Zadaci za vježbu iz teme 5 (Apstraktne klase. Sučelja.)

1. Napravite klasu `Computer`, te `Desktop` i `Laptop` koje nasljeđuju `Computer`. Klasa `Computer` sadrži atribute `model, manufacturer i operatingSystem`(svi `String`), `Desktop` dodatno sadrži atribut `caseHeight` (`double`), dok `Laptop` sadrži atribute `batteryCapacity` (`int`) i `weight`(`double`). Napišite konstruktore za sve atribute, `get` i `set` metode, te metodu `toString` u sve tri klase. Napišite metodu `main` u kojoj ćete stvoriti po jedan objekt svake klase i ispisati njihove podatke.

2. Prepravite kod iz 1. zadatka tako da nije moguće stvoriti objekt tipa `Computer`. Također, dodajte apstraktnu metodu `getComputerType` u klasu `Computer` (metoda ne prima argumente, a vraća `String` „desktop computer“ ili „laptop computer“, ovisno o kojoj vrsti objekta je riječ). Napišite i metodu `calculatePortabilityScore` (nema argumenata, a vraća `int`), koja će vratiti kategoriju prenosivosti nekog uređaja, i to na sljedeći način: za laptope se vrati težina uređaja zaokružena na cijeli broj, a za desktop računala formula je: 5 + `caseHeight`/30. Osigurajte da svako računalo mora implementirati metodu `calculatePortabilityScore`.

3. Napravite klasu `Netbook` koja nasljeđuje `Laptop`. Metoda `calculatePortabilityScore` u ovoj klasi uvijek vraća score 1, a metoda `getComputerType` vraća „netbook laptop computer“. Dodatno, onemogućite daljnje nasljeđivanje klase `Netbook`.

   Za isječak koda:

```
Computer n = new Netbook("Ideapad S12", "Lenovo", "Windows", 50, 1.55);

System.out.println(n);
System.out.println(n.getComputerType());
System.out.println(n.calculatePortabilityScore());

Laptop l = (Laptop) n;
System.out.println(l.calculatePortabilityScore());
```

   Očekuje se ovakav ispis:

```
Ideapad S12, manufacturer=Lenovo, operatingSystem=Windows, batteryCapacity=50, weight=1.55
netbook laptop computer
1
1
```

4. Napravite klasu `Device` koja opisuje neki generički uređaj. Neka `Computer` nasljeđuje `Device`. Kakva zbog toga mora biti klasa `Device`? Ima li smisla neke atribute iz `Computer` preseliti u `Device`? Također, dodajte i klasu `Mp3Player` koja nasljeđuje `Device`, a ima atribute `batteryCapacity`(int) i `memorySize` (int). Uočite da su neki od uređaja prenosivi, npr. `Mp3Player`, `Laptop` i `NetBook`. Osigurajte da svi prenosivi uređaji implementiraju metode `getModel`() i `getBatteryCapacity`().

5. Napravite klasu `Person` koja opisuje neku osobu. `Person` sadrži atribute id (int) i `name` (String). Napravite klase `Student` i `Teacher` koje nasljeđuju klasu `Person`, `Student` sadrži atribut academicYear (short int) i polje ocjena koje je student dobio na ispitima (pretpostavimo da je mogao dobiti najviše 5 ocjena, ocjene su cijeli brojevi od 0 do 5) a `Teacher` sadrži atribute subject (String) i teacherGrade(double), što predstavlja ocjenu nastavnika na studentskoj anketi. Napišite konstruktore za sve atribute, get i set metode, te metodu toString u sve tri klase. Napišite metodu main u kojoj ćete stvoriti po jedan objekt svake klase i ispisati njihove podatke.

6. U klasi Person napišite metodu getGrade (koju ustvari ne možete/ne znate napisati u toj klasi) i osigurajte da se metoda mora implementirati u klasama koje nasljeđuju Person. Ovo nužno zahtijeva još neku promjenu u klasi Person, koju? Za studenta, metoda će vratiti prosječnu ocjenu svih položenih ispita, a za nastavnika ocjenu iz studentske ankete. Također, napišite i metodu `public final boolean isOutstanding()` i smjestite je u klasu Person. Metoda će vratiti true ako je ocjena te osobe veća od 4.5.

