Programozási nyelvek Java

4. gyakorlat

További módosítószavak

static kulcsszó

- Az adattagokból (egyedváltozókból) minden objektum sajáttal rendelkezik
- Ha azonban egy adattag elé kiírjuk a static kulcsszót, akkor abból ún. osztályváltozó vagy statikus adattag lesz, amiből osztályonként csak 1 van.
  Tehát hiába létezik több ilyen típusú objektum, akkor is pontosan 1 darab statikus változója lesz.
- Mivel a statikus adattagok nem objektumhoz, hanem osztályhoz köthetőek, így az osztály nevével is hivatkozhatunk rájuk: <osztály neve>.<változó neve>
- Static kulcsszót metódusokhoz is megadhatunk.
- Picivel hatékonyabb (gyorsabb) a statikus függvények meghívása

final kulcsszó

- final kulcsszóval ellátott adattagok csak egyszer kaphatnak értéket
- paraméterek is elláthatóak ezzel a kulcsszóval
- a final és static kulcsszavakkal ellátott adattagok állnak legközelebb Java-ban a szoftvertechnológiában szereplő konstans kifejezésekhez
- final kulcsszóval ellátott metódusok nem definiálhatóak felül
- final kulcsszóval ellátott osztályokból nem lehet örököltetni

Interfész

- Egy interfész egy osztály látható (publikus) műveleteinek a leírása a belső szerkezet megadása nélkül, azaz csak függvény szignatúrák
- Nem tartoznak hozzá adattagok és implementáció
- Összes művelete publikus
- Ha egy osztály megfelel egy interfésznek, akkor annak minden műveletét deklarálnia és implementálnia kell

- Egy osztály egyszerre akár több interfészt is támogathat. Ha egy művelet több interfészben is szerepel, akkor jelentésüknek meg kell egyezniük, különben konfliktust okoznak.
- Az interfészek között is fennállhat öröklődési kapcsolat. Ekkor a származtatott interfész átveszi az ősinterfész összes metódusát
- Konvenció: „I” betűvel kezdődik, és a névben szereplő összes szó első betűje is nagybetű

Egy interfészimplementáció

```
public interface IValasztasBlokk {

        public abstract void setDeafult(Valasztas v);
        public abstract Valasztas ertek();
}
```

Megfelelés egy interfésznek Java-ban

```
/*
        A Person osztály implementálja az IPerson interfészt,
        azaz tartalmazza az interfész minden metódusának
        implementációját
*/
public class Person implements IPerson {

        /* Person osztály törzse */
}
```

Feladatok

1. Írjatok egy **Person** osztályt, aminek a konstruktorában át kell adni az ember nevét, nemét (férfi == true) és életkorát. Írjátok meg a megfelelő getter/setter metódusokat. Ezután készítsetek egy tesztelő osztályt, amiben létrehoztok egy embert és lekéritek néhány adatát.
2. Írjatok a Person osztályba egy **marry** függvényt, aminek paraméterében át kell adni egy másik személyt. A függvény térjen vissza egy logikai változóval, hogy sikeres volt-e a házasság. Azonos nemű házasságot nem engedélyezzük. Továbbá vegyetek fel a Person osztályba egy logikai változót, ami azt tárolja, hogy az adott személy házas-e. Aki már házas, ő ne tudjon ismét megházasodni.
   Egészítsétek ki a tesztelő osztályt, hogy az új metódust is tesztelje.
3. Írjátok át az előző feladatot úgy, hogy a program ne csak azt tárolja el, hogy valaki házas, hanem azt is, hogy ki a partnere.
4. Írjatok egy **Car** osztályt, ami egy autót reprezentál. Az autó létrehozásakor (konstruktorában) meg kell adni egy színt (String), kereket (Wheel osztály egy példánya) és kilométeróra (Milemeter egy példánya).
   Készítsétek el a **Wheel** osztályt. Kötelező legyen megadni a felni átmérőjét (col-ban), és hogy alufelni-e.
   Továbbá készítsetek egy **Milemeter** osztályt is, ami vár egy egész számot paraméterként, a maximális sebességet.
   Az összes osztály a **car** csomagban legyen.
   Írjátok meg a megfelelő getter/setter metódusokat. Az autón keresztül is le lehessen kérdezni a részeinek az adattagjait.
   Teszteljétek a programot a honlapon levő tesztfájllal (CarTester.java).