Oktatási Hivatal

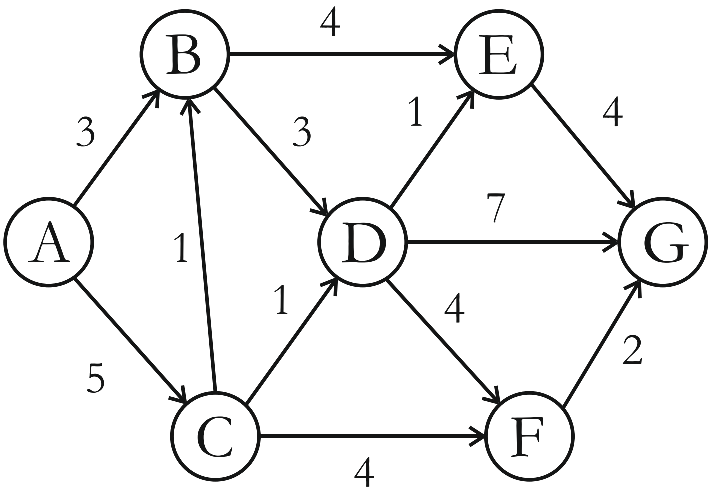
A 2019/2020 tanévi Országos Középiskolai Tanulmányi Verseny
első forduló javítási-értékelési útmutató

INFORMATIKA II. (programozás) kategória

Kérjük a tisztelt tanár kollégákat, hogy a dolgozatokat az egységes értékelés érdekében szigorúan az alábbi útmutató szerint pontozzák, a megadott részpontszámokat ne bontsák tovább! Vagyis ha egy részmegoldásra pl. 3 pontot javaslunk, akkor arra vagy 0, vagy 3 pont adható. (Az útmutatótól eltérő megoldások is lehetnek jók.)

1. feladat: Mikorra érhet oda (60 pont)

Az alábbi ábrán egy sétapálya látható, amelyről tudjuk, hogy egy ember melyik szakaszt mennyi idő alatt tudja bejárni. A pályán a nyilaknak megfelelő irányba lehet haladni.



A. Add meg, hogy merre kell menni A-ból G-be, hogy G-be a lehető leghamarabb érjünk! Több megoldás esetén az összes útvonalat add meg!

B..G. Add meg, hogy az egyes pontokra (B..G) leghamarabb mennyi idő alatt lehet odaérni az A pontból indulva, illetve, hogy ezt hányféleképpen lehet megtenni!

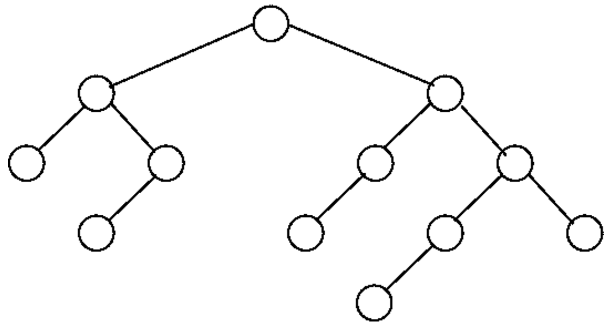
Értékelés:

A. A – B – E – G; A – B – D – E – G; A – C – D – E – G; A – C – F – G	5+5+5+5 pont
B. 3, 1	4+1 pont
C. 5, 1	4+1 pont
D. 6, 2	4+3 pont
E. 7, 3	4+3 pont
F. 9, 1	5+3 pont
G. 11, 4	5+3 pont

2. feladat: Kiegyensúlyozott bináris fa (60 pont)

Bináris fát magasság-kiegyensúlyozottnak nevezünk, ha minden pontjára teljesül, hogy a bal- és jobb-részfájának magassága legfeljebb eggyel tér el. Az üres fa magassága 0, az egy pontú fa magassága 1, egyébként 1+ a két részfa magasságának a maximuma.

Az ábrán látható bináris fa magasság-kiegyensúlyozott, magassága 5, és 12 pontja van.



