„A” melléklet

Programozási környezet

A csapatok számára rendelkezésre álló munkaállomásokon Windows és Linux operációs rendszerek, valamint az alábbi szoftverek lesznek elérhetők:

Windows alatt:

1. 64 bites Windows 10 Enterprise operációs rendszer
2. Microsoft Edge, Internet Explorer, Mozilla Firefox és Google Chrome böngészők
3. Total Commander 9.22a fájlkezelő
4. Notepad++ 7.7.1 szövegszerkesztő
5. Java SE 8 Update 221 és Java SE 12
6. MinGW gcc és g++ 6.3.0
7. Apache NetBeans IDE 11.1
8. Eclipse IDE 2019-06 (Java és C/C++ támogatással)
9. IntelliJ IDEA Community 2019.2.1
10. Microsoft Visual Studio Community 2019 (C/C++ és C# támogatással, .NET 4.8 keretrendszerrel)
11. Code::Blocks 17.12 (C/C++ támogatással)
12. Free Pascal 3.0.4
13. Python 3.7.0 és JetBrains PyCharm Community 2019.2.1 IDE

Linux alatt:

1. 64 bites Ubuntu 18.04.3 LTS Linux operációs rendszer
2. Gnome grafikus felhasználói felület
3. Mozilla Firefox böngésző
4. Midnight Commander fájlkezelő
5. mcedit, nano, vim, Sublime Text szövegszerkesztők
6. Java SE 8 Update 221 és Java SE 11
7. gcc és g++ 7.4.0
8. gdb nyomkövető
9. Apache NetBeans IDE 11.1
10. Eclipse IDE 2019-06 (Java és C/C++ támogatással)
11. IntelliJ IDEA Community 2019.2.1
12. Code::Blocks 16.01 (C/C++ támogatással)
13. Python 3.6.8

Az egyes nyelvek dokumentációi (C standard könyvtár, C++ STL, Java SE API, Free Pascal, Python 3) a verseny közben elérhetők lesznek. Felhívjuk a figyelmet arra, hogy a C# nyelvhez jelenleg nem tudunk dokumentációt biztosítani.

„B” melléklet

A versenyen használható programozási nyelvekről, fordítóprogramokról és a forráskódok lefordításához használt fordítási opciókról, valamint technikai specifikációkról

1. A verseny programozási nyelvei: Java 11, C, C++11, C#, Pascal és Python 3.
2. Egy feladat megoldását egyetlen, a forrásnyelven megírt kódot tartalmazó állományban (továbbiakban forráskód) kell beküldeni. A beküldött állomány nevének az alábbi kiterjesztések egyikével kell rendelkeznie:
   - Java nyelvű program esetén: `.java`
   - C nyelvű program esetén: `.c`
   - C++ nyelvű program esetén: `.cpp`
   - C# nyelvű program esetén: `.cs`
   - Pascal nyelvű program esetén: `.pas`
   - Python nyelvű program esetén: `.py`
3. A versenyzők által beküldött forráskódok az alábbi parancsokkal lesznek lefordítva:
   - Java (Java 11.0.1):
     ```
     javac -encoding UTF-8 <forrás.java>
     ```
   - C (gcc 8.2.0):
     ```
     gcc -std=c99 -Wall -lm -DONLINE_JUDGE -O3 <forrás.c>
     ```
   - C++ (g++ 8.2.0):
     ```
     g++ -std=c++11 -Wall -lm -DONLINE_JUDGE -O3 -msse2
         -static-libstdc++ -s <forrás.cpp>
     ```
   - C# (csc 2.9.0):
     ```
     csc <forrás.cs> /define:ONLINE_JUDGE /optimize
     ```
   - Pascal (Free Pascal 3.0.4):
     ```
     fpc -dONLINE_JUDGE -XS -O2 <forrás.pas>
     ```
   - Python (3.7.1):
     ```
     python -m py_compile <forrás.py>
     ```
4. A beküldött forráskód mérete nem haladhatja meg a 40 KB-ot.
5. A beküldés forráskódjából lefordított futtatható állomány mérete nem haladhatja meg az 5 MB-ot.
6. Minden programnak 512 MB memória áll rendelkezésére a futása során.
7. A feladat megoldását végző programnak a befejeződésekor zéró (0) visszatérési értéket kell adnia az operációs rendszer felé (lásd a „C” mellékletet), ellenkező esetben „Futási hiba” értékelést kap a beküldés.
8. A beküldött forráskód nem tartalmazhat olyan kódrészleteket, amelyek
   - a standard input és standard output csatornákon kívül más állományokat használnak,
   - egynél több szálat vagy folyamatot kezelnek,
   - hálózati szolgáltatásokat vesznek igénybe.

