Programozás I. Gyakorló feladatsor

SZTE Szoftverfejlesztés Tanszék

2023. tavasz

Általános követelmények, tudnivalók

- A feladat elkészítési határideje: **vasárnap 23:59:59**. Ez szigorú határidő, a Bíró előre megadott időben zár, pótlásra nincs lehetőség.
- A feladatokat számítógép előtt kell megoldani, tetszőleges fejlesztői környezetben, tetszőleges operációs rendszer segítségével.
- Az elkészült programot **20** alkalommal lehet benyújtani, a megadott határidőig.
- Csak a leírásban szereplő osztályokat, metódusokat és adattagokat kell megvalósítani, egyéb dolgokért nem jár plusz ont.
- A feladat megoldása során minden megadott előírást pontosan követni kell! Tehát, ha a feladat leírása szerint egy adattag neve a "elsoFoku", akkor az alábbi elnevezések nem megfelelőek: "elsőFokú", "elsofoku", "elso_foku", "elsőFoq". Ugyanez igaz a metódusok, osztályok elnevezésére is!
- A metódusok esetében a visszatérési típus, a név, módosítók és a paraméterek típusai (és azok sorrendje) kerülnek ellenőrzésre, azonban a paraméterek nevei tetszőlegesek lehetnek.
- Az órán tanult konvenciókat követni kell (getter/setter elnevezés, toString, indentálás, stb). Abban az esetben is, ha ezt a feladat külön nem emeli ki, az ellenőrzés során erre is építünk.
- A nem forduló kódok nem kerülnek kiértékelésre, ezt utólagosan a gyakorlatvezető sem bírálhatja felül. (Hiszen mindenki rendelkezésére áll a saját környezete, ahol fordítani, futtatni tudja a forráskódot, így feltöltés előtt ezt mindenképpen érdemes megnézni!)
- Az adattagok és konstruktorok hiányában garantáltan 0 pontos lesz a kiértékelés, mert ezek minden teszt alapját képezik.
- Ha végtelen ciklus van a programban, akkor ezt a Bíró ki fogja dobni 3 másodperc után (ha többször is meghívásra kerül ilyen metódus, akkor ez többszöri 3 másodperc, összesen akár 2 perc is lehet). Ilyenkor NE kattints még egyszer a *Feltöltés* gombra, mert akkor kifagyhat a Bíró, csak a böngésző újraindításával lehet megoldani a problémát (emellett elveszik 1 feltöltési lehetőség is).
- Kérdés/probléma esetén a gyakorlatvezetők tudnak segítséget nyújtani.
- **A feladat megoldása során a default csomagba dolgozz, majd a kész forrásfájlokat tömörítve, zip formátumban töltsd fel, azonban a zip fájlt tetszőlegesen elnevezheted!**

  - Zip készítése: Windowson és Linuxon is lehet a GUI-ban jobb klikkes módszerrel tömörített állományt létrehozni (Windowsban pl. a 7-Zip nevű ingyenes program használatával).
  - Linux terminálon belül például a "zip feladat.zip *.java" paranccsal is elkészíthető a megfelelő állomány.

- A feladatokban az alábbi dolgok az alapértelmezettek (**kivéve**, ha a feladat szövege mást mond)
  - az osztályok láthatósága publikus
  - az egész érték 32 bites
  - a lebegőpontos számok dupla pontosságúak
  - az olyan metódusok void visszatéréssel rendelkeznek, amelyeknél nincs specifikálva visszatérési típus.
  - a metódusok mindenki számára láthatóak
  - az adattagok csak az adott osztályban legyenek elérhetőek

- 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/IB204L/FELADAT/hXXXXXX/4/riport.txt`
  3. Az url-ből visszatörölve a 4-esig (`riport.txt` törlése) megkaphatók a 4-es próbálkozás adatai.

- Szövegek összehasonlításánál 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 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!

- Az elvárt kimeneteknek karakterről karakterre olyan formátumúnak kell lennie, ami a feladatban le van írva (szóközöket és sortöréseket is beleértve).

Bevásárlólista

A feladat egy bevásárlólistát reprezentáló szoftver egyszerűsített változatának megvalósítása.