   Za isječak koda:

```
Person s1 = new Student(111, "Pero Perić", (short)3, new int[] {5,3,4});
Student s2 = new Student(115, "Ana Anić", (short)1, new int[] {5,5,4});
Teacher t = new Teacher(615, "Mirko Mandić", "OOP", 4.83);

Person[] people = new Person[] {s1, s2, t};

System.out.println("Outstanding students and teachers:");
for (Person p : people)
      if (p.isOutstanding())
            System.out.println(p);
```

   Očekuje se ovakav ispis:

```
Outstanding students and teachers:
115, Ana Anić, academicYear=1, grades=[5, 5, 4]
615, Mirko Mandić, subject=OOP, teacherGrade=4.83
```

7. Napravite klase `Vehicle`, `Car`, `Van` i `Limo`, `slično kao i u prošlim vježbama. Vehicle` ima privatne atribute: `registrationNo` (String) i `model` (String). `Car`, `Van` i `Limo nasljeđuju Vehicle. Car` sadrži atribute `carType`(String), `noOfSeats`(int) i `cargoSpace(double, zapremnina prtljažnika u litrama)`, `Van` sadrži atribut `height`(double), a `Limo` sadrži atribute `length`(double), `noOfSeats`(int) te boolean varijablu `sunRoof`. Napravite klase `PassengerVan` i `CargoVan` koje nasljeđuju `Van`, a dodatno imaju privatne atribute `noOfPassengers` (int, za `PassengerVan`) tj. `maxSpace` (double, u litrama, za `CargoVan`). Napravite konstruktore, `get` i `set` metode za svaki atribut svih klasa te metodu toString u svakoj od klasa. Onemogućite stvaranje objekata tipa Vehicle i Van. Neke od navedenih klasa vozila pogodne su za prijevoz tereta. Osigurajte da sva vozila koja mogu prevoziti terete implementiraju metodu getMaxSpace(), a sva vozila koja prevoze putnike implementiraju metodu `getMaxPassengers()`, koje vraćaju `maxSpace` ili `cargoSpace`, `tj. numberOfSeats` ili `noOfPassengers`, ovisno o klasi. Uočite da neka vozila mogu prevoziti i putnike i teret.
   Za isječak koda:

```
Vehicle car = new Car("DA8818BB", "Renault Megane Grandtour", "caravan", 4, 800);
PassengerVan van3 = new PassengerVan("DA6282EA", "IMV 1600", 212, 8);
Limo limo = new Limo("DA2238AB", "Zastava 750 LE", 320, 4, false);

PassengerVehicle[] passengerVehicles =
      new PassengerVehicle[] {limo, (PassengerVehicle) car, van3}; //Zašto cast na car?

for (PassengerVehicle pv : passengerVehicles) {
      System.out.println(pv);
      if (pv instanceof CargoVehicle) {
        System.out.println(" - this passenger vehicle can also transport cargo!");
        System.out.println("   max cargo space: " + ((CargoVehicle)pv).getMaxSpace());
      }
}
```

   Očekuje se ovakav ispis:

```
DA2238AB, Zastava 750 LE, length=320.0, noOfSeats=4, sunRoof=false
DA8818BB, Renault Megane Grandtour, carType=caravan, noOfSeats=4, cargoSpace=800.0
 - this passenger vehicle can also transport cargo!
   max cargo space: 800.0
DA6282EA, IMV 1600, height=212.0, noOfPassengers=8
```

8. Potrebno je osmisliti model podataka za vođenje fundusa školske knjižnice. Napravite klasu LibraryItem koja će imati atribute `id` (`int`) i `name` (`String`). Napravite klase Book, `Magazine`, `DigitalItem` i `EItem` koje nasljeđuju LibraryItem. Book ima atribut `author` (`String`), `Magazine` ima atribute `contentCategory` (`String`) i `issue` (`int`), `DigitalItem` opisuje neki digitalni medij za pohranu podataka i ima atribut `type` (`String`, npr. „DVD“ ili Blueray), dok `EItem` opisuje sadržaj koji je moguće preuzeti direktno na mobilni uređaj ili e-reader, a dodatno ima atribut url (URL, pogledajte https://docs.oracle.com/en/java/javase/15/docs/api/java.base/java/net/URL.html). Konačno, napravite klase `TextBook` i `Dictionary` koje nasljeđuju Book. `TextBook` ima atribut `subject` (`String`), a Dictionary atribute `firstLanguage` i `secondLanguage` (oba `String`). Onemogućite stvaranje objekata tipa `LibraryItem`.

