OKTATÁSI HIVATAL

A 2024/2025. tanévi
Országos Középiskolai Tanulmányi Verseny
első forduló

DIGITÁLIS KULTÚRA II. (PROGRAMOZÁS) KATEGÓRIA
FELADATLAP

Munkaidő: 180 perc
Elérhető pontszám: 400 pont

1. feladat (60 pont)

A lenti függvény bemeneti paraméterei az A és B pozitív egészek, valamint az S pozitív egészeket tartalmazó, N elemű tömb. Feltétel továbbá, hogy az A+B összeg nem nagyobb S elemeinek összegénél. A tömböket 1-től indexeljük.

```
Előállít(A,B,S):
  i:=1; j:=1;
  Ciklus k=1-től N-ig
    Ha A≥S[k] akkor
      SA[i]:=S[k]; i:=i+1;
      A:=A-S[k];
    különben
      Ha B≥S[k] akkor
        SB[j]:=S[k]; j:=j+1;
        B:=B-S[k];
  Ciklus vége
  Ha A>0 vagy B>0 akkor
    Ki: „Hiba!”
  különben
    Ki: SA,SB
Eljárás vége
```

Válaszolj az alábbi kérdésekre!

A. Mi lesz az `Előállít(14,14,[7,6,5,4,3,2,1])` hívás kimenete?

B. Mi lesz az `Előállít(5,11,[4,5,11,2,3,1,1,1])` hívás kimenete?

C. Mi lesz az `Előállít(10,4,[5,4,3,2,1])` hívás kimenete?

Az Országos Középiskolai Tanulmányi Versenyek megvalósulását az NTP-TMV-M-24 projekt támogatja

Kulturális és Innovációs Minisztérium

Nemzeti Tehetség Program

D. Igaz-e, hogy ha az `Előállít(A,B,S)` hívás eredménye **nem** `Hiba!`, akkor az `Előállít(B,A,S)` eredménye sem lesz az?

IGAZ HAMIS

E. Igaz-e, hogy ha az `Előállít(A,B,S)` hívás eredménye **nem** `Hiba!`, akkor az `SA` és `SB` tömbök összes elemeinek összege megegyezik `S` tömb elemeinek összegével?

IGAZ HAMIS

F. Igaz-e, hogy létezik olyan `S` tömb, amire **bármely** $1 \le A, B \le 20$ esetén **nem** lesz `Hiba!` az `Előállít(A,B,S)` hívás eredménye?

IGAZ HAMIS

2. feladat (80 pont)

Egy alagútrendszer `N` teremből áll. A termeket `1`-től `N`-ig számozzuk meg. Az `1`-es számú terem a kijárat a rendszerből. A termek közti közlekedés föld alatti járatokon keresztül lehetséges, ahol minden közvetlen járat pontosan két különböző termet köt össze.

Két robot, `A` és `B` az alábbi algoritmusokkal próbál eljutni a kijárathoz egy kezdeti `X` sorszámú teremből (a `VanJárat(S,T)` érték pontosan akkor igaz, ha van (közvetlen) járat `S` teremből `T` terembe):

```
KijutA(X):
  Ciklus i=X-től 1-ig
    Ha VanJárat(X,i) akkor
      X:=i;
  Ciklus vége
Eljárás vége
```

```
KijutB(X):
  Ciklus i=1-től N-ig
    Volt[i]:=HAMIS;
  Ciklus vége
  Volt[X]:=IGAZ;
  Ciklus amíg X>1
    Ciklus i=N-től 1-ig
      Ha VanJárat(X,i) akkor
        Ha nem Volt[i] akkor
          X:=i;
          Volt[X]:=IGAZ;
    Ciklus vége
  Ciklus vége
Eljárás vége
```

Vegyünk egy `N=8` teremből álló rendszert, melynek járatait az alábbi táblázat adja meg:

Terem	Melyik termekbe vezet járat
1	3, 4, 5
2	3, 4, 8
3	1, 2, 8
4	1, 2, 6
5	1, 6
6	4, 5, 7
7	6
8	2, 3

Add meg a két algoritmus esetén az alábbi táblázat kitöltésével, hogy kijut-e a robot az alagútrendszerből, valamint azt, hogy mely termeket látogatja meg (a látogatás sorrendjében kell megadni; ha nem jut ki a robot, akkor az elakadásáig kell felsorolni a termeket)!

