CS103 - DOMAĆI ZADATAK 10.

Prilikom slanja domaćeg zadatka svom asistentu neophodno je da ispunite sledeće:

- Subject mail-a mora biti CS103-DZbr. Za ovaj domaći zadatak - CS103-DZ10
- Sve fajlove, koji su deo rešenja zadataka, arhivirati (zip, rar, …)
- Poželjno je uraditi i printscreen koda pre pokretanja programa
- U prilogu maila treba da se nalazi arhiva projekta koji se ocenjuje, imenovana na sledeći način: CS103-DZbr-BrojIndeksa-ImePrezime. *Na primer, CS103-DZ10-9999-VeljkoGrkovic.zip*
- Telo mail-a treba da ima pozdravnu poruku

Molimo sve studente da se pridržavaju navedenog, inače zadaci neće biti pregledani i ocenjeni.

Studenti iz Beograda i online studenti zadatke na ocenjivanje šalju mail-om na adresu lazar.mrkela@metropolitan.ac.rs
Studenti iz Niša zadatke na ocenjivanje šalju mail-om na adresu veljko.grkovic@metropolitan.ac.rs

Rok za slanje domaćih zadataka:

- **Tradicionalni studenti: sedam dana od dana održavanja vežbe na koju se odnosi domaći zadatak. Nakon toga, broj poena se umanjuje za 50%. Krajnji rok za slanje rešenja je deset dana pre termina ispita u kome želite da polažete ispit.**
- **Studenti na online nastavi, kao i studenti kojima je predmet prenet u narednu godinu ili je diferencijalni: najkasnije deset dana pre termina ispita u kome želite da polažete ispit.**

Svaki student radi jedan zadatka sa spiska. Ostali zadaci mogu da posluže za vežbanje i pripremu ispita, ali ih ne šaljete na pregled asistentima.

Redni broj zadatak dobijate tako što vaš broj indeksa podelite sa 16, a dobijeni ostatak pri deljenju uvećate za 1. Na primer:
Broj indeksa 2378
2378 % 16 + 1 = 11 – student radi 11. zadatak.

1. Kreirati metod koji za svaki čvor od grafa računa njegovu udaljenost od ostalih čvorova. Smatrati da je čvor od samog sebe udaljen 0 čvorova, od prvog suseda 1 i tako redom. Udaljenosti beležiti u matrici udaljenosti. Napisati program koji testira metod.

2. Kreirati funkciju koji za proizvoljno ***k*** nalazi sve puteve dužine ***k*** u usmerenom grafu koji je dat matricom povezanosti. Napisati program koji testira funkciju.

3. Napisati program koji predstavlja graf preko matrice povezanosti i dozvoljava korisniku da odabere dva čvora. Program zatim treba da proveri da li postoji direktan ili indirektan put od prvog do drugog unetog čvora i o tome da obavesti korisnika.

4. Napisati funkciju koja proverava da li je graf cikličan. Napisati program koji testira ovu funkciju.

5. Na osnovu matrice povezanosti proveriti da li je graf neusmeren. Graf je neusmeren ako ima i ***AB*** i ***BA*** granu. Ovo treba da važi za svaku granu.

6. Napisati funkciju za obilazak grafa u dubinu koja kada štampa svaki od čvorova grafa pored njega ispisuje i redni broj čvora koji je obrađen. Koren računati kao 0. čvor.

7. Napisati funkciju za obilazak grafa u širinu koja kada štampa svaki od čvorova grafa pored njega ispisuje i redni broj čvora koji je obrađen. Koren računati kao 0. čvor.

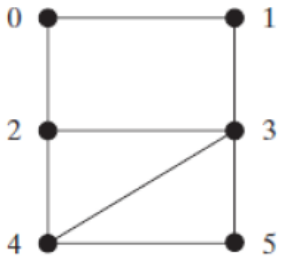
8. Za neusmereni graf koji je dat listom povezanosti u datoteci ***ulaz.txt*** proveriti da li je cikličan.

9. Za neusmereni graf koji je dat listom povezanosti u datoteci ***ulaz.txt*** proveriti koji je najkraći put od čvora **a** do čvora **b**. Čvorove **a** i **b** unosi korisnik.

10. Dat je neusmeren graf **G**. Korisnik unosi oznaku čvora. Napisati program koji ispisuje sve dostižne čvorove iz čvora **A**.

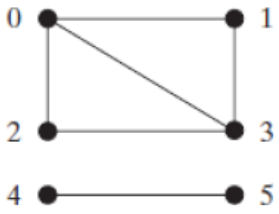
11. Napisati program koji učitava graf iz ***ulaz.txt*** fajla i ispituje da li je graf povezan ili ne. Graf je povezan ako postoji veza između bilo koje dve tačke grafa (slika a). U suprotnom graf nije povezan (slika b). Prva linija u fajlu predstavlja broj čvorova grafa G. U narednih N linija se nalaze podaci u obliku **u v1 v2 ...,** koje predstavljaju grane grafa **(u, v1), (u, v2)**, itd.

```
File
6
0 1 2
1 0 3
2 0 3 4
3 1 2 4 5
4 2 3 5
5 3 4
```



(a)

```
File
6
0 1 2 3
1 0 3
2 0 3
3 0 1 2
4 5
5 4
```



(b)

12. Napraviti klasu za usmereni graf korišćenjem liste povezanosti. Neka je kao ulaz data datoteka koja sadrži broj čvorova, kao i parove **(a, b)** koje označavaju usmerenu granu od ***a*** ka ***b***. Napisati funkciju koja ispituje da li je graf potpun ili ne. U slučaju da je graf nepotpun odštampati kao izlaz sve grane koje nedostaju da bi graf bio potpun (graf je potpun ako postoji direktna veza između svaka dva čvora).

13. Napisati program koji za uneti usmereni graf **G**, i uneti čvor **V** ispituje da li postoji ciklična putanja od čvora **V** kroz ostale čvorove grafa nazad do čvora **V**.

14. Napisati funkciju koja pronalazi cikličnu putanju u grafu i smešta tu putanju u listu. Funkcija može da ima sledeće zaglavlje:
    `public List getACycle(int u);`
    Funkcija kao rezultat vraća listu koja sadrži sve čvorove u ciklusu počevši od čvora **u**. Ukoliko graf ne sadrži nijedan ciklus funkcija treba da vrati **null**.

15. Napisati funkciju koja vrši pretragu grafa u dubinu (DFS) korišćenjem steka umesto rekurzije.

16. Napisati funkciju koja pronalazi putanju između tačaka **u** i v**:**
    `public List getPath(int u, int v);`
    Funkcija kao rezultat vraća listu koja sadrži sve čvorove putanje od **u** do **v**. Pretragu vršiti korišćenjem BFS (pretraga u širinu). Ukoliko ne postoji putanja između **u** i **v,** funkcija vraća **null**.