Programozás II. 1. ZH

SZTE Szoftverfejlesztés Tanszék

2023. 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++17 -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:
- 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).

Különös lakosok

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 Kveliminder szakadékban van egy titokzatos, lélegző vulkán, amiben picike fura lények élnek. Ők a ***Vulkaforrázók****. Feladatuk, hogy a vulkánt életben tartsák úgy, hogy apró drágaköveket gyűjtenek és dobnak a vulkán magjába, amiből az a táplálékot nyeri, és ezáltal életben marad.*

1. feladat: Vulkaforrázó (15 pont)

Készítsd el a `Vulkaforrazo` osztályt, amely egy vulkaforrázó lényt fog reprezentálni. **(1+0 pont)**

A vulkaforrázó adattagjait az alábbi táblázat mutatja be:

Adattag neve	Típusa	Jelentése	Getter neve	Setter neve
nev	string	a vulkaforrázó neve	get_nev	set_nev
kepesseg	string	a vulkaforrázó speciális képessége	get_kepesseg	set_kepesseg

Az adattagok csak az osztályból legyenek elérhetőek, de készíts hozzájuk getter és setter metódusokat a fent látható táblázat szerint. **(1+2 pont)**

Készíts az osztályhoz egy default konstruktort, ami a nevet *Magmamarok*ra, és a képességét *teleportalas*-ra állítja. **(0+2 pont)**

Készíts az osztályhoz egy paraméteres konstruktort is, amely a 2 adattag értékét várja paraméterben és inicializálja az adattagokat azok alapján. A konstruktor paramétereinek sorrendje: *nev*, *kepesseg* **(0+3 pont)**

Lehessen egy vulkaforrázót stringgé konvertálni (string konverziós operátor), amely visszaadja a vulkaforrázó adatait az alábbi formában:

{nev} egy vulkaforrazo, akinek specialis kepessege: {kepesseg}

A {nev} helyére a vulkaforrázó neve, míg a {kepesseg} helyére a speciális képessége kerüljön. **(1+2 pont)**

Definiáld felül az osztályban a *!* *operátor*t is, amely a vulkaforrázó speciális képességét törli, ami azt jelenti, hogy a *kepesseg* adattagot üres stringre állítja. A metódus térjen vissza a módosított objektummal. **(1+2 pont)**

2. feladat: Érző vulkán (13 pont)

Készítsd el a `Vulkan` osztályt, amely egy vulkánt reprezentál, ahol a vulkaforrázók élnek. **(1+0 pont)**

A vulkán adattagjait az alábbi táblázat mutatja be:

Adattag neve	Típusa	Jelentése	Getter neve
meret	int	a vulkán mérete, méterben	get_meret
vulkaforrazok	Vulkaforrazo[4]	a vulkánban élő vulkaforrázók	get_vulkaforrazok
vulkaforrazo_szam	unsigned	jelenleg hányan élnek a vulkánban	get_vulkaforrazo_szam

Az adattagok csak az osztályból és a gyerekosztályokból legyenek elérhetőek, de készíts hozzájuk getter metódusokat! **(0+1 pont)**

Az osztálynak legyen egy 1 paraméteres konstruktora, amely a vulkán méretét várja és az alapján inicializálja az adattagot. Kezdetben a vulkánban 0 vulkaforrázó él. Minden vulkán legalább 5 m magas. Ha ennél kisebb értéket kapunk, akkor az adattag értékét állítsuk 5-re. **(0+2 pont)**

Definiáld felül az osztályban a *prefix* ++ operátort, amely a vulkán méretét 1 méterrel növeli. A metódus úgy viselkedjen, ahogy egy prefix ++ operátortól megszokott. **(1+3 pont)**

Definiáld felül a `Vulkan` osztályban a += *operátor*t, amely segítségével vulkaforrázót lehet hozzáadni a vulkánhoz. Hozzáadáskor a paraméterben kapott vulkaforrázó a tömb legelső üres pozíciójára kerüljön!

Köztudott, hogy egy vulkánban élő vulkaforrázók nevének különbözőnek kell lennie. Amennyiben a tömb tartalmaz már olyan vulkaforrázót, amelynek neve megegyezik a paraméterben érkező vulkaforrázó nevével, akkor a metódus ne csináljon semmit.

Amennyiben a tömb megtelt, akkor az objektum ne változzon.

A metódus adjon vissza referenciát az aktuális (módosított) objektumra! **(1+4 pont)**