INFORMATIKA – tétel

2022

ELIGAZÍTÁS:

- Az 1-4 feladatokban (a pszeudokód programrészletekben): (1) a **`kiír ' '`** utasítás kiír a képernyőre egy szóközt; (2) a „**`/`**” operátor osztási hányadost ad meg, a „**`%`**” operátor pedig osztási maradékot.
- Az 5-9 feladatok esetében használj alprogramot, valahányszor célszerűnek találod. Törekedj hatékony megoldásra az algoritmusok időigénye tekintetében. **Lásd el beszédes kommentekkel programjaidat**.
- A bemeneti adatok helyesnek tekinthetők.

FELADATOK:

1. Legyen az alábbi pszeudokód programrészlet:

```
eredmeny = 0
┌amíg y ≠ 0 végezd
| eredmeny = eredmeny + x
| y = y - 1
└■
kiír eredmeny
```

Mit ír ki a fenti programrészlet, ha **`x=3`** és **`y=5`**? [1 pont]

15

2. Legyen az alábbi pszeudokód programrészlet:

```
eredmeny = 0
┌amíg x % y ≠ 0 végezd
| eredmeny = eredmeny + y % x
| x = x + 1
└■
kiír eredmeny
```

Mit ír ki a fenti programrészlet, ha **`x=3`** és **`y=7`**? [1 pont]

7

3. Legyen az alábbi pszeudokód programrészlet:

```
z = 0
┌amíg y ≠ 0 végezd
| z = y % 10
| y = y / 10
| ┌ha z ≤ x akkor
| |   x = z
| | különben
| |   x = x-1
| └■
└■
kiír x
```

Mit ír ki a fenti programrészlet, ha **`x=10`** és **`y=735`**? [1 pont]

2

4. Legyen az alábbi pszeudokód programrészlet:

```
┌amíg x ≠ 0 végezd
| y = x % 10
| x = x / 10
| ┌ha y%2 ≠ 0 akkor
| |   y = y + 1
| └■
| ┌amíg y > 0 ÉS y < 10 végezd
| | kiír y, ' '
| | y = y * 3
| └■
└■
```

Mit ír ki a fenti programrészlet, ha **x=2022**? [1 pont]

```
2 6 2 6 2 6
```

5. Írj Pascal vagy C/C++ **programot**, amely beolvassa billentyűzetről két személy születési, illetve elhalálozási évét, és kiírja a képernyőre, hogy volt-e olyan év („Igen” vagy „Nem”), amelyben mindketten éltek. (Az időszámításunk előtti éveket negatív számok jelölik) [1 pont]
Példa bement:
```
-2 33 33 98
```
Példa kimenet:
```
Igen
```

```
#include <iostream>
using namespace std;

int main()
{
     int sz1, sz2, h1, h2;
     cin >> sz1 >> h1 >> sz2 >> h2;
     if (h1 >= sz2 || h2 <= sz1){
          cout << "Igen.";
     }
     else {
          cout << "Nem.";
     }
     return 0;
}
```

6. Írj Pascal vagy C/C++ **függvényt**, amely 1-et vagy 0-t térít vissza attól függően, hogy a paraméterként kapott két természetes szám anagrammák vagy sem (ugyanazokat a számjegyeket tartalmazzák ugyanannyi mennyiségben). A számok nem lehetnek hosszabbak, mint 10 számjegy. [1 pont]
Példa bement:
```
1232 3122
```
Példa kimenet:
```
1
```

```
int anagrammak( int x, int y ){
    if ( !x && !y ){ return 1; }
    int stat[10] = {0};
    while( x && y ){
        ++stat[ x%10 ]; x /= 10;
        --stat[ y%10 ]; y /= 10;
    }
    if ( x || y ){ return 0; }
    for( int i = 0 ; i < 10 ; ++ i ){
        if( stat[i] ){ return 0; }
    }
    return 1;
}
```

7. Írj Pascal vagy C/C++ **programot**, amely a `vizsgalando.txt` állományból beolvassa az `n` értéket ($2 \le n \le 100$), valamint `n` darab 10000-nél kisebb természetes számot. A program írjon ki a képernyőre „`Igen`”-t vagy „`Nem`”-et attól függően, hogy a számsorozat Fibonacci-sorozathoz tartozó elemei szigorúan növekvő részsorozatot alkotnak vagy sem. A Fibonacci-sorozat a következőképp épül fel: $f_0=0,\ f_1=1,\ f_n=f_{n-1}+f_{n-2}$, ha $n>1$. (A sorozat tartalmaz legalább két Fibonacci számot) [1 pont]

Példa bement:

```
10
1 9 13 7 23 21 55 47 5 34
```

Példa kimenet:

```
Nem
```

```
#include <iostream>
#include <fstream>
#define N 10000
using namespace std;

int main()
{
    ifstream fin("vizsgalando.txt");
    int fibo[N] = {0}, elozo = 0, kurrens = 1, kovetkezo;
    fibo[0] = 1; fibo[1] = 1;
    while( ( kovetkezo = elozo + kurrens ) < N ){
        fibo[kovetkezo] = 1;
        elozo = kurrens;
        kurrens = kovetkezo;
    }
    int n, x, i;
    fin >> n;
    elozo = -1;
    for( i = 0 ; i < n ; ++i ){
        fin >> x;
        if( fibo[x] ){
            if( elozo == -1 ){ elozo = x; }
            else{
                if( x <= elozo ){ cout << "Nem"; break; }
                else { elozo = x; }
            }
        }
    }
    if( i >= n ){ cout << "Igen"; }
    return 0;
}
```

