Elektrotehnički fakultet
Banja Luka
Programski jezici 2

Projektni zadatak

maj 2025.

Potrebno je implementirati GUI aplikaciju koja korisnicima omogućava pronalaženje optimalne rute za putovanje između gradova unutar jedne države, koristeći kombinaciju autobuskog i željezničkog prevoza. Svaki grad ima autobusku i željezničku stanicu sa različitim vremenima polazaka. Potrebno je osmisliti efikasan način pretrage i kombinovanja različitih vrsta prevoza kako bi se pronašla najbolja ruta.

Država se modeluje kao matrica gradova (n × m), gdje svaki element predstavlja jedan grad. Svaki grad ima autobusku i željezničku stanicu (sa različitim vremenima polazaka). Polasci su definisani za svaku stanicu i imaju: vrijeme polaska, vrijeme dolaska, cijenu karte i minimalno vrijeme čekanja. Putnik može kombinovati oba vida prevoza (autobus i voz) kako bi stigao do cilja.

Aplikacija učitava podatke iz JSON fajla generisanog pomoću *TransportDataGenerator* klase dostupne na *Moodle* stranici predmeta. Trenutna implementacija klase generatora radi sa kvadratnom matricom gradova. Generator klasu je potrebno modifikovati tako da generiše matricu gradova n × m tako da se dimenzije prosljeđuju kroz konstruktor. Generisani fajl sadrži:

- Mapu države (matricu gradova n × m, čije dimenzije se unose pri pokretanju aplikacije),
- Listu stanica (autobuskih i željezničkih) za svaki grad,
- Raspored polazaka za svaku stanicu (uključujući vrijeme polaska, vrijeme dolaska, cijenu i minimalno vrijeme čekanja).

Napomena: Čvorovi su stanice (A_X_Y za autobuse, Z_X_Y za vozove), a grane su veze između njih.

Aplikacija treba da omogući:

1. Odabir početnog i odredišnog grada (putem interaktivne mape ili padajućih menija (*combo box*)).
2. Odabir kriterijuma optimizacije:
    - Najkraće vrijeme putovanja,
    - Najniža cijena,
    - Najmanji broj presjedanja.
3. Prikaz optimalne rute (sekvenca prevoza sa detaljima).
4. Vizualizaciju grafa sa označenim optimalnim putem.

Aplikacija treba da sadrži sljedeće komponente grafičkog korisničkog interfejsa (GUI) implementirane pomoću JavaFX ili Swing biblioteke:

- Interaktivna mapa (prikaz gradova i stanica),
- Forma za unos parametara (polazište, odredište, kriterijum),
- Tabelarni prikaz najbolje rute (vremena, cijene, presjedanja),
- Prikaz top 5 ruta za zadati kriterijum,

- Vizualizacija grafa (koristeći JavaFX Canvas ili biblioteku za grafove, npr. GraphStream - https://graphstream-project.org/).

Pri implementaciji koda koristiti koncepte objetkno-orijentisanog programiranja i omogućiti njegovu fleksibilnu upotrebu i proširivost.

Primjer tabelarnog prikaza rute dat je na slici 1.

Polazak	Dolazak	Tip	Cijena
A_2_3 (08:00)	G_3_3 (08:45)	Autobus	200
Z_3_3 (09:05)	G_5_7 (10:30)	Voz	350
Ukupno: 2h 30min, 550 novčanih jedinica.			

Slika 1. Tabelarni prikaz rute od grada A_2_3 do grada Z_3_3

Ispod tabelarnog prikaza optimalne rute nalaze se dva dugmeta:

- **"Prikaz dodatnih ruta"** – otvara novi prozor sa listom **top 5 ruta** prema zadatom kriterijumu optimizacije. Svaka ruta prikazuje osnovne informacije (vrijeme, cijena, presjedanja) i ima dugme **"Kupi kartu"**.

- **"Kupovina karte"** – omogućava korisniku da kupi prikazanu rutu direktno.

Klikom na dugme za kupovinu (u bilo kojem prozoru):

- Generiše se **račun** u obliku tekstualnog fajla koji sadrži detalje kupovine (relacija, vrijeme, cijena, datum).

- Račun se automatski čuva u lokalni folder **računi**.

Pri svakom pokretanju aplikacije na početnom prozoru se prikazuje: ukupan broj prodatih karata i ukupan prihod od prodaje. Ove vrijednosti se učitavaju na osnovu prethodno generisanih računa.

Napomene:
- ovaj projektni zadatak vrijedi do objave novog projektnog zadatka,
- obavezno dodati JavaDoc komentare i generisati dokumentaciju,
- napisati izvještaj u kom su objašnjeni korišteni algoritmi za pronalaženje optimalne rute po različitim kriterijumima, kao i razlog njihovog odabira,
- obavezno napraviti pakete,
- izbjegavati dupliranje koda i
- voditi računa o kvaliteti koda (nazivi klasa, metoda, promjenljivih, poravnanja, performanse…)