Zadatak Avion Roba

1 Kreiranje projekta i paketa

Kreirati novi Gradle projekat sa Javom koji će imati naziv ZadatakAvionRoba. U generisanom main.java paketu napraviti podpakete:

a) model

b) zadaci

NAPOMENA: Ako IntelliJ ne prepozna build.gradle, potrebno je restartovati ga.

1.1 Kreiranje Git repozitorijuma

Inicijalizovati git repozitorijum u folderu projekta i dodati .gitignore sa pravilima za IntelliJ. Mogu se naći na sledećem linku:
`https://github.com/github/gitignore/blob/master/Global/JetBrains.gitignore`
Na github-u napraviti udaljeni repozitorijum sa imenom ZadatakAvionRoba. Povezati udaljeni repozitorijum sa lokalnim, komandom:

git remote add origin <adresaRepozitorijuma>

Za svaki zadatak napraviti issue na githubu. Naslov issue treba da odgovara rednom broju zadatka (npr. issue za zadatak 2.2 će se zvati "Zadatak 2.2."). Potrebno je da svaki komit kroz tekst poruke zatvori odgovarajući issue (npr. komit za zadatak 2.2 zatvara issue za zadatak 2.2).
Napomena: Ukoliko ste zaboravili da zatvorite issue komitom, dodati komentar u issue "rešeno sa <šifraIssue>" i zatvoriti ga ručno.

Sve izmene iz prethodnog zadatka komitovati u git repozitorijum i za tekst komit poruke napisati "Kreiran projekat i paketi". Pushovati izmene. Napomena: Za prvi push se dodaje "-u origin master" iza komande za push.

1.2 Kreiranje Gradle skripte

U Gradle build skriptu dodati sledeće zavisnosti:

a) compile group: 'org.xerial', name: 'sqlite-jdbc', version: '3.8.11.2'

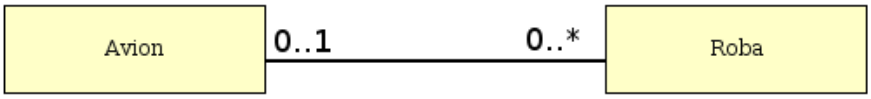
b) compile group: 'com.j256.ormlite', name: 'ormlite-jdbc', version: '5.0'

c) compile group: 'com.j256.ormlite', name: 'ormlite-core', version: '5.0'

Pokrenuti build i proveriti da li su biblioteke prisutne.
Sve izmene komitovati u git repozitorijum i za tekst komit poruke napisati "Uradjena Gradle skripta". Pushovati izmene na github.

2 ZADATAK AVION ROBA

Napisati klase za entitete koji su prikazani na slici **??**.



Slika 2.1: Entiteti Avion Roba

U projektu ZadatakAvionRoba u folderu zadaci napraviti četiri klase:

a) Zadatak1KreiranjeTabela

b) Zadatak2DodavanjeVrednosti

c) Zadatak3IzmenaVrednosti

d) Zadatak4BrisanjeVrednosti

Unutar svake klase napisati main metodu:

public static void main(String[] args)

Unutar main metoda napisati potrebne naredbe za konekciju sa bazom i njih obuhvatiti try catch blokom naredbi. Link za konekciju sa bazom treba da bude:
"jdbc:sqlite:avionRoba.db"
Sve dodate klase komitovati u git repozitorijum i za tekst komit poruke napisati "Kreirane klase za zadatke". Pushovati izmene na github.

