

# Programozás Alapjai 5. 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 tesztessel. Egy tesztet egy bemenet-kimenet pár, amely a megfelelő feladathoz készült. A tesztet 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 tesztet 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)

Írj egy `LEGO_lap` nevű struktúrát, ami az `a_oldal` és `b_oldal` nevű mezőiben egy LEGO lap két oldalát tárolja el. A LEGO lapok oldalhossza egész szám. Írj egy `rendez` függvényt, amelynek első paramétere egy, a LEGO lapok adatait tároló tömb, második paramétere pedig ezen tömb mérete (előjeles egész szám). A függvény állítsa terület szerint növekvő sorrendbe a `LEGO_lapok` tömb elemeit. Amennyiben két, vagy több lap területe megegyezik, akkor a rendezés tartsa meg az eredeti sorrendet. (Természetesen az elemeket ilyenkor is a megfelelő sorrendbe kell állítani csak az azonos területű lapok közötti sorrend ne változzon az eredetihez képest.)

```
void rendez(LEGO_lap LEGO_lapok[], int n);
```