyamatosan figyelnek valamilyen jelzőberendezést (kezdetben mindegyik tilosra van állítva), és ha kell, dolgoznak, ezzel szemben Szundi és Kuka csak akkor dolgozik, ha felszólítást kap (minden felszólításra elindul a programjuk), és a felszólító megvárja, amíg végeznek. Ha már nincs mit olvasniuk, azt is jelzik egy közös logikai változóban. Használnak a jelzőkön kívül két közös adatot ( $A$  és  $B$ ). A két fájlban az adatok növekvő sorrendben vannak, és állományonként egy szám legfeljebb egyszer szerepelhet.

A programjuk így néz ki:

Hapci:

```
Ciklus amíg nem kész1
  Várj amíg jelző1 szabad
  Ha van adat(egyik) akkor Olvas(egyik,A)
                                jelző1 legyen szabad
  különben kész1 legyen igaz
Ciklus vége
```

Eljárás vége.

Morgó:

```
Ciklus amíg nem kész2
  Várj amíg jelző2 szabad
  Ha van adat(másik) akkor Olvas(másik,B)
                                jelző2 legyen szabad
  különben kész2 legyen igaz
Ciklus vége
```

Eljárás vége.

Tudor:

```
Várj amíg jelző1 tilos vagy jelző2 tilos
Ciklus amíg nem kész1 és nem kész2
  Ha A<B akkor jelző1 legyen tilos
  különben ha A>B akkor jelző2 legyen tilos
  különben Hívd Kukát dolgozni
  Várj amíg jelző1 tilos és nem kész1 vagy
                                jelző2 tilos és nem kész2
Ciklus vége
```

Eljárás vége.

Kuka:

```
Ír(harmadik,A); jelző1 legyen tilos; jelző2 legyen tilos
Eljárás vége.
```

**A.** Mi lesz a harmadik-ban, ha egyik=(3,5,7,11,13) és másik=(1,5,8,11,13)?

**B.** Mi lesz a harmadik-ban, ha egyik=(2,4,6,8,10) és másik=(2,4,8,16)?

**C.** Fogalmazd meg általánosan, hogy a két fájl tartalmától függően mi kerül a harmadikba!

Átírjuk Tudor és Kuka programját, és kap munkát Vidor is:

Tudor:

```
Várj amíg jelző1 tilos vagy jelző2 tilos
Ciklus amíg nem kész1 és nem kész2
  Ha A<B akkor Hívd Kukát dolgozni
  különben ha A>B akkor jelző2 legyen tilos
  különben jelző1 legyen tilos; jelző2 legyen tilos
  Várj amíg jelző1 tilos és nem kész1 vagy
                                jelző2 tilos és nem kész2
```

Ciklus vége

Ha kész2 akkor Hívd Vidort dolgozni

Eljárás vége.

Kuka:

```
Ír(harmadik,A); jelző1 legyen tilos
Eljárás vége.
```

Vidor:

```
Ciklus amíg nem kész1
  Várj amíg jelző1 tilos
  Hívd Kukát dolgozni
Ciklus vége
```

Eljárás vége

**D.** Mi lesz a harmadik-ban, ha egyik=(3, 5, 7, 11, 13) és másik=(1, 5, 8, 11, 13)?

**E.** Mi lesz a harmadik-ban, ha egyik=(2, 4, 6, 8, 10) és másik=(4, 8, 16)?

**F.** Fogalmazd meg általánosan, hogy a két fájl tartalmától függően mi kerül a harmadikba! Mi lenne, ha Vidor nem dolgozna?

Tudor programját módosítjuk és bevetjük Szendét és Szundit is:

Tudor:

```
Várj amíg jelző1 tilos vagy jelző2 tilos
Ciklus amíg nem kész1 és nem kész2
  Ha A<B akkor Hívd Kukát dolgozni
  különben ha A>B Hívd Szundit dolgozni
  különben jelző1 legyen tilos; jelző2 legyen tilos
  Várj amíg jelző1 tilos és nem kész1 vagy
  jelző2 tilos és nem kész2
```

Ciklus vége

Ha kész1 akkor Hívd Szendét dolgozni

Ha kész2 akkor Hívd Vidort dolgozni

Eljárás vége.

