Objektum orientált programozás
7. gyakorlat
Öröklődés, felüldefiniálás, referenciák típusai

1. Definiáljon saját csomagban (*myproducts*) **Áru** (*Product*) osztályt.
Adattagjai: név, nettó ár (egész), áfakulcs (egész, százalék)
Konstruktora: mindhárom adat megadásával inicializálja az adattagokat.
Metódusai:

- Számítsa ki a bruttóárát egy árunak (egész, kerekítve).
- Egy sztringbe összefűzve adja vissza az áru nevét és a bruttó árát. Definiálja felül az Object osztálybeli toString metódust!
- Növelje meg az áru nettó árát a paraméterben megadott százalékos értékkel (egész).
- Hasonlítsa össze két áru bruttó árát. 1-et adjon vissza, ha az áru drágább mint a paraméterben megadott; -1-et ha olcsóbb és 0-t ha azonos árúak.

Készítsen az Áru osztállyal azonos csomagban Kenyér osztályt, amely az Áru leszármazottja.
Adattagja: mennyiség (valós, pl. 0.75)
Konstruktora: négy paraméterben kapott adattal inicializálja az objektumot.
Metódusai:

- Definiálja felül a toString metódust. Az ősosztályban definiált metódushoz képest fűzze hozzá az egységárat (bruttóár/mennyiség).
- Adja vissza a mennyiséget.
- Hasonlítsa össze két kenyér egységárát. Az osztályszintű metódus akkor adjon vissza igazat, ha a paraméterként kapott két Kenyér közül az elsőnek nagyobb az egységára.

a) Készítsen egy futtatható osztályt a *myproducts* alcsomagjában. Ebben hozzon létre egy Áru és egy Kenyér objektumot. Írja ki az adataikat, és hasonlítsa össze az árukat. Melyik osztály metódusával lehetséges az összehasonlítás?
b) Hozzon létre még egy Kenyér típusú objektumot, de ennek a referenciáját egy Áru típusú változóban tárolja (neve: product2). Próbálja ki, hogy a "product2" referenciára meghívva milyen eredményt ad a toString metódus. Meg tudja-e hívni ezen a referencián keresztül a Kenyér mennyiségét visszaadó metódust?
c) Hozzon létre még egy Kenyér példányt (neve: bread2) és a "product2" referenciával hivatkozott másik Kenyérrel összehasonlítva a nagyobb egységárút kell kiírni (konvertálásra lesz szükség).

2. Készítsen saját csomagban **Személy** osztályt (pl. *mypersons.Person*).
Adattagjai: név, kor
Konstruktora: két paraméterben kapott értékkel inicializálja az adattagokat
Metódusai:
- Getter metódusok az adattagok lekérdezésére.
- A Személy adatait String-ben összefűző és ezzel visszatérő metódus.
- Adjon vissza igazat, ha a paraméterben megadott személy életkora kisebb.

Készítsen ugyanebben a csomagban **Gyermek** és **Felnőtt** osztályokat a Személy osztály leszármaztatásával.
Adattagjaik: iskola (a Gyermek osztályban, String) ; munkahely (a Felnőtt osztályban, String)
Konstruktoraik: három paraméterben kapott értékkel inicializálják az adattagokat
Metódusaik:
- Mindkét osztályban definiálja felül az adatokat String-ben összefűző öröklött metódust.
- Adja vissza a Gyermek iskoláját, ill. a Felnőtt munkahelyét.

Készítsen futtatható osztályt az előbb definiált osztályokat magába foglaló csomag alcsomagjában (pl. *mypersons.test.PersonTest*). Ebben deklaráljon két személy típusú referenciát, amelyek értéke a személy beolvasó függvény által visszaadott referencia lesz.
Írjon személy beolvasó függvényt, amelyben beolvassa egy személy adatait (név, kor). A referencia dinamikus típusát ellenőrzött módon határozza meg (ha 18 év alatti akkor Gyermek, egyébként Felnőtt), majd értelemszerűen olvassa be a Gyermek iskoláját ill. a Felnőtt munkahelyét. Hozza létre az objektumot, majd térjen vissza a referenciájával.
A main metódusban kérdezze le a létrejött objektumok típusát (*instanceof*) és irassa ki az adataikat.

A Felnőtt osztály kiterjesztéseként hozzon létre ugyanabban a csomagban egy **Alkalmazott** osztályt (pl. *mypersons.Employee*).
Adattagja: fizetés (egész), nyugdíjkorhatár (osztályszintű, egész: értéke 65)
Konstruktora: négy paraméterben kapott értékkel inicializálja az adattagokat (név, életkor, munkahely, fizetés).
Metódusai:
- Adja vissza a fizetést.
- Definiálja felül az adatokat String-ben összefűző öröklött metódust.

Készítsen új futtatható osztályt a *mypersons.test* alcsomagban, és másolja ide az előző futtatható osztályát. Amikor meghatározza a Személy típusúra deklarált referenciák dinamikus típusát, a Felnőttek esetén ellenőrizze a munkahely értékét: csak akkor lehet Alkalmazott, ha a munkahely nem üres (nem minden felnőtt alkalmazott). Ha alkalmazottról van szó, akkor a fizetését is olvassa be. Ezek után kérdezze le a létrejött objektumok típusát (*instanceof*) és irassa ki az adataikat.
Annak vizsgálata, hogy a munkhely értéke (String) üres-e:

```
workplace.isEmpty()
workplace.length() == 0
workplace.equals("")
```

Módosítsa úgy a futtatható osztályban a main metódus kódját, hogy hozzon létre egy n elemű Személy tömböt. N értékét olvassa be ellenőrzött módon. Az adatok beolvasásakor dőljön el, hogy az adott személy Gyermek, Felnőtt, vagy Alkalmazott. Majd életkoruk szerint növekvően rendezve írja ki az adataikat. Használja a minimum kiválasztásos rendező eljárást.

Házi feladat (Github classroomban):
A Könyv osztály leszármazottjaként készítsen egy másik csomagban KönyvStílus osztályt.
Adattagjai: stílus (szöveg)
Konstruktora: minden adatát paraméterben kapott adatokkal inicializálja.
Metódusai:
- Adja vissza a stílust.
- Definiálja felül az ősosztálybeli String-et visszaadó metódust úgy, hogy az ősbeli sztringhez fűzze hozzá a stílust is.

Készítsen egy futtatható osztályt a KönyvStílus-t magában foglaló csomag alcsomagjában. Ebben olvasson be n darab stílusos könyvet egy tömbbe. N-et ellenőrzött módon olvassa be (1 és 10 közötti érték).
a) Számolja meg hányféle különböző stílusú könyv szerepel a tömbben.
b) Írja ki a "Scifi" stílusú könyvek adatait.
c) Számítsa ki a "Scifi" stílusú könyvek átlagárát.
A String-ek egyezésének vizsgálatára használja a String osztály metódusát: `boolean equalsIgnoreCase(String anotherString)`