Legfeljebb mekkora a magassága az olyan magasság-kiegyensúlyozott bináris fának, amelynek

A: 4 pontja van,
B: 5 pontja van,
C: 6 pontja van
D: 7 pontja van,
E: 15 pontja van,
F: 19 pontja van,
G: 33 pontja van,
H: 42 pontja van;
I. 100 pontja van;
J. 1000 pontja van;
K. 2019 pontja van;
L. 3333 pontja van;
M. 5000 pontja van;
N. 9999 pontja van;
O. 25 252 pontja van?

Értékelés:

Minden helyes válasz 4 pontot ér.

A: 3, B: 3, C: 3, D: 4, E: 5, F: 5, G: 7, H: 7, I: 9, J: 14, K: 15, L: 16, M: 17, N: 18, O: 20.

Magyarázat: a H magasságú fa minimális elemszáma T(H)=T(H-1)+T(H-2)+1 – ha kiegyensúlyozott, akkor az egyik oldal eggyel lehet magasabb, mint a másik.

Adott N-re a megoldás az a H, amire T(H)≤N<T(H+1).

A T vektor első 21 értéke: 1, 2, 4, 7, 12, 20, 33, 54, 88, 143, 232, 376, 609, 986, 1596, 2533, 4130, 6664, 10795, 17460, 28256.

3. feladat: Mit csinál (70 pont)

Az alábbi algoritmus egy N elemű, 1-től N-ig indexelt x sorozatot dolgoz fel, amely 1 és M közötti egész értékeket tartalmaz. Eredménye az A, B, C és V változókba kerül. Az u vektort 1-től M-ig indexeljük.

```
Valami(V,A,B,C):
  m:=N; u:=(-N,…,-N)
  Ciklus i=1-től N-ig
    Ha i-u[x[i]]<m akkor m:=i-u[x[i]]; B:=u[x[i]]; C:=i
    u[x[i]]:=i
  Ciklus vége
  V:=(m<N)
  Ha V akkor A:=x[B]
Eljárás vége.
```

A. Mi lesz A, B, C és V értéke, ha `N=10`, `x=(3,5,3,2,2,4,2,8,9,1)`?

B. Mi lesz A, B, C és V értéke, ha `N=10`, `x=(3,5,6,7,8,2,5,6,7,8)`?

C. Mi lesz `u[1]..u[8]`-ban, ha `N=10`, `x=(3,5,6,7,8,2,5,6,7,8)`?

D. Milyen `x` vektor esetén kap `V` hamis értéket?

E. A ciklus i. lépése után mit tartalmaz az `u` vektor?

F. Mi lesz A, B és C értéke, ha V igaz értéket kap az eljárásban?

Értékelés:

A. `A=2`, `B=4`, `C=5`, `V=igaz` 2+4+4+2 pont

B. `A=5`, `B=2`, `C=7`, `V=igaz` 2+4+4+2 pont

C. `u=(-10,6,1,-10,7,8,9,10)` 8*1 pont

D. V hamis lesz, ha x-ben nincs két egyforma érték 7 pont

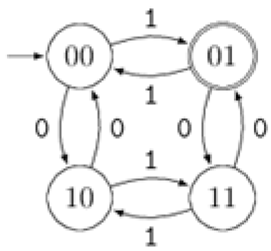
E. `u[k]`=az i. elemig a `k` hol fordult elő utoljára az `x` vektorban; `-N`, ha a `k` addig nem fordult elő 8+5 pont

F. `A` az az érték, amelynek két előfordulása a lehető legközelebb van egymáshoz; `B` ennek a legelső előfordulása; `C` pedig a következő 8+5+5 pont

4. feladat: Automata (65 pont)[1]

Egy automata kezdetben `00` állapotban van, jeleket olvas és a jelek hatására az állapota megváltozhat. Ha `00` állapotban a bemenetére 1-es jel érkezik, akkor `01` állapotba kerül, ha 0 érkezik, akkor `10` állapotba. Ha `01` állapotban a bemenetére 0 érkezik, akkor `11`-be kerül, ha pedig 1, akkor `00`-ba. Ha `10` állapotban a bemenetére 1 érkezik, akkor `11`-be kerül, ha pedig 0, akkor `00`-ba. Ha `11` állapotban a bemenetére 0 érkezik, akkor `01`-be kerül, ha pedig 1, akkor `10`-ba.

