Zadaci za vježbu iz teme 4 (Nasljeđivanje. Polimorfizam.)

1. Napravite klasu `Dessert` koja ima sljedeće privatne atribute: `name` (String), `weight` (double) i `calories` (int). Dodajte klasi konstruktor koji prima kao parametre vrijednosti za sva 3 atributa. Napravite `get` i set metode za svaki atribut, te nadjačajte metodu `toString`. Napišite i javnu metodu `getDessertType` koja nema argumenata a vraća string „dessert“.

2. Napravite klase `Cake` i `IceCream` koje nasljeđuju `Dessert`. Kolač, uz sve atribute iz klase `Dessert` ima i atribute `containsGluten` (boolean) i `cakeType` (String, može biti „birthday“, „wedding“, „regular“ i sl.). Sladoled ima dodatne atribute `flavour` (String) i `color` (String). Napravite `get` i set metode za svaki atribut, kao i metodu `toString` koja vraća sve što vraća i metoda `toString` iz Klase `Dessert`, a dodatno još i atribute specifične za izvedenu klasu. Napišite javnu metodu `getDessertType` u svakoj od izvedenih klasa, koja će za sladoled vratiti tekst „ice cream“, a za tortu vrijednost atributa `cakeType` i tekst „ cake“. Napišite glavni program s kojim ćete testirati sve zadane funkcionalnosti.

3. Napravite klasu `Person` koja opisuje neku osobu. `Person` sadrži atribute `name` (String) `surname` (String), `age` (int). Napišite konstruktor, get i set metode, te metode `toString` i `equals` (dvije osobe su jednake ako imaju isto ime i prezime te broj godina).

4. Napravite klase `Student` i `Teacher` koje nasljeđuju klasu `Person`, `Student` sadrži atribut studentId (String) i academicYear (short int), a `Teacher` sadrži atribute email (String), subject (String) i salary (double). Napišite konstruktore za sve parametre, get i set metode, te metode toString i equals (dva studenta su jednaka ako imaju isti studentId, neovisno o ostalim podacima, a dva nastavnika su jednaka ako imaju isti email, neovisno o ostalim podacima). Dodatno, u klasi `Teacher` napišite metodu `increaseSalary` koja ne vraća ništa, a prima jedan argument tipa integer (koji predstavlja postotak). Metoda treba povećati plaću nastavnika za zadani postotak. Također, napišite i statičku metodu `increaseSalary` koja prima varijabilni broj argumenata, prvi je argument tipa integer (koji predstavlja postotak), a ostali su objekti tipa `Teacher` kojima je potrebno povećati plaću za zadani postotak.

5. Napišite glavni program u kojem ćete kreirati polje od 5 osoba i u njega staviti 3 nastavnika i dva studenta. Nakon toga program treba u petlji ispisati ime i prezime svake osobe te na kraju petlje prosječnu plaću svih nastavnika koji se pojavljuju u polju. Također, za sljedeći isječak koda:

```
Person p1 = new Person("Ivo","Ivic", 20);
Person p2 = new Person("Ivo","Ivic", 20);
Person p3 = new Student("Ivo","Ivic", 20, "0036312123", (short)3);
Person p4 = new Student("Marko","Marić", 25, "0036312123", (short)5);

System.out.println(p1.equals(p2));
System.out.println(p1.equals(p3));
System.out.println(p3.equals(p4));
```

Očekuje se ovakav ispis:

```
true
false
true
```

6. Nastavnici se natječu u fakultetskom „Master Chef“ natjecanju, u kojem svaki nastavnik priprema jedan desert, a studenti ih ocjenjuju. Za to ćemo napraviti klasu `CompetitionEntry` koja sadrži referencu na jedan objekt tipa `Teacher` (osoba koja je pripremila desert), referencu na jedan objekt tipa `Dessert`, te polje referenci na studente koji su ocijenili dani desert i polje ocjena koje su dali (pretpostavite da 3 studenta ocjenjuju jedan desert). Napišite konstruktor koji prima referencu na nastavnika i desert, sve gettere te metodu `addRating` koja ima parametre `Student` i cijeli broj (grade), a vraća boolean ovisno o tome je li uspjela ili ne ubaciti novi zapis u dani `CompetitionEntry` (najviše tri ratinga i studenti se ne smiju ponavljati). Napišite i metodu `getRating` koja vraća prosječnu ocjenu svih studenata koji su ocijenili neki `CompetitionEntry`.