„C” melléklet

Programozási javaslatok a fordítási és futási hibák elkerülésére, illetve a programok futásának optimalizálására

A programkódokat beküldés előtt mindenképpen érdemes a „B” mellékletben megadott fordítási opciókkal lefordítani, elkerülendő a versenybizottság „Fordítási hiba” üzenetét.

A versenybizottság kiértékelő alkalmazásai figyelik a tesztelt beküldés által szolgáltatott visszatérési értéket. Ha ez az érték 0-tól különböző, a beküldés értékelése „Futási hiba” lesz. Egy program az operációs rendszer felé a következő módon tud 0 visszatérési értéket szolgáltatni:

- Java nyelven: **`System.exit(0);`** metódushívással a program tetszőleges pontján, ahol végrehajtható utasítás állhat.
- C, C++ nyelven: **`return 0;`** utasítással a main() függvényben, illetve **`exit(0);`** függvényhívással a program tetszőleges pontján, ahol végrehajtható utasítás állhat.
- C# nyelven: **`Environment.Exit(0);`** metódushívással a program tetszőleges pontján, ahol végrehajtható utasítás állhat.
- Pascal nyelven: **`Halt(0);`** eljáráshívással a program tetszőleges pontján, ahol végrehajtható utasítás állhat.
- Python nyelven: **`sys.exit(0)`** függvényhívással a program tetszőleges pontján, ahol végrehajtható utasítás állhat.

A Java, C#, Pascal és Python programokat nem kötelező a megadott utasításokkal befejezni, mivel azok – normál befejeződés esetén – amúgy is 0-val térnek vissza.

A C++ nyelven programozóknak javasoljuk, hogy a standard bemenetről történő adatbeolvasásra és a standard kimenetre történő adatkiíratásra a program futásának optimalizálása érdekében a C++ nyelvben megszokott **`cin`** és **`cout`** streamek helyett használják a C nyelv standard könyvtári függvényeit, a **`scanf()`** és **`printf()`** függvényeket! A **`cin`** és **`cout`** streameket használó programok kellően nagyméretű adathalmazok esetén mérhetően lassabban futnak!

A versenybizottságnak a beküldések teszteléséhez használt adatai a feladatokban leírt input (bemeneti) specifikációknak megfelelőek lesznek. A beküldéseknek csak a specifikációban megadott formátumú adatokat kell feldolgozniuk. Nem szükséges a beküldött programban ezek helyességét ellenőrizni, ez ugyanis lassíthatja a program futását.

A program által a standard kimenetre küldött adatok formátumának meg kell felelnie a probléma output (kimeneti) specifikációjában leírtaknak. A versenybizottság kiértékelő alkalmazásai nem fogadják el helyesnek például az olyan kimeneti eredményeket, amelyek rövidebbek vagy (akár csak egy üres sorral is) hosszabbak, mint az elvárt helyes kimenet, vagy amelyek valamely sora a kívánt helyes karaktersorozat után további szóköz karaktereket tartalmaz.

Különösen ellenjavallt a beküldött programban olyan, a felhasználónak szóló, ám a feladat kimeneti specifikációjában nem szereplő üzeneteknek a standard kimenetre történő kiíratása, mint például a „`Kérem a következő egész számot:`” vagy az „`Adja meg n és m értékét!`” üzenetek. Ezek ugyanis a begépelt formában jelennek meg a standard kimeneten, és emiatt nagy valószínűséggel nem fognak megfelelni a feladat kimeneti specifikációjában leírtaknak.

A programnak csak addig kell tárolnia a beolvasott bemeneti adatokat, ameddig azok feldolgozásra nem kerülnek; legtöbbször felesleges előre beolvasni az összes bemeneti adatot, és csak utána feldolgozni azokat. Legtöbbször ugyancsak felesleges a már kiszámított részeredményeket a memóriában tárolni, ha a további számításokhoz már nincsen szükség rájuk.

Ha a feladatnak a standard inputról kell adatokat beolvasnia, akkor a megoldást célszerű úgy tesztelni, hogy a tesztelésre szánt bemeneti adatokat előzetesen elmentjük egy szöveges állományba (`teszt.in`), majd egy parancsablakban kiadjuk a

```
program < teszt.in > teszt.out
```

parancsot. A `teszt.in` állomány létrehozásával a tesztelendő adatok ismételt begépelésétől kímélhetjük meg magunkat abban azt esetben, ha a program nem az elvárt eredményeket szolgáltatja valamely tesztesetre. A program lefutása után a kimeneti adatokat a `teszt.out` állományban találjuk.