Programozás II. Gyakorló Feladat

SZTE Szoftverfejlesztés Tanszék

2023. ősz

Ismertető

- A programot C++ nyelven kell megírni.
- **A benyújtandó fájl neve kötelezően `feladat.cpp`.**
- 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://biro2.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.
- A megvalósított függvények semmit se írjanak ki a standard outputra!

Lokális tesztelés

A minta.zip tartalmaz egy kiindulási feladat.cpp-t, amit a megoldással kiegészítve lokális tesztelésre használhattok. A fordítás az előbbiekben leírt módon történjen. A fájl felépítése a következő.

```
#include <iostream>
#include <string>
#include <cassert>

using namespace std;

//////////////////////////
//Ide dolgozz!!
//////////////////////////

//=== Teszteles bekapcsolasa kikommentezessel
//#define TEST_alma
//=== Teszteles bekapcsolas vege

#if !defined TEST_BIRO
/*
Keszits egy fuggvenyt, ami visszaadja az alma sztringet!
*/
void test_alma(){
  #ifdef TEST_alma && !defined TEST_BIRO
   string s = alma();
   assert(s == "alma");
  #endif
}
int main(){
  test_alma();
}
#endif
```

Ha megoldottad az alma feladatot, úgy tudod tesztelni, ha kitörlöd a kommentjelet a `#define TEST_alma` sor elől. Ekkor **újrafordítás** után le fog futni a `test_alma()` függvény tartalma is. Ha a visszaadott sztring nem az elvárt, az assert() függvény ezt jelezni fogja. A **define-ok módosítása nem javasolt**, fordítási hibát idézhet elő a biro-n való teszteléskor! **A tesztelőkód nem végez teljes körű tesztelést!** Saját felelősségre bővíthető. A sikeres megoldás után a feladat.cpp tartalma (mely biro-ra is feltölthető):

```
#include <iostream>
#include <string>
#include <cassert>

using namespace std;

string alma(){
  return "alma";
}

//=== Teszteles bekapcsolasa kikommentezessel
#define TEST_alma
//=== Teszteles bekapcsolas vege

/*
```

```
Keszits egy fuggvenyt, ami visszaadja az alma sztringet!
*/
void test_alma(){
  #ifdef TEST_alma && !defined TEST_BIRO
  string s = alma();
  assert(s == "alma");
  #endif
}

int main(){
  test_alma();
}
```

Feladatsor

Néhány feladat megoldásához szükséges megvalósítani **az előző gyakorló feladatsor Telepes** osztályát vagy legalábbis annak egy minimalizált verzióját: legyenek benne a kívánt adattagok (1-es táblázat), getterek, setterek és a default konstruktor.

Adattag neve	Típusa	Jelentése	Getter neve	Setter neve
nev	std::string	A telepes neve	get_nev	set_nev
szul_bolygo	std::string	Születési bolygó	get_szul_bolygo	set_szul_bolygo
bolygo	std::string	Jelenlegi bolygó	get_bolygo	set_bolygo
ero	unsigned	Munkavégző képesség	get_ero	set_ero

1. táblázat. Telepes adattagok. Default értékek: üres sztringek illetve az ero adattag esetén 1

Bemelegítő feladat (2 pont)

Bővítsd ki a Telepes osztályt egy új, privát láthatóságú adattaggal! A default értéke false legyen. Inicializálja ezt is a default konstruktor!

Adattag neve	Típusa	Jelentése	Getter neve	Setter neve
vegan	bool	Vegán-e a telepes	is_vegan	set_vegan

2. táblázat. Új telepes adattag

1. feladat (4 pont)

Készíts egy Kolonia nevű osztályt, mely a telepesek által létesített kommunákat reprezentálják. Gazdasági okok miatt minden kolónia csak 25 telepes befogadására képes. A kolónia adattagjait az 3. táblázat foglalja össze. Mindegyik rendelkezzen a táblázat szerinti getterrel és setterrel.

Készítsd el a Kolonia osztály konstruktorait is!

Adattag neve	Típusa	Jelentése	Getter neve	Setter neve
nev	std::string	A kolónia neve	get_nev	set_nev
bolygo	std::string	A kolónia helye	get_bolygo	set_bolygo
letszam	unsigned	Telepesek aktuális száma	get_letszam	-
lakok	Telepes[ ]	Telepesek tömbje, 25 méretű	get_lakok	-

3. táblázat. Kolónia adattagok

- Legyen egy konstruktora, mely két string paraméterrel rendelkezik. Az első sztring alapján a kolónia neve, a második alapján a kolónia bolygója legyen beállítva. A letszam kezdetben legyen 0-ra állítva.
- Legyen egy default konstruktor is, mely a kolónia nevét és a bolygóját üres sztringre, a létszámot pedig 0-ra inicializálja.

