



Testiranje softvera

ETF BEOGRAD, 2019/2020.

VEŽBE #7, ASISTENT: DR DRAŽEN DRAŠKOVIĆ



Testiranje strategijama bele kutije

WHITE BOX TESTING

Testiranje metodom toka podataka

- ▶ Lanac "dodele-upotrebe" (DU lanac)
- ▶ $DEF(S1) = \{ X \}$ kao skup svih promenljivih X za koje postoji dodela vrednosti u iskazu S1.
- ▶ $USE(S2) = \{ X \}$ kao skup svih promenljivih X za koje postoji upotreba vrednosti u iskazu S2.
- ▶ DU lanac: [X, S1, S2]

Zadatak 1 - Trgovinska radnja i čokolade

- ▶ Za sledeći programski fragment, strategijom pokrivanja svih DU lanaca, odrediti sve test primere. Analizirati samo promenljive *cokolade*, *zalihe* i *popust*. Izostaviti iz razmatranja promenljive *broj_cokolada*, *broj_zaliha* i *provizija*.

```
broj_cokolada = 0;
broj_zaliha = 0;
S1:  read( cokolade );
S2:  while cokolade != -1 do
      begin
S3:    read( zalihe );
S4:    broj_cokolada := broj_cokolada + cokolade;
S5:    broj_zaliha := broj_zaliha + zalihe;
S6:    read( cokolade );
      end;
S7:  popust := 45 * broj_cokolada + 30 * broj_zaliha;
S8:  if( popust > 1000.0 ) then
      begin
S9:    provizija := 0.1 * 1000;
S10:   provizija := provizija + 0.2 * (popust - 1000);
      end
      else
      begin
S11:   provizija := 0.1 * popust;
      end;
```

Zadatak 1 - Trgovinska radnja i čokolade - Rešenje (1)

- ▶ Za promenljivu *cokolade*: DEF: S1, S6 USE: S2, S4
- ▶ Postoje četiri DU lanca za ovu promenljivu:
 - DU1 = [*cokolade*, S1, S2]
 - DU2 = [*cokolade*, S1, S4]
 - DU3 = [*cokolade*, S6, S2]
 - DU4 = [*cokolade*, S6, S4]
- ▶ Za promenljivu *zalihe*: DEF: S3 USE: S5
- ▶ Postoji jedan DU lanac za ovu promenljivu:
 - DU5 = [*zalihe*, S3, S5]

Zadatak 1 - Trgovinska radnja i čokolade - Rešenje (2)

- ▶ Za promenljivu *popust*: DEF: S7 USE: S8, S10, S11
- ▶ Postoje dva DU lanca za ovu promenljivu:
 - DU6 = [*popust*, S7, S8]
 - DU7 = [*popust*, S7, S10]
 - DU8 = [*popust*, S7, S11]
- ▶ Dva test primera su dovoljna da pokriju sve navedene DU lance:
- ▶ TP1: *cokolade* = 20, *zalihe* = 20 (i tada je *popust* = 1500, pa je uslov u S8 *True*)
Sa TP1 pokriveni su DU lanci: du1, du2, du3, du4, du5, du6, du7.
- ▶ TP2: *cokolade* = 10, *zalihe* = 10 (i tada je *popust* = 750, pa je uslov u S8 *False*)
Sa TP2 pokriveni su DU lanci: du1, du2, du3, du4, du5, du6, du8.

Zadatak 2 - Ocene na predmetu (c- i p- upotrebe)

- ▶ Naznačiti koliko ukupno definicija, c-upotreba i p-upotreba po svakoj od navedenih celobrojnih promenljivih - *domaci*, *poeni*, *ukupno*, *brojac_studenata*, ima u sledećem programu, realizovanom u programskom jeziku Java, koji računa ocene na predmetu.
- ▶ Napomena: pretpostaviti da su sve metode *citajDomaci*, *citajPoeneIspit* i *citajPoeneProjekat* definisane.