Induló terem	`KijutA` kiér (IGEN/NEM)	`KijutA` útja	`KijutB` kiér (IGEN/NEM)	`KijutB` útja
1		1,		1,
2		2,		2,
3		3,		3,
4		4,		4,
5		5,		5,
6		6,		6,
7		7,		7,
8		8,		8,

<u>3. feladat</u> (70 pont)

Egy új találmány, a Kő-Papír-Olló Automata a nevezetes játékot a következő módon játssza. A gépnek van `N` különböző belső állapota, melyeket `1`-től `N`-ig számozunk. A gép kezdetben az `1`-es állapotban van. Minden körben a gép a pillanatnyi belső állapotához rendelt jelet mutatja, mely lehet kő (K), papír (P), vagy olló (O). Ezután attól függően, hogy az **ellenfél** mit mutatott a három jel közül, megváltoztatja a gép a saját belső állapotát. Az alábbi táblázatok két különböző automata leírását tartalmazzák.

1. automata:

Állapot	Mutatott jel	Következő állapot, ha az ellenfél…		
		K-t mutat	P-t mutat	O-t mutat
1	K	2	3	1
2	P	2	3	1
3	O	2	3	1

2. automata:

Állapot	Mutatott jel	Következő állapot, ha az ellenfél…		
		K-t mutat	P-t mutat	O-t mutat
1	P	2	3	3
2	O	4	2	4
3	K	1	4	3
4	P	1	4	3

A szokásos szabályok szerint a kő legyőzi az ollót, az olló a papírt és a papír a követ. Azonos jel mutatása esetén az eredmény döntetlen. Válaszolj az alábbi kérdésekre!

A. Ha az első automata ellen játszol, akkor milyen jeleket kell mutatnod az első 5 játékban, hogy mindegyiket megnyerd?

B. Ha a második automata ellen játszol, akkor milyen jeleket kell mutatnod az első 5 játékban, hogy mindegyiket megnyerd?

C. Ha a két automata egymás ellen játszik, akkor milyen jeleket fognak mutatni az első 5 játék során?

1. automata:
2. automata:

D. Ha a két automata egymás ellen játszik, akkor milyen eredmények fognak születni az első 100 játék során?

1. győzelmeinek száma:
2. győzelmeinek száma:
Döntetlenek száma:

<u>4. feladat</u> (60 pont)

Elektromos autóval haladunk egy hegyvidéki úton. Az út `N` pihenőhelyből áll, melyeket `1`-től `N`-ig számozunk és kezdetben az `1`-es számú pihenőhelyen vagyunk. El szeretnénk jutni az utolsó pihenőhöz, egyesével haladva pihenőről pihenőre.

Minden pihenőnek ismert a tengerszint feletti H_i magassága. Az egymást követő pihenők között az út egyenletesen emelkedik, vagy egyenletesen lejt. Amikor két pihenőhely között utazunk, akkor csökkenő magasság esetén minden egység magasságcsökkenésért egy egységgel nő az akkumulátorunk töltöttsége. Amikor viszont növekszik a magasság, akkor minden egység növekedésért egyel csökken az akkumulátor töltöttsége. Csak a kezdeti, `1`-es számú pihenőhelyen van töltőállomás, itt feltölthető az akkumulátorunk: egy egység töltés 1 percbe telik. Kezdetben az akkumulátor üres. Az autó üres akkumulátorral is képes elindulni és eljutni a következő pihenőhelyre, ha annak magassága kisebb a jelenlegi pihenőhelyénél.

A lehető legkevesebb ideig szeretnénk tölteni az akkumulátort ahhoz, hogy végig tudjunk menni a hegyi úton. Add meg az alábbi magasságértékek esetén, hogy minimum hány percet kell töltésre fordítanunk, mielőtt indulunk, az 1-es pihenőben!

Azt is határozd meg, hogy ebben az esetben minimum hány egység töltés egyidejű tárolására kell alkalmasnak lennie az akkumulátornak (azaz mi az a legkisebb `K` érték, aminél végig tudunk menni az úton az előbb meghatározott kezdeti töltéssel, feltéve, hogy az akkumulátor töltöttsége lejtőn lefelé haladva nem nő `K` fölé, hanem elérve azt `K` egységnél megáll egészen addig, amíg emelkedőn töltést nem veszít ismét)!