Az automata az alábbi rajzzal ábrázolható:



A. Milyen állapotban lesz az automata az 11010 jelsorozat hatására? Add meg az egyes jelek utáni állapotot is!

B. Milyen állapotban lesz az automata a 0101010011 jelsorozat hatására? Add meg az egyes jelek utáni állapotot is!

C. Milyen jelsorozatokra lesz az automata a legvégén 00, 01, 10, illetve 11 állapotban?

D. Add meg ábrával azt a minimális állapotszámú automatát, amely képes felismerni az olyan 0 és 1 jelekből álló sorozatokat, amelyekben az 1-esek száma hárommal osztható! Vagyis legyen egy kitüntetett állapota, amelyben pontosan ezen jelsorozatok végén lesz. Dupla bekarikázással jelöld ezt az állapotot!

[1] A feladat Friedl Katalin, Csima Judit: Nyelvek és automaták című könyvéből származik: https://www.tankonyvtar.hu/hu/tartalom/tamop412A/2011-0064_58_nyelvek_es_automatak/ar01s02.html

Értékelés:

A1. A végső állapot: 01 — 4 pont

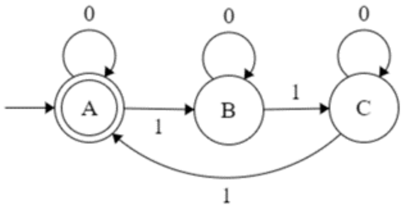
A2. Az egyes jelek utáni állapotok: 01,00,10,11,01 — 5*1 pont

B1. A végső állapot: 11 — 4 pont

B2. Az egyes jelek utáni állapotok: 10,11,01,00,10,11,01,11,10,11 — 10*1 pont

C. Az automata akkor kerül a 00 állapotba, ha páros sok 0 és páros sok 1 karaktert olvasott; a 01 állapot felel meg a páros sok 0 és páratlan 1 karakternek; 10 a páratlan sok 0 és páros sok 1-nek; 11 pedig a páratlan sok 0 és 1 karakternek — 5+5+5+5 pont

D. Három állapottal megoldható, a megoldás egyértelmű:



Pontozás: Három állapot van és az 1-es nyilak jók; a 0-ás (hurok) nyilak jók; duplán be van karikázva az elfogadó (A) állapot; a kezdőállapotba (A) mutat nyíl — 8+6+6+2 pont

<u>5. feladat</u>: Gépközeli programozás (40 pont)

Az alábbiakban látható egy gépközeli nyelven megírt programrészlet, amely a C és B címen levő 32 bites pozitív egész számokból számol ki egy 32 bites pozitív egész számot, amit az A címre tesz. A segédszámításokhoz a processzor három, 32-bites regiszterét (a processzor tárolóegységét) használja (`eax`, `ebx`, `ecx`).

Egy kettes számrendszerbeli szám balra léptetése azt jelenti, hogy a legmagasabb helyiértékű bitjét törli és a végére egy 0-s bitet tesz. A jobbra léptetés a legalacsonyabb helyiértékű bitet törli és az elejére tesz egy 0-s bitet.

```
      mov ecx,C ; az ecx regiszterbe tölt egy értéket a C címről
      mov ebx,B ; az ebx regiszterbe tölt egy értéket a B címről
      mov eax,1 ; az eax regiszterbe 1-et tölt
xxx:  cmp ebx,0 ; összehasonlítja ebx tartalmát a nullával
      je yyy    ; az yyy címen folytatja, ha egyenlők voltak
      bt ebx,0  ; megnézi ebx legalacsonyabb helyiértékű bitjét
      jc zzz    ; a zzz címen folytatja, ha 1-es értékű volt
      sq ecx    ; az ecx regiszter tartalmát négyzetre emeli
      shr ebx   ; az ebx regisztert 1 bittel jobbra lépteti
      jmp xxx   ; az xxx címen folytatja
zzz:  mul ecx   ; megszorozza eax-et ecx-szel
      dec ebx   ; eggyel csökkenti ebx-et
      jmp xxx   ; az xxx címen folytatja
yyy:  mov A,eax ; az A címre teszi az eax regiszter tartalmát
```

