Programozás II. ZH

SZTE Szoftverfejlesztés Tanszék

2024. 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++20 -Wall -Werror -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 bemutatott példákban a stringek 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).
- Ha az objektum másolása nem triviális (azaz a fordító által generált másolás nem elegendő), akkor a megfelelő másolást is meg kell valósítani.

Kiindulási projekt, megoldás feltöltése

- **A megoldáshoz az előre kiadott osztályok módosítása szükséges lehet.**
  - Nem minden ZH esetében van kiindulási projekt.
- 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!

Zh alatt használható segédanyag

- **A ZH során használható segédanyag elérhető bíróban.**
  - https://biro.inf.u-szeged.hu/kozos/prog2/

Kiindulás

A kiindulási feladatban kapott osztály módosítása vagy bővítése szükséges lehet!

`Uzenet` osztály

A `Uzenet` osztály egy üzenetet reprezentál, aminek van feladója és szövege.

- A konstruktorban meg lehessen adni a feladót és a szöveget is.
- A `getUzenet` metódus adja vissza az üzenetet `"<felado>: <uzenet>"` formában.

`MeetingMeghivas` osztály

Ez egy olyan speciális üzenet, ami egy meeting meghívást tartalmaz.

- A meeting időpontját most egyszerűen egy nemnegatív egész szám ábrázolja.
- A feladót, a szöveget és az időpontot meg lehessen adni a konstruktorban (ebben a sorrendben).
- Definiálja felül a `getUzenet` metódust úgy, hogy az Uzenet osztály megvalósítását kiegészíti egy vesszővel elválasztott időponttal (`"<felado>: <uzenet>, <idopont>"`).

TitkosUzenet

Ez egy titkosított üzenetet valósít meg.

- Be lehessen állítani, hogy az üzenet titkosítva van-e (true) vagy nincs (false).
  - Ezt is meg lehessen adni a konstruktorban harmadik paraméternek.
- Definiálja felül a `getUzenet` metódust.
  - Ha az üzenet nincs titkosítva, akkor úgy kell viselkednie, mint az ősosztály metódusa.
  - Ha titkosítva van, akkor a feladó és a uyenetet is titkosítva kell kiírni, azaz a karaktereket `*`-ra kell cserélni. Például: `"***: **********"`

Feladat

`Postafiok` osztály

A `Postafiok` tarolja az üzeneteket.

- Legyen default konstruktora.
- A `<<` operátorral lehessen `Uzenet`-et hozzáadni, és a postafióknak el kell tárolni.
  - Az üzeneteket érkezési sorrendben tárolja.
  - A `Postafiok` posztály **felel az így kapott üzenet megszüntetéséért**.

  - Az operátor legyen láncba fűzhető.

- Legyen egy `kiir` metódusa, ami soronként kiírja a postafiók tartalmát!
  - Minden üzenetet külön sorba írjon ki.
  - A titkos üzenetet úgy kell kiírni, amilyen állapotban van.
- Ki lehessen írni az osztályt a stream-re.
  - Minden üzenetet külön sorba írjon ki.
  - A titkos üzenetet úgy kell kiírni, amilyen állapotban van.
- A >> operátor segítségével elolvashatom az üzeneteimet.
  - Megadom, hogy hány üzenetet szeretnék elolvasni (egész szám), és a végéről visszafelé haladva „elolvasom" az üzeneteket, ami azt jelenti, hogy ki kell írni az üzenetet, majd **törölni kell** azt.
  - Ha kevesebb üzenet van, mint amekkora számot paraméterben kap, akkor csak annyi üzenetet kell kiírni, amennyi van, és nem is kell hibaüzenet adni.
- A << operátor segítségével át lehet irányítani egy másik postafiókot, és akkor annak az üzeneteit átveszi a `Postafiok`.
  - Átvételkor a másik postafiókban tárolt üzeneteket eredeti sorrendjükben kell átvenni, kezdve a legrégebbivel.
  - Titkos üzenetet csak akkor kell átvenni, ha az nincs titkosítva, különben ki kell hagyni.
  - Az eredeti helyen is meg kell hagyni az üzeneteket.
- Gondoskodj a megfelelő memóriafelszabadításról.