7. Napravite klasu `UniMasterChef` koja sadrži polje referenci tipa `CompetitionEntry`, metodu `addEntry` (po uzoru na metodu `addRating`), metodu `getBestDessert` koja će vratiti najbolje ocijenjeni desert, kao i statičku metodu `getInvolvedPeople` koja prima argument tipa `CompetitionEntry`, a vraća referencu na polje osoba koje su sudjelovale u izradi ili ocjenjivanju kolača. Konstruktor za `UniMasterChef` prima jedan cijeli broj (broj prijava na natjecanje). Napišite metodu `main` u kojoj ćete testirati danu funkcionalnost.

   Za isječak koda:

```
Dessert genericDessert = new Dessert("Chocolate Mousse", 120, 300);
Cake cake = new Cake("Raspberry chocolate cake #3", 350.5, 400, false, "birthday");

Teacher t1 = new Teacher("Dario", "Tušek", 42, "dario.tusek@fer.hr", "OOP", 10000);
Teacher t2 = new Teacher("Doris", "Bezmalinović", 43, "doris.bezmalinovic@fer.hr", "OOP", 10000);

Student s1 = new Student("Janko", "Horvat", 18, "0036312123", (short)1);
Student s2  = new Student("Ana", "Kovač", 19, "0036387656", (short)2);
Student s3  = new Student("Ivana", "Stanić", 19, "0036392357", (short)1);

UniMasterChef competition = new UniMasterChef(2);

CompetitionEntry e1 = new CompetitionEntry(t1, genericDessert);
competition.addEntry(e1);
System.out.println("Entry 1 rating: " + e1.getRating());

e1.addRating(s1, 4);
e1.addRating(s2, 5);
System.out.println("Entry 1 rating: " + e1.getRating());

CompetitionEntry e2 = new CompetitionEntry(t2, cake);
e2.addRating(s1, 4);
e2.addRating(s3, 5);
e2.addRating(s2, 5);
competition.addEntry(e2);
System.out.println("Entry 2 rating: " + e2.getRating());

System.out.println("Best dessert is: " + competition.getBestDessert().getName());

Person[] e2persons = UniMasterChef.getInvolvedPeople(e2);

for (Person p : e2persons)
       System.out.println(p);
```

   Očekuje se sljedeći ipis:

```
Entry 1 rating: 0.0
Entry 1 rating: 4.5
Entry 2 rating: 4.666666666666667
Best dessert is: Raspberry chocolate cake #3
Doris Bezmalinović, age=43, email=doris.bezmalinovic@fer.hr
Janko Horvat, age=18, studentId=0036312123, academicYear=1
Ivana Stanić, age=19, studentId=0036392357, academicYear=1
Ana Kovač, age=19, studentId=0036387656, academicYear=2
```

8. Modeliramo potrebne klase i funkcionalnost za Rent-a-car kuću. Kompanija iznajmljuje vozila iz sljedećih kategorija: automobili, kombi vozila i limuzine. Napravite klasu `Vehicle` koja opisuje neko vozilo. Klasa ima sljedeće privatne atribute: `regNo` (String), `model` (String), `year` (int) i `price` (double, označava cijenu unajmljivanja vozila po satu). Dodajte klasi konstruktor koji prima kao parametre sve vrijednosti za atribute. Napravite `get` i `set` metode za svaki atribut, kao i metodu `toString`.