1. Termék (2 pont)

Készítsd el a `Product` adatosztályt! Az adatosztályoknak nincsen viselkedése, csak a megfelelő adattagok egységbezárását valósítják meg. Az adattagokat, valamint a szükséges metódusokat a **??**. ábrán láthatjuk. Ügyelj a megfelelő láthatóságok használatára!

Az osztálynak egyetlen paraméteres konstruktora legyen, amely ezeket az adattagokat várja paraméterként ugyanebben a sorrendben. **(1 pont)**

Készíts publikus lekérdező és beállító metódusokat minden adattaghoz! **(1 pont)**

2. Bevásárlólista (9 pont)

Készítsd el a `ShoppingList` nevű osztályt! Ez az osztály fogja végezni a különböző vásárolandó termékek nyilvántartását. Az adattagokat, valamint a szükséges metódusokat a **??**. ábrán láthatjuk. Ügyelj a megfelelő láthatóságok használatára! Az adattag típusánál interfészt adj meg, ne konkrét lista megvalósítást!

Legyen egy default konstruktora, amely üres listával inicializálja az *items* adattagot. A listához tömbbel megvalósított listatípust használj! **(1 pont)**

Írd meg az `addProduct` metódust, mely nem tér vissza semmivel és egy `Product` paramétert vár. A metódus adja hozzá az *items* listához a paraméterben kapott terméket! **(1 pont)**

Írd meg a `countProducts` metódust, amely nem vár paramétert és visszatér az *items* lista elemszámával! **(1 pont)**

Írd meg a `getProduct` metódust, amely paraméterben egy egész számot vár, és egy `Product` objektummal tér vissza. A metódus adja vissza az *items* lista paraméterben szereplő indexén lévő elemet, ha az létezik. Ha nincs annyi elem, amekkora indexet kapott paraméterben, vagy negatív indexet kapott, akkor térjen vissza null-lal! **(2 pont)**

Írd meg a `printProducts` metódust, amely nem vár paramétert és nincs is visszatérési értéke. A metódus írja ki a standard kimenetre az *items* listán szereplő termékek nevét, mindegyiket új sorba. **(1 pont)**

Írd meg a `delete` metódust, amely paraméterben egy `Product` objektumot vár, visszatérési értéke nincs. A metódus törölje az *items* listáról a paraméterben kapott terméket! **(1 pont)**

Írd meg a `deleteUnimportant` metódust, amely nem vár paramétert és egy egész számmal tér vissza. A metódus törölje az *items* listáról azokat a `Product` objektumokat, amelyek nem fontosak (az *important* adattagjuk hamis). Végül térjen vissza a törölt elemek számával! **(2 pont)**

3. Kedvenc üzletek (6 pont)

Készítsd el a `FavouriteShopSet` osztályt. Ez az osztály a felhasználó kedvenc üzleteit tudja kezelni. Az adattagokat, valamint a szükséges metódusokat a **??**. ábrán láthatjuk. Ügyelj a megfelelő láthatóságok használatára! Az adattag típusánál interfészt adj meg, ne konkrét halmaz megvalósítást!

Legyen egy default konstruktora, amely üres halmazzal inicializálja a *shops* adattagot. A halmazhoz piros-fekete fával megvalósított halmaztípust használj! **(1 pont)**

Írd meg az `addShop` metódust, mely nem tér vissza semmivel és egy szöveg paramétert vár. A metódus adja hozzá a *shops* halmazhoz a paraméterben kapott boltot! **(1 pont)**

Írd meg a `countShops` metódust, amely nem vár paramétert és visszatér a *shops* halmaz elemszámával! **(1 pont)**

Írd meg az `isFavourite` metódust, amely egy szöveget vár paraméterül és egy logikai értékkel tér vissza. A metódus eldönti, hogy a paraméterben kapott bolt a felhasználó kedvenc-e vagy sem. Azaz eleme-e a kedvenc boltok halmazának. **(1 pont)**

Írd meg a `printShops` metódust, amely nem vár paramétert és nincs is visszatérési értéke. A metódus írja ki a standard kimenetre a *shops* halmazban szereplő boltok nevét, mindegyiket új sorba. **(1 pont)**

Írd meg a `delete` metódust, amely paraméterben egy szöveget vár, visszatérési értéke nincs. A metódus törölje a *shops* halmazból a paraméterben kapott boltot! **(1 pont)**

4. Legjobb vétel (7 pont)

Készítsd el a `BestPriceFinder` nevű osztályt! Ez az osztály egy adott termék nevéhez meg tudja mondani, hogy melyik üzletben kapható a legolcsóbban. Az adattagokat, valamint a szükséges metódusokat a **??**. ábrán láthatjuk. Ügyelj a megfelelő láthatóságok használatára! Az adattag típusánál interfészt adj meg, ne konkrét leképezés megvalósítást!

