Dežurni nastavnik: _____

Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku
Studijski program „Softversko inženjerstvo“

Predmet: Testiranje softvera (13S113TS)
Nastavnici: prof. dr Dragan Bojić, prof. dr Dražen Drašković
Saradnik: mast. inž. Nikola Stanković
Ispitni rok: Drugi kolokvijum (januar 2026.)
Datum: 13.01.2026.

Kandidat:* _____

Broj indeksa:* _____

„Danas polažete dva ispita: iz struke i iz poštenja. Ako treba jedan da padnete, neka to bude iz struke, jer ćete kao pošteni ljudi, lako savladati struku, a ako padnete na poštenju, nećete ga savladati nikad.“

Kolokvijum traje 120 minuta. U toku prvih 60 minuta nije dozvoljeno napuštanje kolokvijuma. Upotreba literature nije dozvoljena.

Zadatak 1 _____ / 5

Zadatak 2 _____ / 8

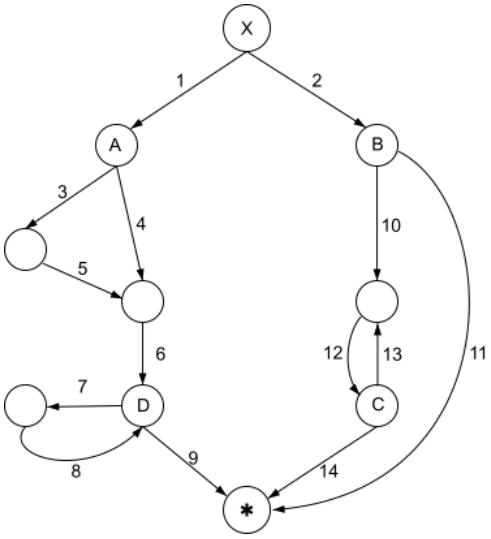
Zadatak 3 _____ / 7

Ukupno na kolokvijumu: _____ / 20

Napomena: Ukoliko u zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku, da je uokviri (da bi se lakše prepoznala prilikom ocenjivanja) i da nastavi da izgrađuje preostali deo svog odgovora na temeljima uvedene pretpostavke. Kod pitanja koja imaju ponuđene odgovore treba **samo zaokružiti** jedan odgovor. Na ostala pitanja odgovarati **čitko i precizno**.

* popunjava student.

1. [5] Neka je dat graf kao na slici, koji predstavlja jedan program. Početni čvor je označen sa X, a završni sa *. U korenom čvoru, X predstavlja složeni uslov (*K<10 AND P>250*). U ostalim čvorovima odluka, uslovi su elementarne logičke promenljive (*A*, *B*, *C* i *D*).



a) **[3]** Napisati minimalan skup test primera za pokrivanje sledećih vrsta testiranja belom kutijom datih u tabeli. Popuniti datu tabelu i numerisati redom testove. Za svaki test navesti vrednosti svih relevantnih promenljivih i redosled grana prilikom izvršavanja testa.

b) **[1]** Napisati formulu (i objasniti sve simbole u formuli) po kojoj se računa i izračunati *McCabe*-ov broj ciklomatske kompleksnosti datog grafa:

V = _____ = _____

c) **[1]** Odrediti sve nezavisne putanje u ovom programu i napisati da li putanja 1-3-5-6-9 može da se pokrije uz pomoć navedenih nezavisnih putanja i ukoliko je moguće, preko kojih putanja?

Rešenje:

Vrsta testiranja	Test primeri
Pokrivanje uslova (*Condition Coverage*)	
Pokrivanje odluka (*Decision Coverage*)	
Pokrivanje višestrukih uslova (*Multiple Condition Coverage*)	

2. [8] Data je funkcija *check_special_number* napisana u programskom jeziku Java koja određuje i ispisuje da li je prosleđeni ceo broj specijalan ili ne. Broj je specijalan ukoliko je proizvod njegovih cifara na neparnim pozicijama jednak proizvodu njegovih cifara na parnim pozicijama, pri čemu se cifre redom posmatraju od pozicije najmanje težine ka najvećoj težini.

```
0.   public static void check_special_number(int num) {
1.      int x = num, e = 1, o = 1;
2.      boolean f = true;
3.      while (x > 0) {
4.         int d = x % 10;
5.         if (f) {
6.            o *= d;
7.         }
8.         else e *= d;
9.         f = !f;
10.        x /= 10;
11.     }
12.     if (e == o)
13.        System.out.println(String.format("%d jeste specijalan!", num));
14.     else
15.        System.out.println(String.format("%d nije specijalan.", num));
16.  }
```

a) [2] Odrediti sve definicije, *p*-upotrebe, *c*-upotrebe i popuniti sledeću tabelu:

Promenljiva	definicije	*p*-upotrebe	*c*-upotrebe
num			
x			
e			
o			
f			
d			

b) [2] Odrediti sve DU lance za promenljive *num*, *x*, *e*, *o*, *f* i *d* date funkcije. Numerisati svaki DU lanac.

c) [2] Napisati minimalni skup test primera kojima se pokrivaju svi dostižni DU lanci iz tačke b). Uz svaki test primer navesti koje DU lance on pokriva i to samo one DU lance koje do tog trenutka nije pokrio nijedan prethodni test primer. Broj redova tabele ne mora da odgovara minimalnom broju test primera.

TP	*num*	Pokriveni DU lanci	Izlaz (konzola)

d) [2] Odrediti sve LCSAJ sekvence za datu funkciju.

3. [7] Za sledeću metodu implementiranu u programskom jeziku Java, koja proverava validnost ocene unete pri zaključivanju ispita, u informacioni sistem ETF-a, primeniti mutacione operatore prvog reda tipa *ROR* (*Relational Operator Replacement*), koji menja relacione operatore sa svim drugim relacionim operatorima, i tipa *ABS* (*Absolute Value Insertion*), koji menja vrednosti promenljive i čitavog izraza u apsolutnu vrednost, a takođe i u negativnu vrednost.

```
1.   public static String validateGrade(int grade) {
2.      if (grade >= 5) {
3.         if (grade <= 10) {
4.            if (grade >= 6) {
5.               return "PASS";
6.            } else {
7.               return "FAIL";
8.            }
9.         } else {
10.           return "INVALID";
11.        }
12.     } else {
13.        return "INVALID";
14.     }
15.  }
```

Izlazi glavnog programa za test primere:
TP1: grade = 8 Izlazna vrednost: _____
TP2: grade = 11 Izlazna vrednost: _____
TP3: grade = 10 Izlazna vrednost: _____
TP4: grade = 3 Izlazna vrednost: _____

a) [3] Popuniti sledeće tabele izmenama u mutiranim programima i odrediti mutacioni skor:

Primena mutacionih operatora *ROR* i *ABS*			Napisati izlaze (return vrednosti) kao rezultate izvršavanja testova nad mutiranim programima				
R.br. mutanta	Izmena u liniji br.	Novi izmenjeni izraz	TP1: grade=8	TP2: grade=11	TP3: grade=10	TP4: grade=3	**L**IVE ili **K**ILL
M1							
M2							
M3							
M4							

Mutacioni skor = _____

b) [2] Ukoliko mutacioni skor nije maksimizovan, koji još test primeri su minimalno neophodni, da bi mutacioni skor bio bolji? Napisati nove test primere, kao i koje mutant programe ubijaju takvi test primeri.

c) [2] Ukoliko bismo napravili izmene u glavnom programu i dobili ovakva dva nova program (dole prikazani), da li su ti mutirani programi ekvivalentni gorenavedenom programu? Ukoliko je odgovor odričan, napisati zašto nije ekvivalentan (tj. za koje ulazne vrednosti ne daju iste rezultate kao glavni program), a ukoliko neki od njih jeste ekvivalentan, napisati na osnovu toga još jedan ekvivalentan mutant, ali koristeći samo De Morganove zakone.

Novi mutirani program #1	Odgovor:

1. public static String validateGrade(int grade) {
2. if (grade >= 5) {
3. if (grade <= 10) {
4. return "PASS";
5. } else {
6. return "INVALID";
7. }
8. } else {
9. return "INVALID";
10. }
11. }	

Novi mutirani program #2:	**Odgovor:**

1. public static String validateGrade(int grade) {
2. if (grade >= 5 && grade <= 10) {
3. if (grade >= 6) {
4. return "PASS";
5. } else {
6. return "FAIL";
7. }
8. } else {
9. return "INVALID";
10. }
11. }	