A. Mi lesz az A címen, ha C=42, B=1?

B. Mi lesz az A címen, ha C=3, B=4?

C. Mi lesz az A címen, ha C=2, B=10?

D. Hogyan függ a kiszámított A értéke B-től és C-től?

E. Milyen esetben lesz az eredmény értelmezhetetlen (hibás)?

F. Hányszor lép a program a `zzz` címre és ez mitől függ?

G. Hányszor lép a program az `xxx` címre és ez mitől függ?

Értékelés:

A. A=42	5 pont
B. A=81	5 pont
C. A=1024	5 pont
D. A=C^B	6 pont
E. Ha C^B túl nagy lenne (nem fér el 32 bites egész számban)	5 pont
F. Annyiszor, ahány 1-es bit van B kettes számrendszerbeli felírásában	7 pont
G. A B kettes számrendszerbeli felírásának legmagasabb helyiértékű, 1-es értékű bitjéig levő számjegyek száma+ az 1-es értékű bitjei száma+1 (ha a +1 lemarad, akkor csak 6 pont)	7 pont

<u>6. feladat</u>: Hasító táblák (50 pont)

A hasító tábla egy olyan adatszerkezet, amely segítségével hatékonyan kereshetünk egy kulcs alapján egy adathalmazban. A kereséshez használt kulcsot előbb mindig egy számmá képezzük. A tárolásra egy egyszerű, `0`-tól `K-1`-ig indexelt tömböt használunk. Beszúráskor egy `N` kulcsú adatot a tömb `(N mod K)`. pozíciójába szeretnénk tárolni, azonban, ha az már foglalt, akkor sorra a következő pozíciókkal próbálkozunk, ha a tömb végére érünk, akkor az elején folytatjuk a szabad hely keresését.

Például egy 7 elemű hasító táblába szeretnénk a 44-es kulcsú adatot beszúrni. Ez a `44 mod 7=2`. pozícióra kerül.

		44				

Ezután egy 130-as kulcsú adatot szúrunk be, ami a `130 mod 7=4`. pozícióra kerül.

		44		130		

Ezután a 2-es beszúrásával folytatjuk, ami a `2 mod 7=2`. pozícióra kerülne, de ez foglalt, ezért próbálkozunk a következő pozícióval, ami szabad.

		44	2	130		

A keresés a beszúráshoz hasonlóan működik. A kulcs alapján számított helyen keressük az adatot. Például, ha a 2-es kulcshoz tartozó adatra vagyunk kíváncsiak, akkor a `2 mod 7=2`. pozícióban keressük először, ott másik elem van, ezért a következő pozíción folytatjuk, ahol meg is találjuk a keresett értéket, ehhez 2 tömbelem olvasására volt szükség. Ha például a 18-as kulcsot keressük, azt a 4-es pozícióban kell keresnünk, ahol nem ez az elem szerepel, ezért a következő helyen próbálkozunk, az viszont üres, így 2 tömbelem olvasásával megtudtuk, hogy a keresett kulcs nem szerepel az adatok között.