Legyen egy default konstruktora, amely üres leképezéssel inicializálja az *bestBuys* adattagot. A leképezéshez hash-sel megvalósított leképezéstípust használj! **(1 pont)**

Írd meg az `addProduct` metódust, mely nem tér vissza semmivel és egy `Product` típusú, illetve egy szöveges paramétert vár. A metódus adja hozzá a *bestBuys* leképezéshez a paraméterben kapott terméket, a következő módon:

- a kulcs a termék neve (*name* adattag)
- az érték a paraméterben kapott szöveg (a bolt)

(1 pont)

Írd meg a `getShopFor` metódust, amely paraméterben szöveget vár, és egy szöveggel tér vissza. A metódus adja vissza a paraméterben kapott termék nevéhez tartozó boltot, ha az létezik. Ha a leképezés nem tartalmazza a paraméterben kapott kulcsot, akkor térjen vissza null-lal! **(1 pont)**

Írd meg a `printBestBuys` metódust, amely nem vár paramétert és nincs is visszatérési értéke. A metódus írja ki a standard kimenetre a *bestBuys* leképezés kulcsait és értékeit a következő formában:

- "Buy {productName} at {shopName}."
- A productName helyére a kulcs, a shopName helyére az érték kerüljön behelyettesítésre!
- Minden kiírás végén legyen sortörés!

(2 pont)

Írd meg a `deleteShop` metódust, amely paraméterben egy szöveget vár, visszatérési értéke pedig egy egész szám. A metódus törölje a *bestBuys* leképezésből azokat a termékeket (kulcsokat), amelyek értéke a paraméterben kapott szöveg, azaz az adott boltban lehet a legolcsóbban megvenni. A metódus térjen vissza a törölt bejegyzések számával. **(2 pont)**

5. Extrém vásárlás (4 pont)

Készítsd el a `ExtremeShopping` nevű osztályt! Ez az osztály az extrém vásárlók igényeit szolgálja ki. A bevásárló listánkat boltonként csoportosíthatjuk, hogy hol szeretnénk azt megvenni. Az adattagokat, valamint a szükséges metódusokat a **??**. ábrán láthatjuk. Ügyelj a megfelelő láthatóságok használatára! Az adattag típusánál interfészt adj meg, ne konkrét leképezés megvalósítást!

Legyen egy default konstruktora, amely üres leképezéssel inicializálja az *extremeList* adattagot. A leképezéshez hash-sel megvalósított leképezéstípust használj! **(1 pont)**

Írd meg az `addShoppingList` metódust, mely nem tér vissza semmivel és egy szöveget, illetve egy `Product` objektumokat tartalmazó listát vár paraméterül. A metódus adja hozzá az *extremeList* leképezéshez a paraméterben kapott bevásárlólistát, a következő módon:

- a kulcs a paraméterben kapott szöveg (a bolt)
- az érték a paraméterben kapott termékeket tartalmazó lista

(1 pont)

Írd meg a `printShoppingLists` metódust, amely nem vár paramétert és nincs is visszatérési értéke. A metódus írja ki a standard kimenetre az *extremeList* leképezés kulcsait és értékeit a következő formában:

- Az első sorba kerüljön a bolt neve (a kulcs).
- Ezt egy sortörés kövesse.
- Ezután a kulcshoz tartozó bevásárló lista elemeinek nevét soroljuk fel szóközökkel elválasztva.
- Ügyelj arra, hogy az utolsó termék neve után ne legyen szóköz!
- A terméklista után is legyen sortörés!

Például:
Lidl
tej cukor liszt
Tesco
alma szilva barack

(2 pont)

Jó munkát!

1. ábra. UML osztálydiagramok