GRUPA 3
BERINA SULJIĆ 18385 ELMA ŠEREMET 18318 AMINA ŠILJAK 18496

INSPEKCIJA KODA

Za inspekciju koda bit će korištena Walkthroughs metoda (prolaz-kroz)

1. Klasa Korisnik, metoda UlogujSe()

```
77      //Berina Suljic
78      /// <summary>
79      /// Metoda u kojoj se korisnik pokušava ulogovati na sistem.
80      /// Ukoliko se ulogovao više od 99 puta, ne dozvoljava mu se logovanje na sistem.
81      /// U suprotnom, povećava se broj logovanja i dozvoljava se login.
82      /// </summary>
83      /// <returns></returns>
        5 references
84      public bool UlogujSe()
85      {
86          if (brojLogovanja > 99) return false;
87          brojLogovanja++;
88          return true;
89      }
```

Posmatrajući kod, ne detektuje se niti jedna greška na prvi pogled.

Sada ćemo provesti par testnih slučajeva, kako bismo pokrili svaku liniju koda i zaključili da zaista nema greške.

Na početku je brojLogovanja jednak 0, i svakim novim pozivanjem ove metode nad nekom instancom klase Korisnik, taj atribut se povećava.

- Ukoliko je brojLogovanja veći od 99, neće se dozvoliti logovanje na sistem, te će povratna vrijednost funkcije biti false (jer se korisnik nije uspio logovati)
- Ukoliko je brojLogovanje manji ili jedan 99, logovanje će biti dozvoljeno, povećat će se brojLogovanja i povratna vrijednost funkcije će biti true (jer se korisnik uspio logovati).

Prolazeći kroz dva moguća scenarija (toka) ove metode, ustanovljeno je da je metoda ispravno napisana te da nema grešaka.

2. Klasa Clan, metoda RezervišiMjesto(Lokacija l)

```
74      //Berina Suljic
75      /// <summary>
76      /// Metoda u kojoj se vrši rezervacija parking mjesta za korisnika.
77      /// Ako je osobi istekla članarina, dolazi do pojave izuzetka, kao i u slučaju
78      /// da već postoji rezervisano parking mjesto.
79      /// </summary>
80      /// <param name="l"></param>
        8 references | 5/5 passing
81      public void RezervišiMjesto(Lokacija l)
82      {
83          ProvjeriJeLiČlanarinaIstekla();
84          if (status == Status.Neaktivna) throw new Exception();
85          if (rezervisanoParkingMjesto != null) throw new Exception();
86          rezervisanoParkingMjesto = new Tuple<int, Lokacija>(l.DajTrenutniBrojSlobodnogMjesta(), l);
87      }
```

Metoda već u prvoj liniji poziva metodu ProvjeriJeLiČlanarinaIstekla().

```
89      public void ProvjeriJeLiČlanarinaIstekla()
90      {
91          if (DateTime.Now > aktivnaDo)
92              status = Status.Neaktivna;
93      }
```

Vidimo da će metoda provjeriti da li je datum aktivnaDo prošao. Ukoliko je prošao, postavlja statut na vrijednost Neaktivna.

Sada, krećući se dalje kroz kod metode RezervišiMjesto vidimo da se provjerava da li je status Neaktivan. Ako jeste, baca se izuzetak.

Dalje, ukoliko atribut rezervisanoParkingMjesto nije null vrijednost, tj. ukoliko je član već rezervisao neko parking mjesto, također se baca izuzetak.

Ukoliko se ne baci izuzetak, dolazi se do zadnje linije koda ove metode, u kojoj se atributu rezervisanoParkingMjesto dodjeljuje vrijednost tipa Tuple pri čemu je item1 broj parking mjesta (koje se dobije pozivom metode DajTrenutniBrojSlobodnogMjesta() nad instancom klase Lokacija l, koju smo primili kao parametar), a item2 je l. Tako se u tom atributu čuva koji je broj parkinga koji je član rezervisao, kao i na kojoj lokaciji se nalazi.