9. Napravite klase `Car`, `Van` i `Limo` koje nasljeđuju `Vehicle`. Dodatno, `Car` sadrži atribut `carType`(String), `Van` sadrži atribut `height`(double) i `noOfSeats`(int), a `Limo` sadrži atribute `length`(double) te boolean varijable `miniBar` i `sunRoof`. Napravite klase `PassengerVan` i `CargoVan` koje nasljeđuju klasu `Van`, a dodatno imaju privatne atribute `noOfPassengers`(int, za `PassengerVan`) tj. `maxLoad` (double, u kg, za `CargoVan`). Napravite `get` i `set` metode za svaki atribut svih klasa.

10. U svim klasama napišite metodu `getPricePerDay`, i to tako da, za objekte tipa `Vehicle` i `Car` metoda vraća cijenu iz varijable price pomnoženu s 24, za putnička kombi vozila vraća 80% cijene po danu pomnožene s 24, za teretna kombi vozila 110% cijene * 24, a za limuzine 150% cijene * 24. Također, u klasi `Vehicle` napišite metodu `getPricePerMonth`, koja će vratiti cijenu vozila ako se unajmljuje na mjesec dana (30 * odgovarajuća cijena po danu). Ovu metodu označite kao `final`. U klasi `Vehicle` napišite i statičku metodu `newestVehicle` koja prima varijabilni broj parametara tipa `Vehicle`, a vraća vozilo koje ima najnoviju godinu proizvodnje (ako ih je više, vrati prvo po redu). Dodajte argument `double cargoSpace` u klasu `Car` (označava volumen prtljažnika). U klasi `Vehicle` napišite statičku metodu `printAllVehiclesWithCargoSpaceGreaterThan` koja kao prvi argument prima double broj koji predstavlja traženu zapremninu, a nakon toga slijedi varijabilni broj argumenata tipa `Vehicle`. Metoda treba ispisati sva vozila koja imaju zapremninu teretnog prostora ili prtljažnika veću od tražene.

    Za isječak koda koji slijedi:

```
Vehicle v = new Car("DA1234AA", "Renault Clio", 2001, 20, "coupe", 200);
Car car = new Car("DA8818BB", "Renault Megane Grandtour", 2007, 25, "caravan",800);
Van van1 = new CargoVan("DA0009PO", "Volkswagen Transporter", 2018, 28, 2, (short)3, 4500);
PassengerVan van2 = new PassengerVan("DA6282EA", "IMV 1600", 1978, 35, 2, (short)3, 0);
Vehicle limo = new Limo("DA2238AB", "Zastava 750 LE", 1983, 220, 3.2, false, false);

System.out.println(v.getModel() + " price per day: " + v.getPricePerDay());
System.out.println(van1.getModel() + " price per day: " + van1.getPricePerDay());
System.out.println(van2.getModel() + " price per month: " + van2.getPricePerMonth());

Vehicle newest = Vehicle.getNewestVehicle(v, car, van1, van2, limo);
System.out.println("Newest: " + newest.getModel() + ", " + newest.getYear());

Vehicle.printAllVehiclesWithCargoSpaceGreaterThan(500, v, car, van1, van2, limo);
```

    očekuje se ovakav ispis:

```
Renault Clio price per day: 480.0
Volkswagen Transporter price per day: 739.2
IMV 1600 price per month: 20160.0
Newest: Volkswagen Transporter, 2018
Wehicles with cargo space greater than 500.0 litres:
 - Renault Megane Grandtour: 800.0
 - Volkswagen Transporter: 4500.0
```

Rješenja zadataka dostupna su na sljedećim poveznicama:

1. https://github.com/FER-OOP/Lectures/blob/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e01/Dessert.java
2. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e02
3. https://github.com/FER-OOP/Lectures/blob/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e03/Person.java
4. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e04
5. https://github.com/FER-OOP/Lectures/blob/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e05/MainClass.java
6. https://github.com/FER-OOP/Lectures/blob/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e06/CompetitionEntry.java
7. https://github.com/FER-OOP/Lectures/blob/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e07/UniMasterChef.java
8. https://github.com/FER-OOP/Lectures/blob/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e08/Vehicle.java
9. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e09
10. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-04/src/main/java/hr/fer/oop/homework_04/e10