Szundi:

```
Ír(harmadik,B); jelző2 legyen tilos
```

Eljárás vége.

Szende:

```
Ciklus amíg nem kész2
  Várj amíg jelző2 tilos
  Hívd Szundit dolgozni
Ciklus vége
```

Eljárás vége

**G.** Mi lesz a harmadik-ban, ha egyik=(3, 5, 7, 11, 13) és másik=(1, 5, 8, 11, 13)?

**H.** Mi lesz a harmadik-ban, ha egyik=(2, 4, 6, 8, 10) és másik=(4, 8, 16)?

**I.** Fogalmazd meg általánosan, hogy a két fájl tartalmától függően mi kerül a harmadikba! Mi lenne, ha Szende nem dolgozna?

#### 4. feladat: Kannák (67 pont)

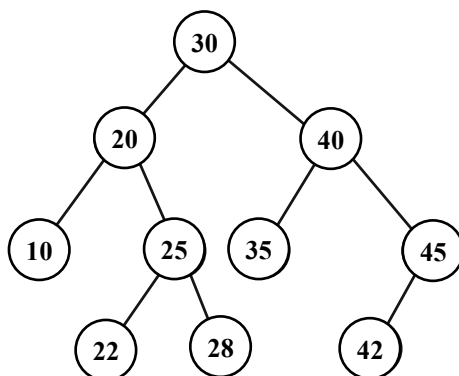
Egy gazdának három kannája van, az egyik **A** literes, a másik **B**, a harmadik pedig **C** literes. Kezdetben az első kanna tele van, a másik kettő pedig üres. Szeretne kimérni pontosan **L** liter vizet. Az alábbi műveleteket lehet végezni a kimérés során:

- Áttöltés az A-litereseből a B-literesbe (amíg az tele nem lesz, ill. van A-ban)
- Áttöltés az A-litereseből a C-literesbe (amíg az tele nem lesz, ill. van A-ban)
- Áttöltés a B-litereseből az A-literesbe (amíg az tele nem lesz, ill. van B-ben)
- Áttöltés a B-litereseből a C-literesbe (amíg az tele nem lesz, ill. van B-ben)
- Áttöltés a C-litereseből az A-literesbe (amíg az tele nem lesz, ill. van C-ben)
- Áttöltés a C-litereseből a B-literesbe (amíg az tele nem lesz, ill. van C-ben)