Sada ćemo proći kroz par testnih slučajeva:

TESTNI SLUČAJ 1

- Ukoliko imamo instancu klase Član, nazvat ćemo je clan, koja ima atribut aktivnaDo = new DateTime(2021, 5, 6). Status po defaultu na početku ima vrijednost Aktivna. Dok, rezervisanoParkingMjesto ima null vrijednost.
- Također, imamo objekat Lokacija l, sa svim svojim validnim atributima, čiji je atribut kapacitet = 50, dok je trenutna vrijednost brojača 5.
- Ukoliko nad objektom clan pozovemo metodu clan.RezervišiMjesto(l) ovako će ići tok metode: prvo će se pozvati metoda ProvjeriJeLiČlanarinaIstekla(). Obzirom da članarina važi do 6.5.2021, ova metoda neće uraditi ništa jer uslov neće biti zadovoljen. Sljedeća

linija je provjera Statusa, obzirom da on ima vrijednost Aktivna, neće se baciti izuzetak. Zatim, provjerava se da li je rezervisanoParkingMjesto null. Obzirom da jeste, opet se neće baciti izuzetak. Još preostaje posljednja linija koda kojom se uspješno izvršava rezervacija.

TESTNI SLUČAJ 2

- Ukoliko imamo instancu klase Član, nazvat ćemo je clan2, koja ima atribut aktivnaDo = new DateTime(2019, 5, 6). Status po defaultu na početku ima vrijednost Aktivna. Dok, rezervisanoParkingMjesto ima null vrijednost.
- Također, imamo objekat Lokacija l, sa svim svojim validnim atributima, čiji je atribut kapacitet = 50, dok je trenutna vrijednost brojača 5.
- Ukoliko nad objektom clan pozovemo metodu clan.RezervišiMjesto(l) ovako će ići tok metode: prvo će se pozvati metoda ProvjeriJeLiČlanarinaIstekla(). Obzirom da članarina važi do 6.5.2019, ova metoda će postaviti status na vrijednost Neaktivna. Zatim, odmah u narednoj liniji metode RezervišiMjesto bacit će se izuzetak jer je članarina istekla, tj. vrijednost atributa status je Neaktivna. Rezervacija neće biti uspješna.

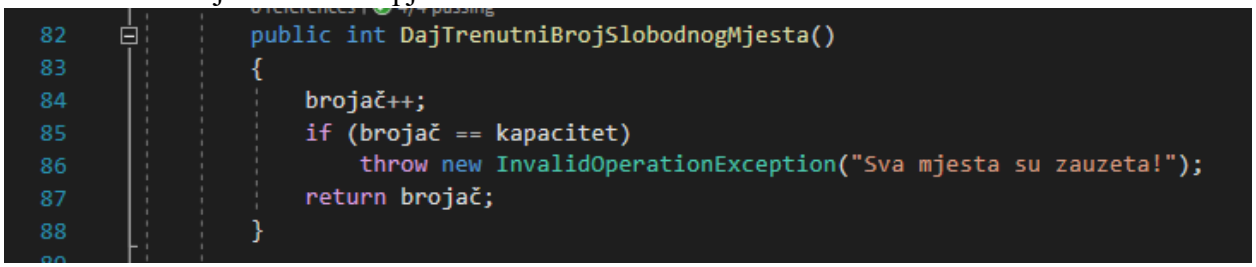
TESTNI SLUČAJ 3

- Ukoliko imamo instancu klase Član, nazvat ćemo je clan, koja ima atribut aktivnaDo = new DateTime(2021, 5, 6). Status po defaultu na početku ima vrijednost Aktivna. Dok, rezervisanoParkingMjesto nije null vrijednost već je tipa Tuple gdje je item1 jednak 4, a item2 je neka lokacija.
- Također, imamo objekat Lokacija l, sa svim svojim validnim atributima, čiji je atribut kapacitet = 50, dok je trenutna vrijednost brojača 5.
- Ukoliko nad objektom clan pozovemo metodu clan.RezervišiMjesto(l) ovako će ići tok metode: prvo će se pozvati metoda ProvjeriJeLiČlanarinaIstekla(). Obzirom da članarina važi do 6.5.2021, ova metoda neće uraditi ništa jer uslov neće biti zadovoljen. Sljedeća linija je provjera Statusa, obzirom da on ima vrijednost Aktivna, neće se baciti izuzetak. Zatim, provjerava se da li je rezervisanoParkingMjesto null. Obzirom da nije, bacit će se izuzetak, jer član već ima rezervisano parking mjesto. Rezervacija neće biti uspješna.