2.1 KREIRANJE KLASA ZA ENTITETE

2.1.1 ZADATAK KLASA AVION

U paketu model napraviti novu klasu Avion. Za ovu klasu definisati sledeće atribute:

a) id koji je tipa int

b) oznaka koji je tipa String

c) rasponKrila koji je tipa int

Svi atributi treba da imaju modifikator pristupa postavljen na private. Za klasu Avion treba dodati sledeće statičke atribute i njihove vrednosti:

a) POLJE_OZNAKA vrednost je: "oznaka"

b) POLJE_RASPON_KRILA vrednost je: "raspon_krila"

Za klasu Avion dodati odgovarajuću anotaciju da u bazi odgovara tabeli "avion". Za atribute klase Avion, osim statičkih atributa, dodati odgovarajuće anotacije da bi nazivi kolona odgovarali vrednostima statičkih atributa i da bi vrednosti atributa klase Avion morale biti unete prilikom upisa u bazu. Klasi Avion bi trebalo da se automatski generiše primarni ključ.
Jedan avion može da prevozi više robe. U klasi Avion ubaciti odgovarajući atribut i anotaciju atributa kako bi se predstavio više kraj veza između Roba i Avion klasa.
Za klasu Avion napisati i konstruktor bez parametara i konstruktor koji očekuje parametre oznaka i rasponKrila. Za odgovarajuće atribute dodati i get i set metode. Za statičke atribute ne treba da se dodaju get i set metode.
Redefinisati toString metodu, tako da prikazuje vrednosti atributa id, oznaka i rasponKrila.
Implementiranu funkcionalnost komitovati u git repozitorijum i za tekst komit poruke napisati "Uradjena klasa Avion". Pushovati izmene na github.

2.1.2 ZADATAK KLASA ROBA

U paketu model napraviti novu klasu Roba. Za ovu klasu definisati sledeće atribute:

a) id koji je tipa int

b) naziv koji je tipa String

c) opis koji je tipa String

d) tezina koji je tipa double

Svi atributi treba da imaju modifikator pristupa postavljen na private. Za klasu Roba treba dodati sledeće statičke atribute i njihove vrednosti:

a) POLJE_NAZIV vrednost je: "naziv"

b) POLJE_OPIS vrednost je: "opis"

c) POLJE_TEZINA vrednost je: "tezina"

Za klasu Roba dodati odgovarajuću anotaciju da u bazi odgovara tabeli "roba". Za atribute klase Roba, osim statičkih atributa, dodati odgovarajuće anotacije da bi nazivi kolona odgovarali vrednostima statičkih atributa i da bi vrednosti atributa klase Roba morale biti unete prilikom upisa u bazu. Klasi Roba bi trebalo da se automatski generiše primarni ključ.
Jedna roba može da se prevozi samo sa jednim avionom. U klasi Roba ubaciti odgovarajući atribut i anotaciju atributa kako bi se predstavio jedinični kraj veze između Avion i Roba klasa.
Za klasu Roba napisati i konstruktor bez parametara i konstruktor koji očekuje parametre naziv, opis i tezina. Za odgovarajuće atribute dodati i get i set metode. Za statičke atribute ne treba da se dodaju get i set metode.
Redefinisati toString metodu, tako da prikazuje vrednosti atributa id, naziv, opis i tezina.
Implementiranu funkcionalnost komitovati u git repozitorijum i za tekst komit poruke napisati "Uradjena klasa Roba". Pushovati izmene na github.

2.2 ZADATAK KREIRANJE TABELA

U klasi Zadatak1KreiranjeTabela, unutar main metode, prvo obrisati tabele za klase Roba i Avion, a zatim ih kreirati u bazi. Paziti na redosled brisanja i kreiranja tabela, zbog ograničenja referencijalnog integriteta.
Implementiranu funkcionalnost komitovati u git repozitorijum i za tekst komit poruke napisati "Uradjen zadatak kreiranja tabela". Pushovati izmene na github.

2.3 ZADATAK DODAVANJE VREDNOSTI

U klasi Zadatak2DodavanjeVrednosti, unutar main metode, instancirati objekte klasa Avion i Roba. Objekte klase Avion instancirati sa sledećim vrednostima atributa:

a) oznaka: "Avion1", rasponKrila: 34

b) oznaka: "Avion2", rasponKrila: 21

Instancirane objekte upisati u bazu.
Objekte klase Roba instancirati sa sledećim vrednostima atributa:

a) naziv: "Patike", opis: "Duboke patike",tezina: 1 i za avion koji prevozi robu postaviti objekat Avion sa oznakom "Avion1"