Zadatak 2 - Ocene na predmetu (nastavak)

8

```
package ispit;
public class Ocena {
public static void main(String args[]) {
(1) int ukupno_studenata = args.length;
(2) int brojac_studenata = 0;
(3) while (brojac_studenata < ukupno_studenata) {
(4) System.out.println(args[brojac_studenata]);
(5) int broj_domacih = 4;
(6) int domaci = 0, ukupno = 0;
(7) while (domaci < broj_domacih) {
(8) int poeni =
        citajDomaci(args[brojac_studenata], domaci);
(9) if (poeni < 0)
(10) System.out.println("Nekorektan unos");
(11) else {
(12) ukupno += poeni;
(13) domaci++;
(14) }
(15) }
(16) int poeni = citajPoeneIspit(args[brojac_studenata]);
(17) if (poeni < 0)
(18) System.out.println("Nekorektan unos");
(19) else if (poeni >= 31 && poeni <= 50)
(20) ukupno += poeni;
(21) poeni = citajPoeneProjekat(args[brojac_studenata]);
(22) if (poeni >= 10 && poeni <= 30)
(23) ukupno += poeni;
(24) System.out.println("Ukupno poena " + ukupno);
(25) if (ukupno >= 91)
(26) System.out.println("Ocena 10");
(27) else if(ukupno >= 81)
(28) System.out.println("Ocena 9");
(29) else if(ukupno >= 71)
(30) System.out.println("Ocena 8");
(31) brojac_studenata++;
(32) }
(33) } //main
(34)} //class
```


Zadatak 2 - Ocene na predmetu - Rešenje (1)

- ▶ Da se podsetimo sa predavanja:
 - ▶ Upotreba se naziva predikatskom (eng. *predicate use*, skraćeno *p-use*) ako se pojavljuje u predikatskom izrazu naredbi kontrole toka (IF, WHILE, SWITCH itd.).
 - ▶ Upotreba se naziva računskom (eng. *computational use*, skraćeno *c-use*), ako nije predikatska.

Zadatak 2 - Ocene na predmetu - Rešenje (2)

Naziv promenljive	definicija	c-upotreba	p-upotreba
<i>domaci</i>	(6), (13)	(8), (13)	(7)
<i>poeni</i>	(8), (16), (21)	(12), (20), (23)	(9), (17), (19), (22)
<i>ukupno</i>	(6), (12), (20), (23)	(12), (20), (23), (24)	(25), (27), (29)
<i>brojac_studenata</i>	(2), (31)	(4), (8), (16), (21), (31)	(3)

Zadatak 2 - Ocene na predmetu - Rešenje (3)

- ▶ Kada je promenljiva `ukupno_studenata = 0`, može se realizovati samo: `[brojac_studenata, 2, 3]`
- ▶ Kada je promenljiva `ukupno_studenata ≥ 1`, analiziramo svaku promenljivu i njene DU lance:
 - `[domaci, 6, 7]` `[poeni, 8, 9]`
 - `[domaci, 6, 8]` `[poeni, 8, 12]`
 - `[domaci, 6, 13]` `[poeni, 16, 17]`
 - `[domaci, 13, 7]` `[poeni, 16, 19]`
 - `[domaci, 13, 8]` `[poeni, 16, 20]` uz uslov da su ispitni poeni ≥ 31 i ≤ 50
 - `[domaci, 13, 13]` `[poeni, 21, 22]`
`[poeni, 21, 23]` uz uslov da su projektni poeni ≥ 10 i ≤ 30

Zadatak 2 - Ocene na predmetu - Rešenje (4)

► DU lanci za promenljivu *ukupno*:

[ukupno, 6, 12] uz uslov da broj poena na domaćem N bude ≥ 0

[ukupno, 12, 12] uz uslov da broj poena na domaćem M > N, bude ≥ 0

[ukupno, 12, 20]

[ukupno, 6, 20] samo ako nijedan domaći nema korektan unos

[ukupno, 12, 23] uz uslov da su korektni poeni za bar 1 domaći zadatak,
ispitni poeni ≥ 0 i moraju da budu < 30 ili > 50

[ukupno, 20, 23] samo ako su ispitni poeni u korektnim granicama

[ukupno, 6, 23] samo ako su nekorektni poeni i sa domaćih i sa ispita

[ukupno, 6, 24] - ne smemo da redefinišemo u linijama 12, 20 i 23,
sve do upotrebe u 24. liniji

[ukupno, 12, 24] [ukupno, 20, 25]

[ukupno, 20, 24] [ukupno, 20, 27]

[ukupno, 23, 24] [ukupno, 20, 29]

[ukupno, 6, 25] [ukupno, 23, 25]

[ukupno, 6, 27] [ukupno, 23, 27]

