Programozás II. 1. ZH

SZTE Szoftverfejlesztés Tanszék

2022. ősz

Technikai ismertető

- A programot C++ nyelven kell megírni.
- A megoldást a *Bíró* fogja kiértékelni.
  - A Feladat beadása felületen a Feltöltés gomb megnyomása után ki kell várni, amíg lefut a kiértékelés. **Kiértékelés közben nem szabad az oldalt frissíteni vagy a Feltöltés gombot újból megnyomni** különben feltöltési lehetőség veszik el!
- Feltöltés után a *Bíró* a programot `g++` fordítóval és a
  `-std=c++1y -static -O2 -DTEST_BIRO=1`
  paraméterezéssel fordítja és különböző tesztesetekre futtatja.
- A program működése akkor helyes, ha a tesztesetek futása nem tart tovább 5 másodpercnél és hiba nélkül (0 hibakóddal) fejeződik be, valamint a program működése a feladatkiírásnak megfelelő.
- A *Bíró* által a `riport.txt`-ben visszaadott lehetséges hibakódok:
  - Futási hiba 6: Memória- vagy időkorlát túllépés.
  - Futási hiba 8: Lebegőpontos hiba, például nullával való osztás.
  - Futási hiba 11: Memória-hozzáférési probléma, pl. tömb-túlindexelés, null pointer használat.
- A riport.txt és a fordítási log fájlok megtekinthetőek az alábbi módon:
  1. Az Eredmények megtekintése felületen a vizsgálandó próba új lapon való megnyitása
  2. A kapott url formátuma:
     `https://biro.inf.u-szeged.hu/Hallg/IBL302g-1/1/hXXXXXX/4/riport.txt`
  3. Az url-ből visszatörölve a 4-esig (`riport.txt` törlése) megkaphatók a 4-es próbálkozás adatai
- A programot 20 alkalommal lehet benyújtani, a megadott határidőig.
- A programban szerepelhet `main` függvény, amely a pontszámításkor nem lesz figyelembe véve. Azonban ha fordítási hibát okozó kód van benne az egész feladatsor 0 pontos lesz.

Általános követelmények, tudnivalók

- Csak a leírásban szereplő osztályokat, metódusokat és adattagokat kell megvalósítani, egyéb dolgokért nem jár plusz pont.
- Minden metódus, amelyik nem változtatja meg az objektumot, legyen konstans! Ha a paramétert nem változtatja a metódus, akkor a paraméter legyen konstans!
- string összehasonlításoknál az egyezés a pontos egyezést jelenti, azaz ha kis-nagy betűben térnek el, akkor már nem tekinthetők egyenlőnek (pl. a "piros" != "Piros")
- A leírásokban bemutat példákban a string-ek köré rakott idézőjelek nem részei az elvárt kimenetnek, azok csak a string határait jelölik. Például ha az szerepel, hogy a példa bemenetre az elvárt kimenet az, hogy `"3 alma"`, akkor az elvárt kimenet idézőjelek nélkül az `3 alma`, de a szóköz szükséges!
  - A tesztesetekben nem lesz ékezetes szöveg kiíratása.
- Az elvárt kimeneteknek karakterről karakterre olyan formátumúnak kell lennie, ami a feladatban le van írva (szóközöket és sortöréseket is beleértve).

Ügyelj rá, hogy minden olyan metódus konstans legyen, ami nem módosít az adattagok értékein! Ha egy metódus nem változtat a paraméterén, akkor az legyen konstans! A feladat megoldása során a feladatok az előre adott osztályok módosítását írhatják elő. Feltöltéskor ezeket az osztályokat is fel kell tölteni és a módosításokat is pontozhatja a bíró! Egyes tesztesetekben a bíró módosított osztályt is használhat ezen kiinduló osztályok helyett, ezzel tesztelve a valóban helyes működést!

1. Jarmu osztály (4 pont)

Készítsd el a `leszed` metódust, ami nem tér vissza semmivel, és egy előjeles egész számot vár paraméterül. A metódus az Jarmu osztályban ne legyen megvalósítva.**(2 pont)**

Módosítsd az Jarmu osztályt, hogy az adattagokat a leszármazott osztályok is lássák! **(1 pont)**

Biztosítsd, hogy bármilyen leszármazott törlése helyesen történjen meg, akkor is, ha azt Jarmu-ként törlik! **(1 pont)**

2. Motor (34 pont)

Készíts egy Motor osztályt, mely egy motort reprezentál amin különféle díszek (matricák) lehetnek! Lehessen Jarmu típusként hivatkozni rá! **(2 pont)**

Az oldalán lévő díszek egy dinamikusan foglalt, Diszek-et tartalmazó `diszek` nevű adattagban tárold le!

Készíts egy konstruktort, mely egy nevet és 2 unsigned értéket vár! Az első unsigned érték határozza meg, hogy alapból hány dísz fér el a motor oldalán. A második érték azt mondja meg, hogy mekkora a motor oldala. (Ennek jelentőségét lásd += operátor, és leszed metódus) Vezess be egy értéket, ami a motor oldalának szabad részét követi! A konstruktor foglalja le a megfelelő méretű tömböt! **(2 + 1)**

Definiáld felül a += operátort, mellyel egy díszt lehet hozzáadni a `diszek` tömbhöz! Ha a dísz megegyezik egy már tárolt dísszel, akkor ne történjen semmi! Ha ilyen dísz még nem szerepelt, akkor add hozzá a `diszek` tömbhöz! Ha a dísz már nem férne el a motor oldalán, akkor a motorról szedjük le az összes díszt egyszerre (egyetlen leszed hívás, lásd következő feladat), majd tároljuk el az új díszt a felszabadult tömb első helyén!
(1+5 pont)

Valósítsd meg a `leszed` metódust! Egy motorról képesek vagyunk leszedni az oldalán levő díszeket, egyszerre akár többet is, hogy pontosan mennyit a metódus paramétere adja meg. A legutóbb feltett dísszel kezdi, azaz azt szedjük le először, amit legutoljára az oldalára tettünk. Minden dísz rendelkezik egy mérettel, ami azt jelzi, hogy a dísz levétele után mennyivel nőtt a motor oldalának szabad része. Amikor a díszt leszedjük az permanensen növeli a motor oldalának szabad részét. Ha a motorról annyit vagy több díszt akarunk leszedni, mint amennyit lehetséges, akkor bővítsd a `diszek` tömböt, hogy 2 Disszel több férjen bele, és állítsd 0-ra motor szabad oldalának részét. Ha a motor kifogyna a díszekből mielőtt leszedtük volna az összeset,

ne történjen semmi különleges, a metódus haladjon tovább hiba nélkül! A leszedett díszek helyére kerülhessen új dísz! **(1+8 pont)**

Biztosítsd, hogy az osztályra a másolás helyesen legyen megvalósítva (deep-copy, erőforrás megosztás nélkül)! Mindkét tanult másolási lehetőségre legyen ez igaz! **(2+10 pont)**

A helyes erőforrás felszabadítás is legyen megoldva! **(0+2 pont)**