2. feladat (3 pont)

Valósítsd meg a += operátort, hogy telepest lehessen hozzáadni a kolóniához.

```
Kolonia k;
Telepes t;
k+=t;
```

A telepes úgy legyen letárolva, hogy a bolygo adattagja egyezzen meg a kolónia bolygo adattagjával. Ha a 25 fős limit miatt már nem lehet újabb telepest a kolóniához adni, akkor a metódus pontosan ezt a sztringet írja ki a standard outputra: „A kolonia megtelt". A kiíratást sortörés kövesse.

3. feladat (3 pont)

Készíts egy globális függvényt a Kolonia osztályon kívül, amely megvizsgálja, hogy a kolónia vegán-e, többségben vannak-e a vegán életfelfogású lakók. A neve legyen `veganE`, egy `Kolonia` objektumot vár paraméterül. A függvény akkor ad vissza igaz értéket, ha kolónia lakóinak több, mint 50%-a vegán (a vegan adattag értéke igaz). Ha nincs egy lakó sem a kolóniában, akkor alapértelmezetten nem vegán (azaz false a függvény visszatérési értéke).

4. feladat (2 pont)

Fejleszd tovább a 2-es feladatban létrehozott += operátort! Az új telepes letárolása előtt azt is vizsgáld meg, hogy a telepes vegán orientációja egyezik-e a kolónia vegán orientációjával. Ha a kolónia vegán, akkor az új telepes letárolása csak akkor lehetséges, ha ő is vegán. Amennyiben nem vegán az alábbi hibaüzenet kerüljön a standard outputra (utána sortörés következzen): „A kolonia vegan".

Figyelem, a `veganE` használatához a függvény és/vagy a Kolonia osztály forward deklarációja szükséges. Linkek itt és itt. Alternatíva még a friend függvény használat.

5. feladat (4 pont)

Valósítsd meg a += operátort kolóniák összeolvasztására! A jobboldali paraméter tehát most egy kolónia objektum lesz. Az összevonás csak akkor lehetséges, ha a két kolónia azonos bolygón található, ehhez ellenőrizni kell a két kolónia bolygo adattagját. Két bolygó neve egyenlőnek számít, ha csak kisbetű/nagybetű eltérés van a nevükben (pl. „Mars" és „mars" egyenlőek), két üres sztring szintén egyenlő. (Ez ugyanaz a bolygónév összehasonlító logika, ami a Telepes feladatsorban volt) Ha a kolóniák bolygói nem egyeznek meg az alábbi hibaüzenet legyen kiírva (utána sortörés): „Hiba az egyesitesben: bolygo". Az összeolvasztás másik feltétele, hogy a befogadó kolóniában legyen legalább annyi szabad hely, amennyi az érkező kolónia létszáma. Ha kevés a hely ez a hibaüzenet legyen kiíratva (a kiíratás után sortörés következzen): „Hiba az egyesitesben: meret". A kolóniák vegán orientációjával itt nem kell foglalkozni. Ha lehetséges az összeolvasztás, akkor másoljuk át az új telepeseket a befogadó kolónia lakok tömbjébe az első szabad helytől kezdve.

6. feladat (3 pont)

Valósítsd meg az == operátort a Telepes osztályra. A metódus akkor adjon vissza igaz értéket, ha az alábbiak mindegyike teljesül a két operandusra :

- a telepesek neve karakterre pontosan megegyezik
- a telepesek erő értéke azonos
- a bolygó adattagok a kis- és nagybetűktől eltekintve megegyeznek (ugyanaz az egyenlőség feltétele, mint amit a 4. feladat ír)
- a születési bolygó adattagok a kis- és nagybetűktől eltekintve megegyeznek
- a vegan adattagok megegyeznek

7. feladat (4 pont)

Valósítsd meg a - operátort egy kolónia és egy telepes között, hogy telepest lehessen likvidálni a kolóniából. A baloldali operandus a kolónia, a jobboldali a telepes legyen:

```
Kolonia k;
Telepes t;
Kolonia k2 = k - t;
```

A likvidálás azt jelenti, hogy meg kell vizsgálni, hogy a kolónia lakói között van-e a paraméterben kapott telepessel egyező telepes (a == operátort felhasználva). Ha van ilyen telepes, akkor a lakok tömbben legelőször előfordulót törölni kell. A következő táblázatok mutatják a törlés menetét, a törlendő elem a telepes1:

0	1	2	3	4	
telepes0	telepes1	telepes1	telepes2	-	...

4. táblázat. Törlés előtt, létszám 4

0	1	2	3	4	
telepes0	telepes1	telepes2	-	-	...

5. táblázat. Törlés után, létszám 3

Tehát a törlés során eggyel balra kell shiftelni a lakókat tároló tömb tartalmát, az aktuális létszám karbantartásával pedig jelezni kell, hol ér véget a tömb.