Programozás Alapjai 2. ZH

14. feladatsor

Szoftverfejlesztés Tanszék

2023, Ősz

Feladat Töltsd le a bíróról a `minta.zip` állományt, majd tömörítsd ki! A `feladat.c` fájlban megtalálod a feladatok megoldás-kezdeményeit. Bővítsd ezt az alább olvasható feladatok alapján! Lehetőség szerint ellenőrizd megoldásod, majd töltsd fel a `feladat.c` fájlt a bíróra!

Kiértékelés A bíró lefordítja a programot, majd lefuttatja azt a feladat pontszámának megfelelő számú tesztesettel. Egy teszteset egy bemenet-kimenet pár, amely a megfelelő feladathoz készült. A teszteset akkor helyes, ha az adott bemenethez tartozó kimenet **minden egyes karaktere** megegyezik az előre eltárolt referencia kimenettel. *További feltételek: a program futása nem tarthat tovább 5 másodpercnél, egyszerre nem fogyaszthat többet 16 MiB memóriánál és nem történhet futási hiba (pl. illetéktelen memória hozzáférés).*

Ellenőrzés Feltöltés előtt érdemes ellenőrizni a megoldásod.

1. **Fordítás** Ellenőrizd, hogy a programod lefordul-e! A bíró a `gcc -O2 -static -o feladat feladat.c` paranccsal fordít, érdemes ezt használni. A `-Wall` kapcsoló is hasznos lehet.
2. **Példa tesztesetek** Ellenőrizd, hogy a programod helyesen működik-e! A `minta.zip` tartalmaz a bíró által futtatott tesztesetek közül feladatonként egyet-egyet. Az első feladat teszteléséhez másold a programod mellé az `ex1.be` fájlt `be.txt` néven, futtasd le a programod, majd az így kapott `ki.txt` tartalmát hasonlítsd össze az `ex1.ki` fájlban található referencia kimenettel.
3. **Extra tesztesetek** Ellenőrizd a programod működését további példák segítségével! Néhány további teszteset is elérhető, de ezek csupán ellenőrzésre használhatóak, a bíró nem futtatja őket. Ezek használatához futtasd a programod a `-t` vagy `-test` kapcsolóval, például a `./feladat -test` paranccsal. Csak az első feladat teszteléséhez futtasd a programod a `./feladat -t 1` paranccsal.

1. feladat (2 pont)
A feladat egy minimum függvény megírása. A függvény a három paramétere közül a legkisebb sorszámával tér vissza (`a`:1, `b`:2, `c`:3). A három kapott szám páronként különböző lesz.

Kódold le C nyelven a függvényt! A függvény fejlécén ne változtass! A függvény inputjai a paraméterek, outputja a visszatérési érték. A függvény nem végez IO műveleteket!

```
int minimum(int a, int b, int c);
```

2. feladat (3 pont)
A feladat meghatározni egy szám valódi osztóinak számát. A függvény egyetlen paramétere a kérdéses szám, amelyről meg kell mondani, hogy 1-en és önmagán kívül hány darab osztója van. A visszatérési érték a valódi osztók száma.

A számítás elvégezhető egy olyan ciklussal, ami 2-től $n/2$-ig megy, és minden lépésben megvizsgálja, hogy a ciklusváltozó aktuális értéke osztója-e a paraméternek, azaz elosztva vele a paramétert 0 maradékot kapunk-e. Ha igen, akkor eggyel növeljük az osztók számát tároló változó értékét, amivel majd a függvény a ciklus befejezése után visszatér.

Kódold le C nyelven a függvényt! A függvény fejlécén ne változtass! A függvény inputjai a paraméterek, outputja a visszatérési érték. A függvény nem végez IO műveleteket!

```
int osztokszama(int n);
```