Programozás Alapjai 7. próba feladat

1. 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: BUSZLOPÁS (3 pont)
Mindigis szerettem volna buszt vezetni, mert a busz nagyon kék. Ahhoz, hogy valakiből buszvezető válhasson, hosszú és nehéz út vezet. Namármost, nekem nincs arra időm, hogy ilyenekkel foglalkozzak, szóval inkább úgy döntöttem, hogy ellopok egy buszt a buszállomásról.

Készítsd el a buszszerzes függvényt, amely paraméterben egy int pointert vár. A függvény olvasson be a standard inputról egy egész számot, és tárolja el a pointer által mutatott értékben.

Nyilván csak pozitív járatok léteznek. A függvény visszatérési értéke mondja meg, hogy helyes adatot kaptunk-e. Amennyiben a járat pozitív, akkor térjünk vissza 1-gyel, különben 0-val.

2. feladat: TANKOLÁS (4 pont)
A buszállomásra érve több szimpatikus buszt is láttam, ami megfelelő lenne a célra. Mindegyik csak úgy hívogatott. Alapos számítások után a busz kiválasztott engem, mert a busz választja a sofőrt, és nem fordítva. Szóval odamentem a kiválasztott buszhoz, és beszálltam. Nyilván ellenőriztem az üzemanyagszintet. Sajnos nem volt elég benne, így a többi buszból átöntöttem a benzint.

Készítsd el a benzin függvényt, aminek három paramétere van, amelyek leírják 3 busznak a benzinmennyiségét. A függvény készítsen egy float pointert, ami által mutatott értékbe tedd bele a három buszban lévő benzinmennyiségének az összegét.

3. feladat: SOFŐRKÖDÉS (4 pont)
Kigurultam az utakra. Mármint nem szó szerint értem, mert én nem tudok gurulni, hanem a buszt kivezettem az útra. Akkor azonban jött egy probléma. Az utasok azt hitték, hogy a busz, amit vezetek, az egy tényleges, menetrend szerint közlekedő busz. Nyilván nem volt más választásam, muszáj volt elszállítanom az utasokat, mert különben gyanús lesz nekik az eset és a végén még feljelentenek indokulatlanul.

Már csak egy megálló volt hátra, amikor hátranéztem, de nem láttam senkit. Vajon tényleg nincs már senki a buszon?

Készítsd el az utasok függvényt, amelynek 4 paramétere van:

- egy megállókból álló tömb
- a megállók darabszámát, azaz a tömb méretét
- két pointert, amit kimeneti paraméterként használunk

A függvény módosítsa a két pointer által mutatott értékeket úgy, hogy abban rendre az összes leszálló ember száma, illetve az összes felszálló ember száma legyen.

A függvény adjon vissza 1-et, amennyiben a busz kiürült, 0-t egyébként. A megállók nem feltétlenül sorrendben jönnek. A buszon nem szállítunk kecskéket.

4. feladat: GAZDAGSÁG (4 pont)

Nem számítottam rá, de az akció során még pénzt is kerestem, ugyanis az utasok tőlem vásároltak jegyeket és napijegyeket. Én egy random bevásárlólista darabjait adtam oda nekik, de nem nagyon voltak szomjasak, így viszonylag könnyen meggyőzhetőek voltak, hogy ez így rendben van. Számoljuk össze, hogy mennyit kerestem az út során!

Készítsd el a nyereseg függvényt, ami paraméterben várja az eladott jegyek, illetve az eladott napijegyek számát. A függvény adjon vissza egy tömböt, amiben benne vannak az egyes vásárlások során elköltött összegek. A tömb elején a jegyvásárlások vannak, utána a napijegy vásárlások.

Példa:

input: 5, 3

output: 600, 600, 600, 600, 600, 1300, 1300, 1300

magyarázat: 5 db jegyet vettek, így a tömb 5 db 600-assal indul, majd utána 3 db 1300-as következik, mivel 3 db napijegyet vásároltak.

5. feladat: AUTÓS ÜLDÖZÉS (6 pont)

A rendőrség rájött a tettemre, és a nyomomba eredtek. Próbáltam menekülni, ahogy tudtam, de az előttem lévő útszakaszra csapdákat állítottak, hogy megakadályozzanak ebben. Ki kell kerülnöm a csapdákat. A busz szenzorai felmérték a területet, már csak olyan formába kell alakítani az adatokat, hogy megértsem, és aszerint tudjak menekülni. Segíts kérlek!

Készítsd el a csapda_atalakitas függvényt, ami első paraméterben egy tömböt vár, amelyben az előttem lévő útszakasz csapdássága látható (0/1 értékek). A második paraméter megmondja, hogy hány sávos az útszakasz, a harmadik pedig hogy milyen hosszú az útszakasz. A tömbben az elemek úgy vannak az elemek, hogy először a hozzám legközelebb álló útszakasz sávjai egymás után, utána az eggyel távolabbi, stb. A feladat, hogy átalakítsuk 2D-s tömbbé úgy, hogy az első index a sáv indexe legyen.

Példa:

| 1 1 1 |
| 0 0 1 |
| 0 1 0 |
| 1 1 0 |
| 0 0 1 |
| 1 0 0 |
| (bus) |

input: [1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0] (savok: 3, utszakaszok: 6)

output: [[1, 0, 0, 1, 0, 1], [1, 0, 1, 1, 0, 0], [1, 1, 0, 0, 1, 0]]

6. feladat: FÉNYMÁSOLÓ (4 pont)

A rendőrségnek sikerült megállítania a buszt. Nagyon ügyikék voltak. Kinyitottam az ajtót, rögtön letámadtak, hogy letartóztassanak. Én meg így mondom nekik, hogy ácsika nagyfiúk, lassítsatok már. Aztán lassítottak, én meg elmagyaráztam nekik, hogy igazából én nem is loptam el a buszt, mert az még most is az állomáson van. Nem igazán akartak nekem hinni, azt mondták fixen hülyének nézem őket. Én meg mondom figyuka már kiscsibék, eskü ott van, csak nézzétek meg az állomáson. Így végül magamra hagytak, mentem tovább, ők meg elindultak az állomás fele.

És nem vicceltem, mielőtt elloptam a buszt, tényleg lemásoltam, és a másolat most ott van az állomáson. A rendőrség megnyugodott, és onnantól kezdve békén hagytak.

Készítsd el a busz_masolas függvényt, amely paraméterben egy buszra mutató pointert vár, és lemásolja a buszt, tehát egy másik busz pointert kell készíteni, ami egy teljes másolata (deep copy) az eredeti busznak.