# Programozás Alapjai 3. házi 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** (5 pont)
A hollandok gátakkal hódítanak el területet a tengertől, és nem szeretik, ha ezeken a gátakon átcsapnak a hullámok. Egy régi gát felújítását a következő módon tervezik. A gát előtt egy ponton megmérik a hullámok magasságát, majd veszik azokat a hullámokat, melyek átcsapnak (azaz amelyek magasabbak a jelenlegi gátnál), és ezen hullámok magasságának az átlaga lesz az új gátmagasság. Például, ha van egy 20 méteres gát, és mérnek előtte [12, 32, 16, 40, 21] méter magas hullámokat, akkor a 2., 4., és 5. csap át, ezek értéke 32, 40, 21, aminek az átlaga (32+40+21)/3=31, azaz az új gát 31 méteres kell, hogy legyen.

Írjunk függvényt, mely kiszámítja az új gátmagasságot! Az első bemeneti paramétere egy egészekből álló tömb, aminek a lezáró eleme 0. Ez tartalmazza a hullámok magasságát. A másik bemeneti paraméter a jelenlegi gát magassága, ami egész szám. A függvény a gátnál magasabb hullámok átlagának egészrészével tér vissza. Ha egyetlen hullám sem csap át, akkor a függvény az eredeti gát magasságával tér vissza.

```
int atlag(int bemenet[], int magassag);
```