TESTNI SLUČAJ 4

- Ukoliko imamo instancu klase Član, nazvat ćemo je clan, koja ima atribut aktivnaDo = new DateTime(2021, 5, 6). Status po defaultu na početku ima vrijednost Aktivna. Dok, rezervisanoParkingMjesto ima null vrijednost.
- Također, imamo objekat Lokacija l, sa svim svojim validnim atributima, čiji je atribut kapacitet = 50, dok je trenutna vrijednost brojača 49.
- Ukoliko nad objektom clan pozovemo metodu clan.RezervišiMjesto(l) ovako će ići tok metode: prvo će se pozvati metoda ProvjeriJeLiČlanarinaIstekla(). Obzirom da članarina važi do 6.5.2021, ova metoda neće uraditi ništa jer uslov neće biti zadovoljen. Sljedeća linija je provjera Statusa, obzirom da on ima vrijednost Aktivna, neće se baciti izuzetak. Zatim, provjerava se da li je rezervisanoParkingMjesto null. Obzirom da jeste, opet se

neće baciti izuzetak. Još preostaje posljednja linija koda. U ovoj liniji, metoda klase Lokacija DajTrenutniBrojSlobodnogMjesta() će baciti izuzetak jer će se u prvoj liniji te metode brojač povećati na 50, te će time biti jednak kapacitetu i bacit će se izuzetak. Rezervacija neće biti uspješna.



```
82      public int DajTrenutniBrojSlobodnogMjesta()
83      {
84          brojač++;
85          if (brojač == kapacitet)
86              throw new InvalidOperationException("Sva mjesta su zauzeta!");
87          return brojač;
88      }
```

Prošli smo kroz par testnih slučajeva na osnovu kojih smo zaključili da ova metoda nema grešku te da radi ispravno.

3. Klasa Lokacija, metoda ZauzmiMjesto(Clan c)

```
97      public void ZauzmiMjesto(Clan c)
98      {
99          // ovdje se nece provjeravati da li je clanarina istekla jer ce metoda klase Clan baciti izuzetak ako jeste
100         if (brojač < kapacitet) c.RezervišiMjesto(this);
101         else throw new Exception();
102     }
```

U prvoj liniji koda, provjerava se da li je brojač manji od kapaciteta. Ukoliko jeste, nad clanom c poziva se metoda RezervišiMjesto, a kao parametar se šalje this (tekući objekat klase Lokacija).

Ukoliko brojač nije manji od kapaciteta, baca se izuzetak.

Sada ćemo proći kroz par testnih slučajeva kako bismo utvrdili da metoda radi ispravno.

TESTNI SLUČAJ 1

- Ukoliko imamo objekat klase Lokacija l sa svim validnim atributima, te da je brojač jednak 0, a kapacitet 20.
- Zatim, pretpostavimo da imamo objekat klase Clan c sa svim validnim atributima, te da je status Aktivna, rezervisanoParkingMjesto null i aktivnaDo godine 2022.
- Sada, ako nad l pozovemo metodu l.ZauzmiMjesto(c), u prvoj liniji se provjerava da li je brojač manji od kapaciteta. Obzirom da je 0 manje od 20, poziva se metoda c.RezervišiMjesto(this).
- Unutar metode RezervišiMjesto uspješno se mjesto rezerviše na način kako je opisano u inspekciji prethodne metode, jer su svi podaci validni i neće doći do izuzetka.

TESTNI SLUČAJ 2

- Ukoliko imamo objekat klase Lokacija l sa svim validnim atributima, te da je brojač jednak 19, a kapacitet 20.
- Zatim, pretpostavimo da imamo objekat klase Clan c sa svim validnim atributima, te da je status Aktivna, rezervisanoParkingMjesto null i aktivnaDo godine 2022.
- Sada, ako nad l pozovemo metodu l.ZauzmiMjesto(c), u prvoj liniji se provjerava da li je brojač manji od kapaciteta. Obzirom da je 19 manje od 20, poziva se metoda c.RezervišiMjesto(this).
- Unutar metode RezervišiMjesto, doći će se do zadnje linije metode. Međutim unutar te linije koda, poziva se metoda DajTrenutniBrojSlobodnogMjesta() koja će povećati brojač na 20 i baciti izuzetak jer je brojač jednak kapacitetu.
- Zauzimanje mjesta neće biti uspješno.