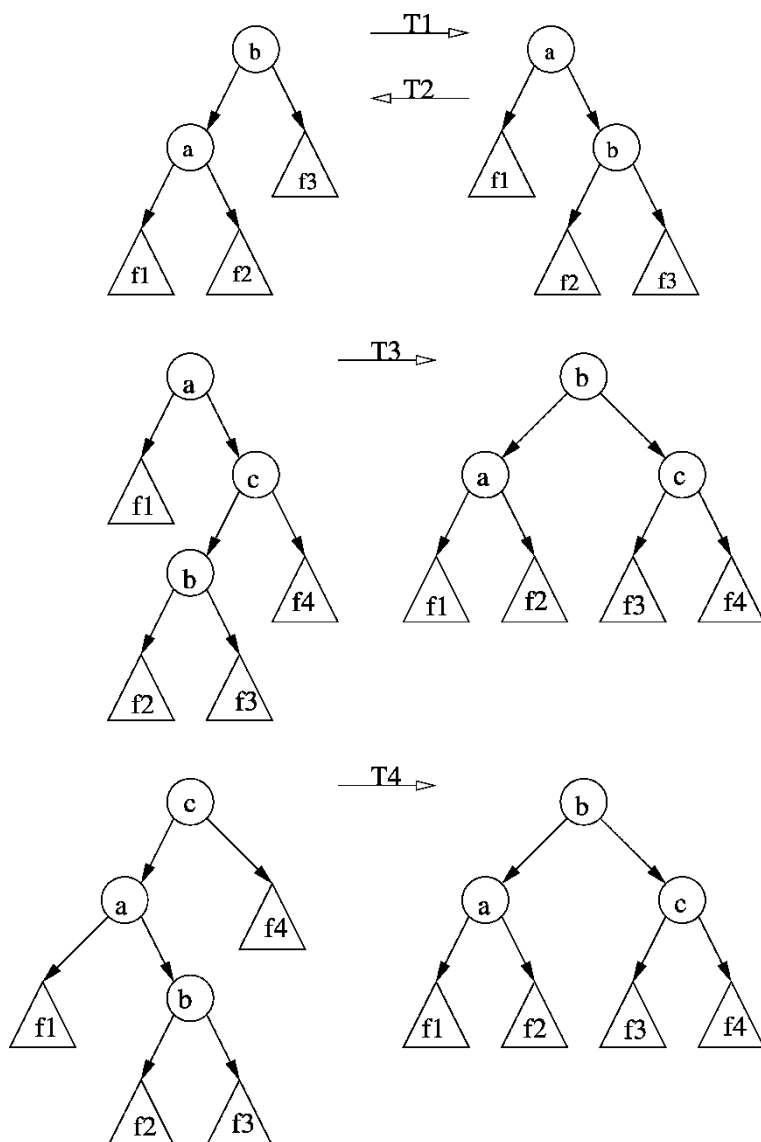
- A.** Minimum hány öntéssel tud kimérni 6 liter vizet, ha a három kanna 8, 5 és 3 literes? Adj is meg egy lehetséges lépéssort!
- B.** Minimum hány öntéssel tud kimérni 4 liter vizet, ha a három kanna 8, 5 és 3 literes? Adj is meg egy lehetséges lépéssort!
- C.** Minimum hány öntéssel tud kimérni 1 liter vizet, ha a három kanna 8, 5 és 3 literes? Adj is meg egy lehetséges lépéssort!
- D.** Minimum hány öntéssel tud kimérni 7 liter vizet, ha a három kanna 10, 8 és 5 literes? Adj is meg egy lehetséges lépéssort!
- E.** Minimum hány öntéssel tud kimérni 7 liter vizet, ha a három kanna 14, 8 és 5 literes? Adj is meg egy lehetséges lépéssort!
- F.** Minimum hány öntéssel tud kimérni 9 liter vizet, ha a három kanna 11, 7 és 5 literes? Adj is meg egy lehetséges lépéssort!
- G.** Minimum hány öntéssel tud kimérni 3 liter vizet, ha a három kanna 11, 7 és 5 literes? Adj is meg egy lehetséges lépéssort!
- H.** Minimum hány öntéssel tud kimérni 8 liter vizet, ha a három kanna 11, 7 és 5 literes? Adj is meg egy lehetséges lépéssort!

### 5. feladat: Kiegyensúlyozás (56 pont)

Egy keresőfa minden csomópontjára igaz, hogy tőle balra csak nála kisebb értékű elemek vannak, jobbra pedig nagyobbak, ahogy az ábrán látható.