b) naziv: "Kosulja", opis: "Na duge rukave",tezina: 0.4 i za avion koji prevozi robu postaviti objekat Avion sa oznakom "Avion1"

c) naziv: "Voda", opis: "Voda za pice",tezina: 1.4 i za avion koji prevozi robu postaviti objekat Avion sa oznakom "Avion1"

d) naziv: "Ploce", opis: "Drvene ploce",tezina: 3.4 i za avion koji prevozi robu postaviti objekat Avion sa oznakom "Avion2"

e) naziv: "Stolica", opis: "Plasticna stolica",tezina: 2.4 i za avion koji prevozi robu postaviti objekat Avion sa oznakom "Avion2"

Sve objekte klase Roba upisati u bazu.
Prikazati sve vrednosti iz tabela roba i avion da bi se moglo proveriti da su sve vrednosti dobro upisane.
Implementiranu funkcionalnost komitovati u git repozitorijum i za tekst komit poruke napisati "Uradjen zadatak dodavanje vrednosti". Pushovati izmene na github.

2.4 ZADATAK IZMENA VREDNOSTI

U klasi Zadatak3IzmenaVrednosti, unutar main metode, izmeniti vrednost atributa opis za objekat Roba.
Prikazati sve vrednosti iz tabele roba. Pronaći robu koja za opis ima postavljenu vrednost "Plasticna stolica" i promeniti vrednost opisa u "Drvena stolica". Izmene sačuvati u bazi.
Prikazati sve vrednosti iz tabele roba da bi se potvrdilo da je izmenjena vrednost opisa.
Implementiranu funkcionalnost komitovati u git repozitorijum i za tekst komit poruke napisati "Uradjen zadatak izmena vrednosti". Pushovati izmene na github.

2.5 ZADATAK BRISANJE VREDNOSTI

U klasi Zadatak4BrisanjeVrednosti, unutar main metode, obrisati jednu Robu.
Prikazati sve vrednosti iz tabele roba. Pronaći robu koja za naziv ima postavljenu vrednost "Voda" i obrisati tu robu iz baze.
Prikazati sve vrednosti iz tabele roba da bi se potvrdilo da je roba obrisana.
Implementiranu funkcionalnost komitovati u git repozitorijum i za tekst komit poruke napisati "Uradjen zadatak brisanja vrednosti". Pushovati izmene na github.

2.6 ZADATAK NITI

Svakog jutra avioni poleću sa aerodroma da dostave natovarenu robu. Avioni su implementirani kao niti AvionNit i pre poletanja je potrebno da prođu kroz provere opreme koje traju od 0 do 2 sekunde. Kada je utvrđeno da je avion spreman, traži dozvolu za poletanje. Ako je pista slobodna, kreće da poleće i zauzima pistu. Poletanje traje 2 sekunde. Nakon toga oslobađa pistu i sledeći avion može da poleti. Kada su svi avioni poleteli, na ekran se ispisuje "Svi avioni su poleteli".
Za dozvolu poletanja koristiti statičku promenljivu dozvoljenoSletanje u klasi AvionNit tipa Boolean. Pristup ovoj promenljivoj mora biti sinhronizovan.
AvionNit ima polje avion kao referencu na klasu Avion iz zadatka 2.1.1.
Nit treba da ispisuje sledeće na ekran:

- "Počinju provere za avion <id_aviona>" na početku izvršavanja
- "Avion <id_aviona> je spreman za poletanje i čeka dozvolu za poletanje" nakon što se završi provera aviona
- "Avion <id_aviona> izlazi na pistu i poleće" na početku poletanja
- "Avion <id_aviona> je poleteo" na kraju poletanja

Iz baze učitati sve avione iz zadatka 2.3 i kreirati po nit za svaki. Pokrenuti ove niti u main metodi. Main metoda ide u klasu AvionNit. Ispis na ekran nakon što su svi poleteli se vrši u main metodi.
Implementiranu funkcionalnost komitovati u git repozitorijum i za tekst komit poruke napisati "Implementiran zadatak iz niti". Pushovati izmene na github.