[ukupno, 6, 29] [ukupno, 23, 29]

[ukupno, 12, 25]

[ukupno, 12, 27]

[ukupno, 12, 29]

Zadatak 2 - Ocene na predmetu - Rešenje (5)

- ▶ DU lanci za promenljivu *brojac_studenata*:

[brojac_studenata, 2, 3]	[brojac_studenata, 31, 3]
[brojac_studenata, 2, 4]	[brojac_studenata, 31, 4]
[brojac_studenata, 2, 8]	[brojac_studenata, 31, 8]
[brojac_studenata, 2, 16]	[brojac_studenata, 31, 16]
[brojac_studenata, 2, 21]	[brojac_studenata, 31, 21]
[brojac_studenata, 2, 31]	[brojac_studenata, 31, 31]

Zadatak 3 - Binarno pretraživanje

- ▶ Data je funkcija *bintr()* u programskom jeziku C, za binarno pretraživanje po ključu K sadržaja rastuće sortiranog vektora M od N elemenata. Rezultat izvršavanja ove funkcije je indeks elementa koji sadrži ključ K, ili vrednost -1, ako pretraživanje nije uspelo.
- ▶ a) Odrediti sve DU lance po promenljivama ID i IG. Formirati test primere koji pokrivaju pronađene DU lance.
- ▶ b) Nacrtati graf toka kontrole i odrediti broj ciklomatske kompleksnosti za navedeni programski kod.

Navesti bazični skup putanja i svaku putanju pokriti sa po jednim test primerom.

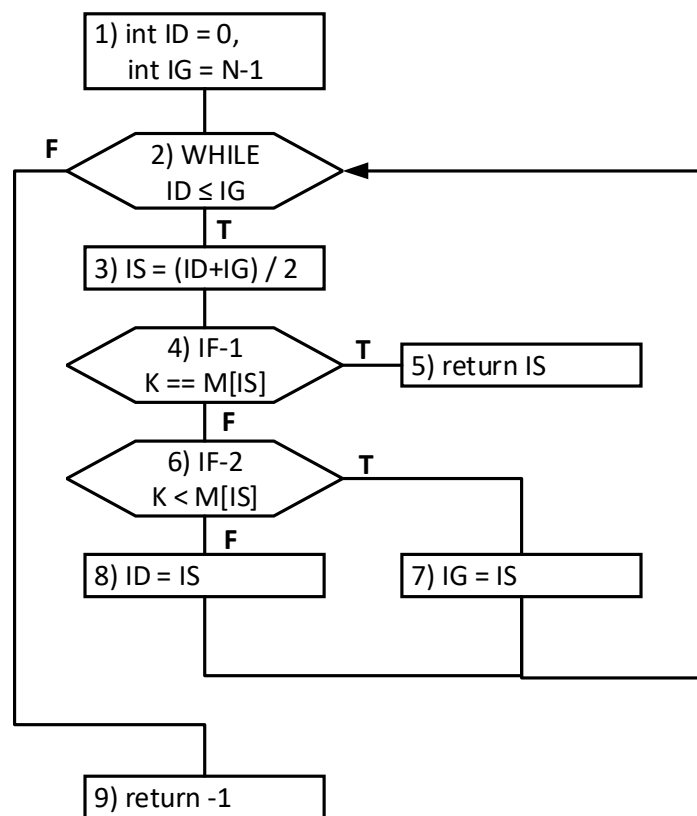
```
0 int bintr(int K, int M[], int N){
1     int ID = 0;
2     int IG = N - 1;
3     while (ID <= IG) {
4         IS = (ID + IG) / 2;
5         if (K == M[IS]) return IS;
6         else if (K < M[IS]) IG = IS;
7         else ID = IS;
8     }
9     return -1;
10 }
```

Zadatak 3 - Binarno pretraživanje - Rešenje (1)

- ▶ Promenljiva ID DEF: 1, 7 USE: 3, 4
- ▶ Promenljiva IG DEF: 2, 6 USE: 3, 4
- ▶ Od datih promenljivih, formiraćemo sledeće DU lance:
 [ID, 1, 3], [ID, 1, 4], [ID, 7, 3], [ID, 7, 4],
 [IG, 2, 3], [IG, 2, 4], [IG, 6, 3], [IG, 6, 4]
- ▶ Test primer koji bi pokrивao sve DU lance može biti niz M sa 10 elemenata:
 M = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
 N = 10
 K = 4

Zadatak 3 - Binarno pretraživanje - Rešenje (2)

- Graf kontrole toka:



Problem:
imamo 2 izlazna čvora!