Lehetőség van törlésre is. A keresés segítségével megkeressük az adott kulcsú adatot, majd bejelöljük, hogy az adott elem törölve lett. Azért van szükség a törölt mezők megkülönböztetésére, mert a keresést ezeknél még nem hagyhatjuk abba. Beszúráskor viszont tehetjük az elemet törölt pozícióra. Például a 44 törlése után így néz ki a tömb:

		törölt	2	130		

Egymás után végezd el a következő műveleteket egy kezdetben üres, 13 elemű tömböt használó hasítótáblán! A beszúrásoknál add meg, hogy melyik pozícióba kerül az adott elem, a kereséseknél pedig, hogy hány tömbelemet kellett olvasni, ahhoz hogy megtaláljuk azt, vagy megbizonyosodjunk a keresett elem hiányáról!

A. `Beszúr: 134`

B. `Beszúr: 39`

C. `Beszúr: 6`

D. `Beszúr: 265`

E. `Beszúr: 12`

F. `Beszúr: 388`

G. `Beszúr: 5`

H. `Keres: 265`

I. `Keres: 22`

J. `Keres: 5`

K. `Keres: 135`

`Töröl: 265`

L. `Keres: 11`

M. `Keres: 5`

`Töröl: 12`

N. `Keres: 12`

O. `Beszúr: 25`

P. Rajzold le a tömb végső állapotát!

Értékelés:

A. A 134 a **4.** pozícióra kerül. 3 pont

B. A 39 a **0.** pozícióra kerül. 3 pont

C. A 6 a **6.** pozícióra kerül. 3 pont

D. A 265 az **5.** pozícióra kerül. 3 pont

E. A 12 a **12.** pozícióba kerül. 3 pont

F. A 388 a **11.** pozícióba kerül. 3 pont

G. Az 5 a **7.** pozícióba kerül. 3 pont

H. 1 olvasás 3 pont

I. 1 olvasás 3 pont

J. 3 olvasás 3 pont

K. 4 olvasás 3 pont

L. 4 olvasás 3 pont

M. 3 olvasás 3 pont

N. 3 olvasás 3 pont

O. A 25 a **12.** pozícióba kerül. 3 pont

P. 5 pont

39				134	törölt	6	5				388	25

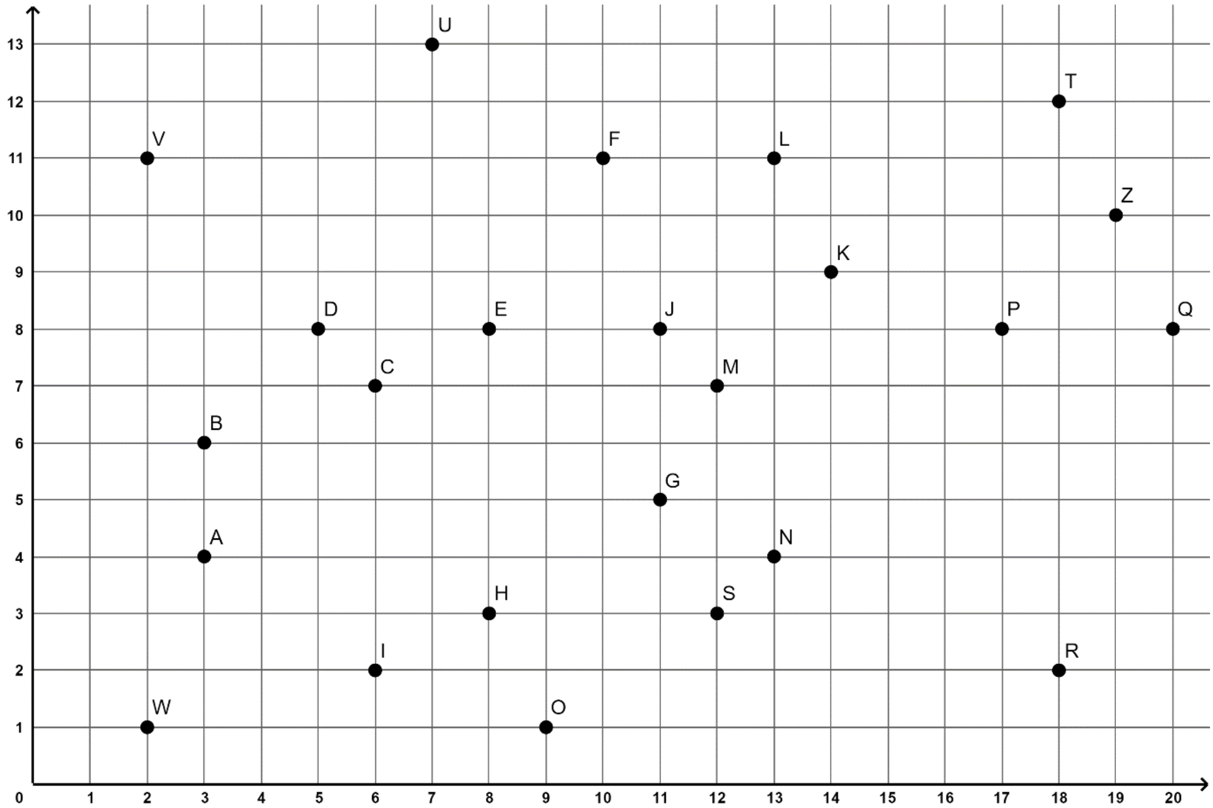
7. feladat: Szenzorhálózat (55 pont)