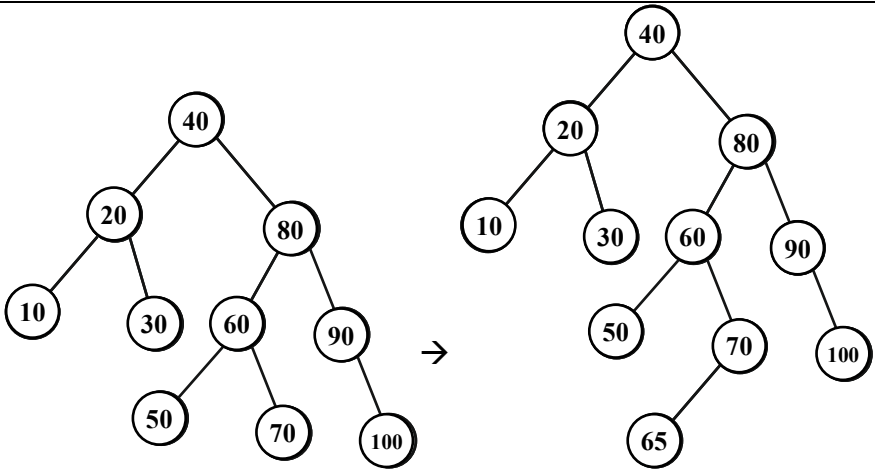
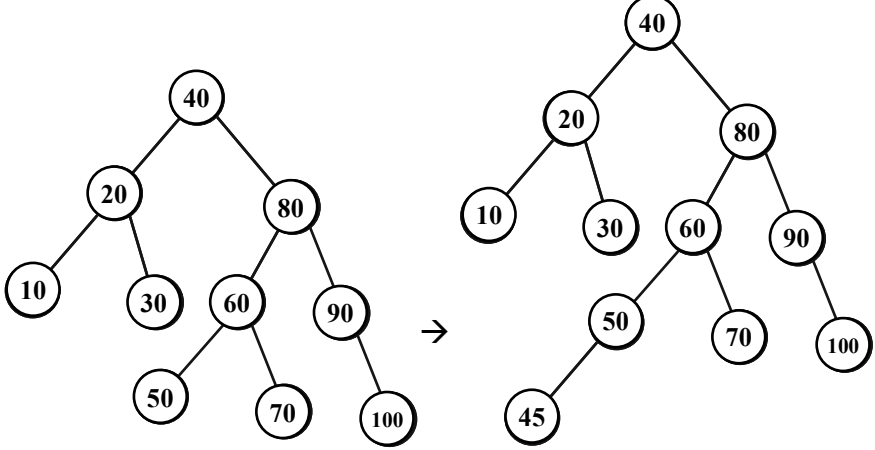


Kiegyensúlyozott az a bináris fa, amelynek tetszőleges pontjában „gyökerező” részfáinak magassága legfeljebb eggyel tér el egymástól (AVL-fa). Ha egy fa kiegyensúlyozottsága elromlik, akkor azt helyre lehet állítani a keresőfa tulajdonság megtartásával, a fa ügyes transzformálásával. Ha a fa egy része válik kiegyensúlyozatlanná, akkor a transzformációt a legelső kiegyensúlyozatlan részre kell elvégezni! Ezek a szabályos transzformációk:



<p><b>A.</b> Ha ebbe a keresőfába a 42-t szúrjuk be, akkor így romlana el a kiegyensúlyozottsága. Rajzold le a kiegyensúlyozott fát egy szabályos transzformáció után!</p>	
<p><b>B.</b> Ha ebbe a keresőfába a 10-et szúrjuk be, akkor így romlana el a kiegyensúlyozottsága. Rajzold le a kiegyensúlyozott fát egy szabályos transzformáció után!</p>	

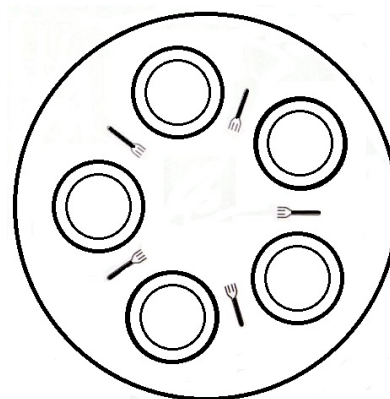
<p><b>C.</b> Ha ebbe a keresőfába az 5-öt szűrjük be, akkor így romlana el a kiegyensúlyozottsága. Rajzold le a kiegyensúlyozott fát egy szabályos transzformáció után!</p>	
<p><b>D.</b> Ha ebbe a keresőfába a 45-öt szűrjük be, akkor így romlana el a kiegyensúlyozottsága. Rajzold le a kiegyensúlyozott fát egy szabályos transzformáció után!</p>	
<p><b>E.</b> Ha ebbe a keresőfába a 65-öt szűrjük be, akkor így romlana el a kiegyensúlyozottsága. Rajzold le a kiegyensúlyozott fát egy szabályos transzformáció után!</p>	

<p><b>F.</b> Ha ebbe a keresőfába a 65-öt szűrjük be, akkor így romlana el a kiegyensúlyozottsága. Rajzold le a kiegyensúlyozott fát egy szabályos transzformáció után!</p>	
<p><b>G.</b> Ha ebbe a keresőfába a 45-öt szűrjük be, akkor így romlana el a kiegyensúlyozottsága. Rajzold le a kiegyensúlyozott fát egy szabályos transzformáció után!</p>	

## 6. feladat: Étkező filozófusok (45 pont) <sup>1</sup>

Adott öt filozófus (0-tól 4-ig sorszámozva), akik egy asztal körül ülnek és beszélgetnek, filozofálnak egymással. Mindegyik filozófus előtt van egy tál spagetti, hogyha megéhezne, tudjon enni. Mindegyik tányér mellett van egy villa. Egy apró probléma van: túlságosan csúszosra sikerült a spagetti, így két villára van szükség az evéshez.