9. U klasu LibraryItem dodajte metodu getLoanPeriod, vrijeme posudbe je redom: za knjige 14 dana, osim za školske udžbenike za koje je 120 dana, za digitalne i e-sadržaje 30 dana, a za časopise 7 dana. Za klasu LibraryItem nije moguće odrediti vrijeme posudbe. Za školske udžbenike i digitalne sadržaje potrebno je pri posudbi ostaviti novčani polog, i za njih moramo osigurati da implementiraju metode `public boolean hasCashDeposit()` i `public double getCashDepozitAmmount()`. Metoda `getCashDepozitAmmount()` vraća različite iznose ovisno o svakom konkretnom sadržaju, dok `hasCashDeposit()` za sve sadržaje vraća `true`, osim za neke udžbenike, kojima je iznos depozita eksplicitno postavljen na 0. Kako ćemo ispuniti ove zahtjeve?

   Za isječak koda:

```
Book book = new Book(1351, "The Hitchhiker's Guide to the Galaxy", "Douglas Adams");
Dictionary dict = new Dictionary(1652, "Englezko-hrvatski rječnik obavjestničkoga nazivlja",
       "Zdenko Škiljan", "Hrvatski", "Engleski");
Magazine magazine = new Magazine(1871, "Hacker", "computer games", 321);
DigitalItem di = new DigitalItem(2162, "Microsoft Encarta", "DVD", 50);
TextBook tb = new TextBook(3176, "Demistificirani C++", "Šribar & Motik", "ASP", 0);
EItem ei = new EItem(3217, "Programiranje u Javi",
       new URL("http://java.zemris.fer.hr/nastava/opjj/book-2015-09-30.pdf"));

LibraryItem[] items = new LibraryItem[] {book, dict, magazine, di, tb, ei};

for (LibraryItem item : items) {
  System.out.println(item.getId() + " " + item.getName());
  if (item instanceof ItemWithCashDepozit && ((ItemWithCashDepozit)item).hasCashDeposit())
    System.out.println(" +- cash depozit: " + ((ItemWithCashDepozit)item).getCashDepozitAmmount());
}
```

   Očekuje se ovakav ispis:

```
1351 The Hitchhiker's Guide to the Galaxy
1652 Englezko-hrvatski rječnik obavjestničkoga nazivlja
1871 Hacker
2162 Microsoft Encarta
 +- cash depozit: 50.0
3176 Demistificirani C++
3217 Programiranje u Javi
```

10. Svi predmeti iz knjižnice osim `Eitem`-a imaju svoju fizičku lokaciju unutar knjižnice (polica i red na polici) te moraju imati metodu `String getLocation()` koja će vratiti tekstualni opis mjesta gdje se predmet nalazi. Koje su vam opcije za modeliranje ove funkcionalnosti na raspolaganju?

Rješenja zadataka dostupna su na sljedećim poveznicama:

1. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-05/src/main/java/hr/fer/oop/homework_05/e01
2. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-05/src/main/java/hr/fer/oop/homework_05/e02
3. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-05/src/main/java/hr/fer/oop/homework_05/e03
4. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-05/src/main/java/hr/fer/oop/homework_05/e04
5. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-05/src/main/java/hr/fer/oop/homework_05/e05
6. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-05/src/main/java/hr/fer/oop/homework_05/e06
7. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-05/src/main/java/hr/fer/oop/homework_05/e07
8. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-05/src/main/java/hr/fer/oop/homework_05/e08
9. https://github.com/FER-OOP/Lectures/tree/master/Exercises/Homework-05/src/main/java/hr/fer/oop/homework_05/e09
10. Imamo dvije mogućnosti:
    - možemo restrukturirati model tako da dodamo dvije nove (apstraktne) klase RealItem i VirtualItem, koje nasljeđuju LibraryItem, a ostale klase izvodimo iz njih.
    - možemo napraviti sučelje koje će implementirati sve klase osim EItem.