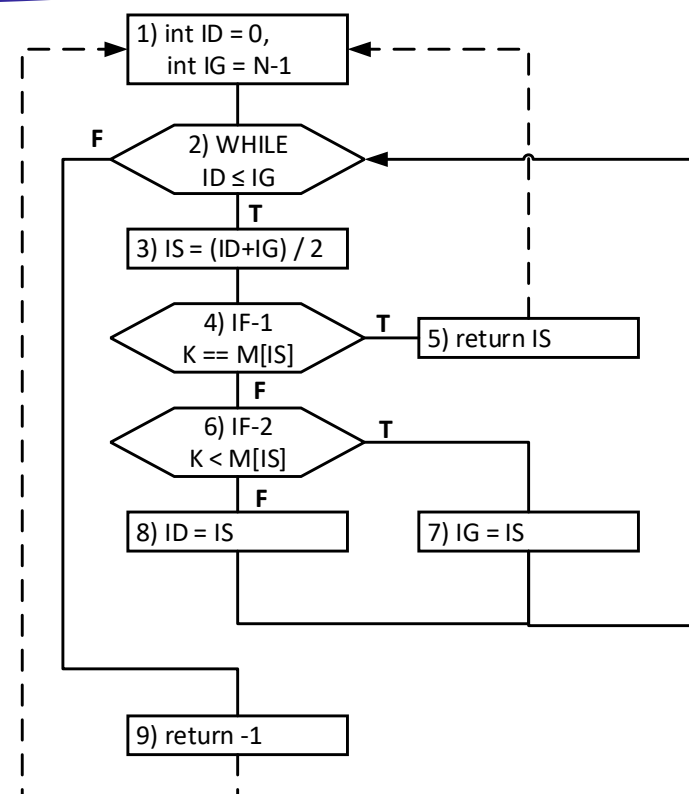
Zadatak 3 - Binarno pretraživanje - Rešenje (3)

- ▶ Graf kontrole toka - spojimo izlazne čvorove (5 i 9) sa ulaznim čvorom (1).
- ▶ $e = 12$ grana, $n = 9$ čvorova
- ▶ Po formuli: $V = e - n + 1 = 12 - 9 + 1 = 4$ bazične putanje
- ▶ { **[1, 2, 9]**, u iskazu 2 uslov (*While* = F);

[1, 2, 3, 4, 5], u iskazu 2 uslov (*While* = T), u iskazu 4 (*IF-1* = T);

[1, 2, 3, 4, 6, 8, 2, 9], u iskazu 2 uslov (*While* = T), u iskazu 4 (*IF-1* = F), u iskazu 6 (*IF-2* = F) i opet u iskazu 2 uslov je sada (*While* = F);

[1, 2, 3, 4, 6, 7, 2, 9], u iskazu 2 uslov (*While* = T), u iskazu 4 (*IF-1* = F), u iskazu 6 (*IF-2* = T) i opet u iskazu 2 uslov je sada (*While* = F); }



Zadatak 4 - Sportka radnja i popusti (za vežbu)

► U datom programu jedne sportske radnje u Beogradu, koji želimo da testiramo, potrebno je:

- a) nacrtati graf kontrole toka i izračunati broj ciklomatske kompleksnosti;
- b) odrediti sve definicije, c-upotrebe i p- upotrebe;
- c) odrediti sve DU lance i sa koliko minimalno test primera je neophodno da ih testiramo.

```
1. Program Sport ()
2.     var rabat, ukupno, finalnaCena, popust, cena
3.     rabat = 0.1
4.     ukupno = 0
5.     input (cena)
6.     while (cena != -1) do
7.         ukupno = ukupno + cena
8.         input (cena)
9.     end_while
10.    print ("Ukupna cena: " + ukupno)
11.    if (ukupno > 15.00) then
12.        popust = (rabat * ukupno) + 0.50
13.    else
14.        popust = rabat * ukupno
15.    end_if
16.    print ("Iznos popusta: " + popust)
17.    finalnaCena = ukupno - popust
18.    print ("Finalni iznos: " + finalnaCena)
19. end_program
```