Ostali testni slučajevi su slični testnim slučajevima opisanim u inspekciji metode klase Clan RezervišiMjesto. Ti testni slučajevi se odnose na „nevalidne“ podatke objekta clan koji će proizvesti bacanje izuzetka.

Provjeravajući testne slučajeve, prolaskom-kroz, utvrdili smo da metoda nema grešku i da radi onako kako se od nje očekuje, ispravno.

4. Klasa Parking, metoda RezervišiParking(Korisnik k, Lokacija l)

```
85      ///Elma
86      /// <summary>
87      /// Metoda u kojoj se vrši rezervisanje parking mjesta za željenu lokaciju korisnika.
88      /// Ukoliko korisnik nije član, ne smije mu se omogućiti rezervacija, kao ni u
89      /// slučaju da na lokaciji nema slobodnih parking mjesta ili da je korisnik
90      /// već rezervisao neko drugo parking mjesto. U suprotnom,
91      /// vrši se rezervacija parking mjesta za korisnika.
92      /// </summary>
93      /// <param name="k"></param>
94      /// <param name="l"></param>
        13 references | 10/10 passing
95      public void RezervišiParking(Korisnik k, Lokacija l)
96      {
97          if (k.GetType() != typeof(Clan))
98              throw new Exception();
99          if(((Clan)k).RezervisanoParkingMjesto != null)
100             throw new Exception();
101         try{
102             ((Clan)k).RezervišiMjesto(l);
103         } catch (InvalidOperationException e) {
104             throw e;
105         }
106     }
107
```

Krenut ćemo od opisa ove metode. Dakle, u prvoj liniji koda provjeravamo da li je korisnik koji je poslan kao parametar zapravo objekat tipa Clan. Ukoliko nije, treba se baciti izuzetak.

Zatim, provjeravamo da li taj clan ima rezervisano parking mjesto, te ukoliko ima ne može rezervisati još jedno, pa se baca izuzetak.

Sada se pokušava pozvati metoda RezervišiMjesto nad objektom k.

Provest ćemo nekoliko testnih slučajeva za ovu metodu:

TESTNI SLUČAJ 1:

- Ukoliko imamo objekat tipa Parking p, sa svim svojim validnim podacima. Također imamo objekat tipa Lokacija l sa svim validnim atributima, gdje je brojač na 0, a kapacitet 20. Zatim imamo i objekat tipa Clan c, sa svim validnim atributima gdje je status Aktivna, aktivnaDo je DateTime(2022, 5,5), a rezervisanoParkingMjesto je null.
- Pozivom metode p.RezervisiParking(c, l), prvo se vrše provjere da li je korisnik član, te da li mu je rezervisano parking mjesto null. Neće se baciti izuzeci.
- Zatim se u try blocku poziva metoda RezervišiMjesto(l) koja će izvršiti uspješnu rezervaciju na način kako je to opisano u inspekciji te metode.

TESTNI SLUČAJ 2:

- Ukoliko imamo objekat tipa Parking p, sa svim svojim validnim podacima. Također imamo objekat tipa Lokacija l sa svim validnim atributima, gdje je brojač na 0, a kapacitet 20. Zatim imamo i objekat tipa Korisnik k, sa svim validnim atributima ali on nije tipa Clan.
- Metoda će baciti izuzetak odmah u prvoj provjeri, te rezervacija neće biti uspješna.

Ono što možemo primijetiti jeste da je u ovoj metodi bespotrebna provjera da li je rezervisanoParkingMjesto null, jer se ta provjera vrši u metodi klase Clan RezervišiMjesto. Tako da je to jedini nedostatak ovog koda koji će se naravno ispraviti.

Prethodni testni slučajevi za metodu RezervišiMjesto su zapravo pokrili i testne slučajeve za metodu RezervišiParking, te smo ispitali sve varijante.

Sada metoda izgleda ovako:

```
95      public void RezervišiParking(Korisnik k, Lokacija l)
96      {
97          if (k.GetType() != typeof(Clan))
98              throw new Exception();
99          try{
100             ((Clan)k).RezervišiMjesto(l);
101         } catch (InvalidOperationException e) {
102             throw e;
103         }
104     }
```

Nakon ispravke viška koda, nema više potencijalnih defekata.