8. Írj Pascal vagy C/C++ **programot**, amely a `matrix.txt` állományból beolvassa az `n` értéket ($1 \le n \le 100$), valamint egy `n×n` méretű mátrixot (elemei 10000-nél nem nagyobb természetes számok). A program írja ki az `ujmatrix.txt` kimeneti állományba a következőképpen módosított mátrixot: (1) amely sorok tartalmazzák a mátrix maximumát, azon sorok értékeit növelje meg a mátrix minimumának értékével; (2) amely oszlopok tartalmazzák a mátrix minimumát, azon oszlopok értékeit csökkentse a mátrix maximumának értékével. [1 pont]

Példa bement:

```
4
2 4 6 3
4 7 2 3
3 4 2 6
5 3 2 7
```

Példa kimenet:

```
-5 4 -1 3
-1 9 -3 5
-4 4 -5 6
 0 5 -3 9
```

```
#include <fstream>
#define N 100
#define MAX 10000

using namespace std;

void NovelMaxsorokMinertekkel(int n, int a[][N], int sor_max_ind[], int
uj_max, int min_ertek);
void CsokkentMinoszlopokMaxertekkel(int n, int a[][N], int
oszlop_min_ind[], int uj_min, int max_ertek);
void KiirMatrix(int n, int a[][N]);

int main()
{
    ifstream fin("matrix.txt");
    int n, a[N][N], sor_max_ind[N]={0}, oszlop_min_ind[N]={0};
    int min_ertek = MAX+1, max_ertek = -1;
    int uj_min = 0, uj_max = 0;
    fin >> n;
    for( int i = 0 ; i < n ; ++i ){
        for( int j = 0 ; j < n ; ++j ){
            fin >> a[i][j];
            if( a[i][j] < min_ertek ){
                min_ertek = a[i][j];
                ++uj_min; oszlop_min_ind[j] = uj_min;
            }
            else if( a[i][j] == min_ertek ){ oszlop_min_ind[j] = uj_min; }
            if( a[i][j] > max_ertek ){
                max_ertek = a[i][j];
                ++uj_max; sor_max_ind[i] = uj_max;
            }
            else if( a[i][j] == max_ertek ){ sor_max_ind[i] = uj_max; }
        }
    }
    fin.close();
    NovelMaxsorokMinertekkel(n,a,sor_max_ind,uj_max,min_ertek);
    CsokkentMinoszlopokMaxertekkel(n,a,oszlop_min_ind,uj_min,max_ertek);
    KiirMatrix(n,a);
    return 0;
}
void NovelMaxsorokMinertekkel(int n, int a[][N], int sor_max_ind[], int
uj_max, int min_ertek){
    for( int i = 0 ; i < n ; ++i ){
        if( sor_max_ind[i] == uj_max ){
            for( int j = 0 ; j < n ; ++j ){
                a[i][j] += min_ertek;
            }
        }
    }
}
void CsokkentMinoszlopokMaxertekkel(int n, int a[][N], int
oszlop_min_ind[], int uj_min, int max_ertek){
    for( int j = 0 ; j < n ; ++j ){
        if( oszlop_min_ind[j] == uj_min ){
            for( int i = 0 ; i < n ; ++i ){
                a[i][j] -= max_ertek;
            }
        }
    }
}
void KiirMatrix(int n, int a[][N]){
    ofstream fout("ujmatrix.txt");
    for( int i = 0 ; i < n ; ++i ){
        for( int j = 0 ; j < n ; ++j ){
```

```
            fout << a[i][j] << ' ';
        }
        fout << endl;
    }
    fout.close();
}
```

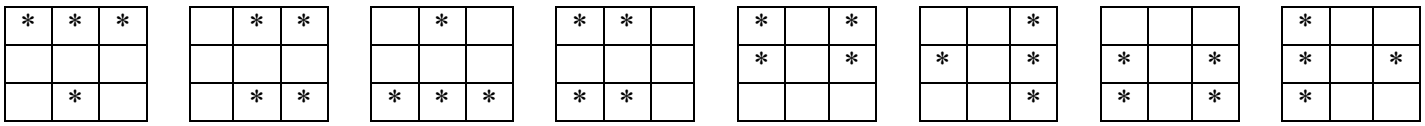
9. Írj Pascal vagy C/C++ **függvényt**, amely paraméterként megkapja az `n` értéket (`1≤n≤100`), és kiírja a képernyőre, hogy egy `n×n` méretű sakktáblára legtöbb hány futó helyezhető el úgy, hogy ne üssék egymást. A függvény írja ki azt is, hogy hány különböző módon helyezhető el a maximális számú futó az `n×n` méretű sakktáblán. (Két futó akkor üti egymás, ha ugyanazon átlón vannak) [1 pont]

Példa bement:

```
3
```

Példa kimenet:

```
4 8
```



$2n-2$ *és* 2^n

```
#include <iostream>
#include<cmath>

using namespace std;

int main()
{
    int n;
    cin >> n;
    switch(n){
        case 1:
            cout << "1" << " " << "1";
            break;
        default:
            cout << 2*n-2 << " " << pow(2,n);
    }
    return 0;
}
```