A. `H=[1,3,7,8,11,12,15]`

Töltési idő:	Minimum kapacitás:

B. `H=[8,20,7,21,15]`

Töltési idő:	Minimum kapacitás:

C. `H=[15,12,11,8,7,3,1]`

Töltési idő:	Minimum kapacitás:

D. `H=[20,24,15,10,18,21,23,20]`

Töltési idő:	Minimum kapacitás:

E. `H=[30,15,18,23,16,20,25,10,16,19]`

Töltési idő:	Minimum kapacitás:

5. feladat (80 pont)

Sári leejtette a számológépét és sajnos a gombok nagy része tönkrement az eséstől. Csak az `1`, `2`, `-` (kivonás) és `x` (szorzás) gombok maradtak működőképesek. Ráadásul azt is észrevette, hogy minden esetben, amikor lenyomja az egyik számot, a számológép azonnal kiértékeli és kiírja az aktuális művelet eredményét. Ha a szám megnyomását nem előzte meg műveleti jel beírása, akkor csak egyszerűen törli az eddigi kiírást és kiírja a most lenyomott számot a gép.

Például a `2-1x2x2` gombsorozat beírásakor az első `2`-es lenyomásakor megjelenik a `2`, az `1`-es lenyomásakor `2-1=1` lesz a kijelzőn, az második `2`-es lenyomásakor `1x2=2` lesz látható, míg végül a `2x2=4` szám fog a kijelzőn szerepelni. De például a `12-1` beírásakor a kijelzőn az `1`-es szám fog állni, mert a `2` szám lenyomásakor a kijelzőn törlődik az `1` és a `2` jelenik meg helyette.

Sári hamar rájött, hogy még ezzel a számológéppel is képes minden pozitív egész értéket előállítani. Most szeretné a számokat a lehető leggyorsabban, azaz a legkevesebb gombnyomással megkapni. A fenti példában láttuk, hogy a 4 megkapható 7 gombnyomással. Lehet, hogy ennél kevesebb gombnyomás is elegendő hozzá?

A. Add meg az első 15 pozitív egész mindegyikére, hogy legkevesebb hány gombnyomással állíthatók elő!

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

B. Minimum hány gombnyomás kell a 73 előállításához?

C. Minimum hány gombnyomás kell az 1629 előállításához?

6. feladat (50 pont)

Karaktersorozatok tömörítése előtt gyakran próbáljuk olyan módon átalakítani azokat, hogy a karakterek sorrendjének megváltoztatásával a lehető legtöbb azonos karakter kerüljön szomszédos pozíciókba, ami jelentősen növelheti a tömörítés hatásfokát. Az egyik módszer a karakterek átrendezésére a Burrows-Wheeler transzformáció, mely a következő lépésekből áll:

1. Írjunk a karaktersorozat végére egy egyedi új karaktert: esetünkben ez a `$` lesz. Példának tekintsük a `BANAN` karaktersorozatot, melyből ez a lépés a `BANAN$` karaktersorozatot állítja elő.

2. Soroljuk fel az eredeti karaktersorozat összes ciklikus eltoltját (melyeket az eredeti sorozat karaktereit valahány pozícióval körkörösen eltolva kapunk meg). A példában ezek:
```
BANAN$
$BANAN
N$BANA
AN$BAN
NAN$BA
ANAN$B
```
3. Rendezzük az eltolt karaktersorozatokat betűrendbe (a `$` karaktert minden betű és szám megelőzi):
   `ANAN$`**`B`**
   `AN$BA`**`N`**
   `BANAN`**`$`**
   `NAN$B`**`A`**
   `N$BAN`**`A`**
   `$BANA`**`N`**
4. Vegyük azt a karaktersorozatot, ami az eltoltak utolsó karaktereinek összeolvasásából adódik a rendezés után: ez lesz a transzformáció kimenete. A példában a `BN$AAN` karaktersorozatot kaptuk.

Válaszolj az alábbi kérdésekre!

A. Mi lesz a `FEKETE` karaktersorozat Burrows-Wheeler transzformáltja?

B. Melyik karaktersorozatból indultunk ki, ha a transzformáció eredménye: `D$FASAD`

C. Ha a transzformáció első lépését elhagyjuk (nem írunk `$` karaktert a bemenet végére), akkor több különböző karaktersorozatra is ugyanaz a kimenet adódik. Sorold fel az összes karaktersorozatot, aminek így ugyanaz lenne a transzformáltja, mint az `ABCD` karaktersorozatnak.