Repülőgépről ledobálva telepítünk szenzorokat egy területen. A szenzoroknak szükségük van egymás közötti kommunikációra, azonban a jeladók teljesítménye miatt két szenzor csak akkor képes egymással kommunikálni, ha a légvonalbeli távolságuk legfeljebb 5 egység. Távolabbi szenzorpárok ezért csak úgy tudnak egymásnak üzenni, ha más szenzorokon keresztül teszik ezt meg.

Kétféle mohó stratégiával választhatják a szenzorok a csomagok továbbításakor a következő szenzort:

- Távolság: Az elérhető szenzorok közül – akár önmagát választva - annak továbbítjuk a csomagot, amely a célponthoz legközelebb van.
- Irány: Az elérhető szenzorok közül - önmagát kivéve - annak továbbítjuk a csomagot, amely leginkább a cél irányában van, azaz amelynek az aktuális pontból nézve a célhoz képest vett szögelfordulása minimális.

A képen látható pozíciókba sikerült a repülőről telepítenünk a szenzorokat.



A. Add meg a csomag által bejárt útvonalat a K szenzorból az I szenzorba küldéskor **távolság** alapú útválasztás esetén!

B. Add meg a csomag által bejárt útvonalat az U szenzorból a W szenzorba küldéskor **távolság** alapú útválasztás esetén!

C. Add meg a csomag által bejárt útvonalat a T szenzorból a V szenzorba küldéskor **távolság** alapú útválasztás esetén! Mi történik?

D. Add meg a csomag által bejárt útvonalat a W szenzorból az M szenzorba küldéskor **irány** alapú útválasztás esetén!

E. Add meg a csomag által bejárt útvonalat a P szenzorból a D szenzorba küldéskor **irány** alapú útválasztás esetén!

F. Add meg a csomag által bejárt útvonalat a V szenzorból az R szenzorba küldéskor **irány** alapú útválasztás esetén! Mi történik?

Értékelés:

A. (K →) G → H (→ I) (ameddig helyes, max. 2) * 2

B. (U →) F → E → H → I (→ W) (ameddig helyes, max. 4) * 2

C. (T →) K → F → U (ameddig helyes, max. 3) * 2

Nem lehet tovább menni / U-ban marad / elakad (előzőtől függetlenül) 5

D. (W →) I → H → G (→ M) (ameddig helyes, max. 3) * 3

E. (P →) K → J → E (→ D) (ameddig helyes, max. 3) * 3

F. (V →) D → C → H → (ameddig helyes, max. 3) * 3

S ↔ N / S-N oda-vissza / S-N kör (előzőtől függetlenül) 5

Összpontszám: 400 pont **Beküldési határ: 150 pont**