Mindegyik filozófus gondolkodik, majd ha megéhezik, próbálja megszerezni a tányérja melletti villákat, hogy egyen. Ha evett, visszateszi a villákat a helyükre, és folytatja a gondolkodást. A feladat az, hogy készítsünk olyan programot, ami szimulálja ezt a folyamatot!



Egy kézenfekvő megoldás lehet az alábbi, ahol a `kell_villa` művelet addig várakozik, amíg valaki kezében van a villa, a villát pedig a `nemkell_villa` művelettel lehet letenni:

<sup>1</sup> [https://regi.tankonyvtar.hu/hu/tartalom/tamop412A/2010-0011\\_szamalap2/lecke6\\_lap4.html](https://regi.tankonyvtar.hu/hu/tartalom/tamop412A/2010-0011_szamalap2/lecke6_lap4.html) tankönyv alapján.



```

filozófus(i):
    Ciklus
        gondolkodom()
        kell_villa(i) // A bal oldali villát kell megszerezni.
        kell_villa((i+1) mod 5) // A jobb oldali villa is kell.
        eszem()
        nemkell_villa(i) // A bal oldali villát visszarakom.
        nemkell_villa((i+1) mod 5) // A jobb oldali villát vissza.
    Ciklus vége
Eljárás vége.

```

Sajnos kialakulhat éhezési helyzet, azaz szakszóval holtpont, amikor senki nem tud eljutni az evéshez.

**A.** Fogalmazd meg, milyen esetben lehetséges ez a holtpont?

**B.** Maximum hány filozófus tudna egyszerre enni?

**C.** Maximum hány filozófus foghat meg legalább 1 villát egyszerre, hogy ne alakuljon ki holtpont? Magyarázd meg, hogy miért! Mennyi időegység alatt végeznek az evéssel ebben az esetben, ha mindenki egyszer eszik és egy evés 1 időegységig tart? Magyarázd meg, hogy miért!

**D.** Legjobb esetben mennyi időegység alatt végeznek az evéssel ebben az esetben, ha mindenki egyszer eszik és egy evés 1 időegységig tart? Fogalmazd meg, hogy hogyan!

## 7. feladat: Sorozat (60 pont)

Az alábbi algoritmus egy sorozat értékeit számolja ki:

```

a[1]:=1
Ciklus i=1-től n-1-ig
    Ha valami(i,i) akkor a[i+1]:=a[i]+2 különben a[i+1]:=a[i]+1
Ciklus vége

```

A számítás használja a valami, logikai értékű függvényt:

```

valami(m,n):
    Ha m<1 akkor valami:=hamis
    különben ha a[m]=n akkor valami:=igaz
    különben valami:=valami(m-1,n)
Függvény vége.

```

**A.** Mi kerül az a vektorba, ha n=10?

**B.** Mi a feladata a valami(m,n) függvénynek?

**C.** A valami függvény kiszámítása hatékonyabbá tehető egyetlen elágazás feltétele megváltoztatásával. Melyik feltétel és mire cserélendő? Miért?

**D.** A valami függvény rekurzió helyett ciklussal is megoldható az alábbi struktúrában. Egészítsd ki, hogy hatása azonos legyen a rekurzív megvalósítással!

```

valami(m,n):
    Ciklus amíg [ ] { * }
        [ ] { ** }
    Ciklus vége
    Ha [ ] akkor valami:=hamis { *** }
        különben valami:=igaz
Függvény vége.

```

**E.** Egy újabb tömb bevezetésével, egy elemének vizsgálatával a valami függvény hívása megszüntethető. Az új tömb elemei egyszerű értékekkel számíthatók. Írd le, hogy hogyan?