Univerzitet u Sarajevu
Elektrotehnički fakultet Sarajevo
Odsjek za računarstvo i informatiku

Drugi dio: White box testiranje

e-Demokratija

ČLANOVI TIMA:

1. Dalila Kršlak – 18906 (vođa tima)
2. Nejra Adilović – 19061
3. Vildana Tabaković – 18968
4. Berina Zejnilović – 18805

ODGOVORNI NASTAVNIK:

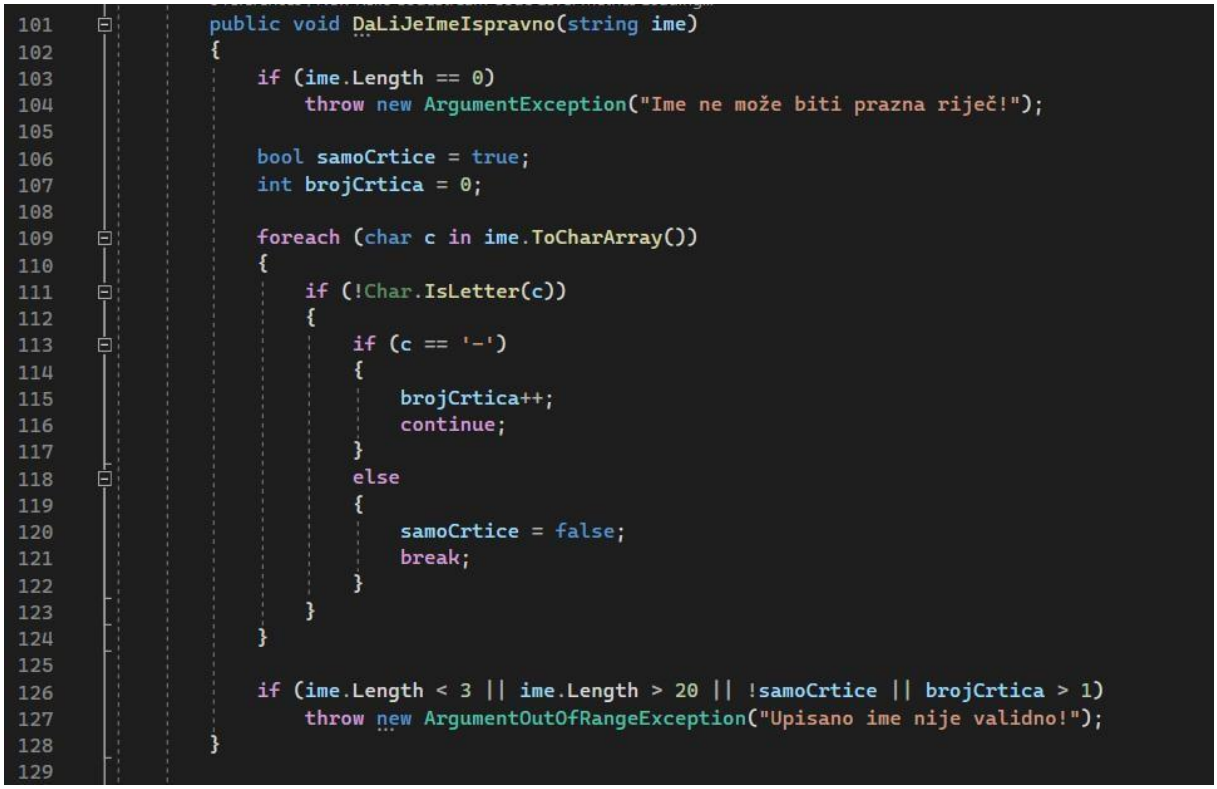
R. prof. dr. Dženana Đonko dipl.el.ing

ODGOVORNI ASISTENT:

Mr. dipl. ing. Neira Novalić

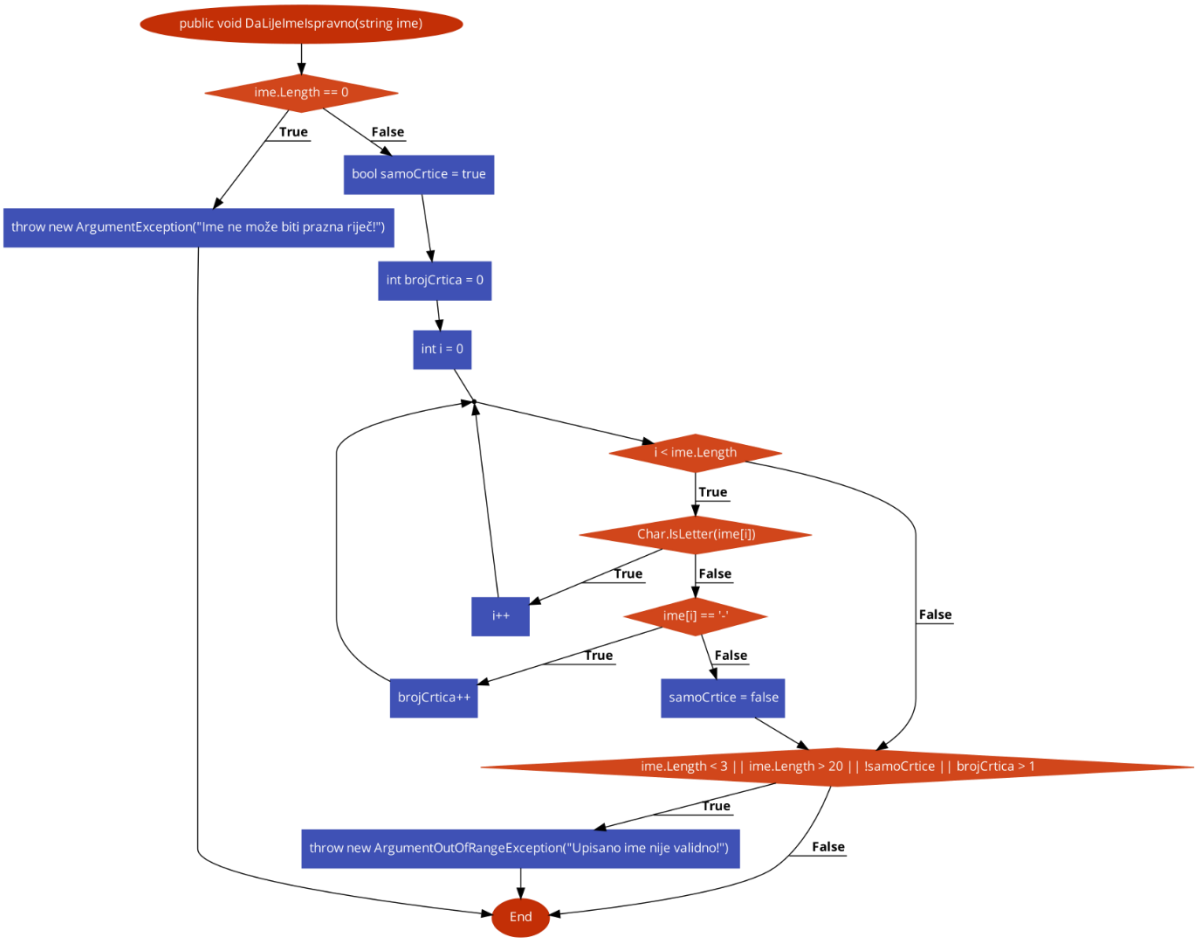
ČLAN 1: Dalila Kršlak

U metrikama programskog rješenja pronađena je metoda koja ima visok stepen održivosti, a da pritom sadrži uslove i petlje. To je metoda DaLiJeImeIspravno.

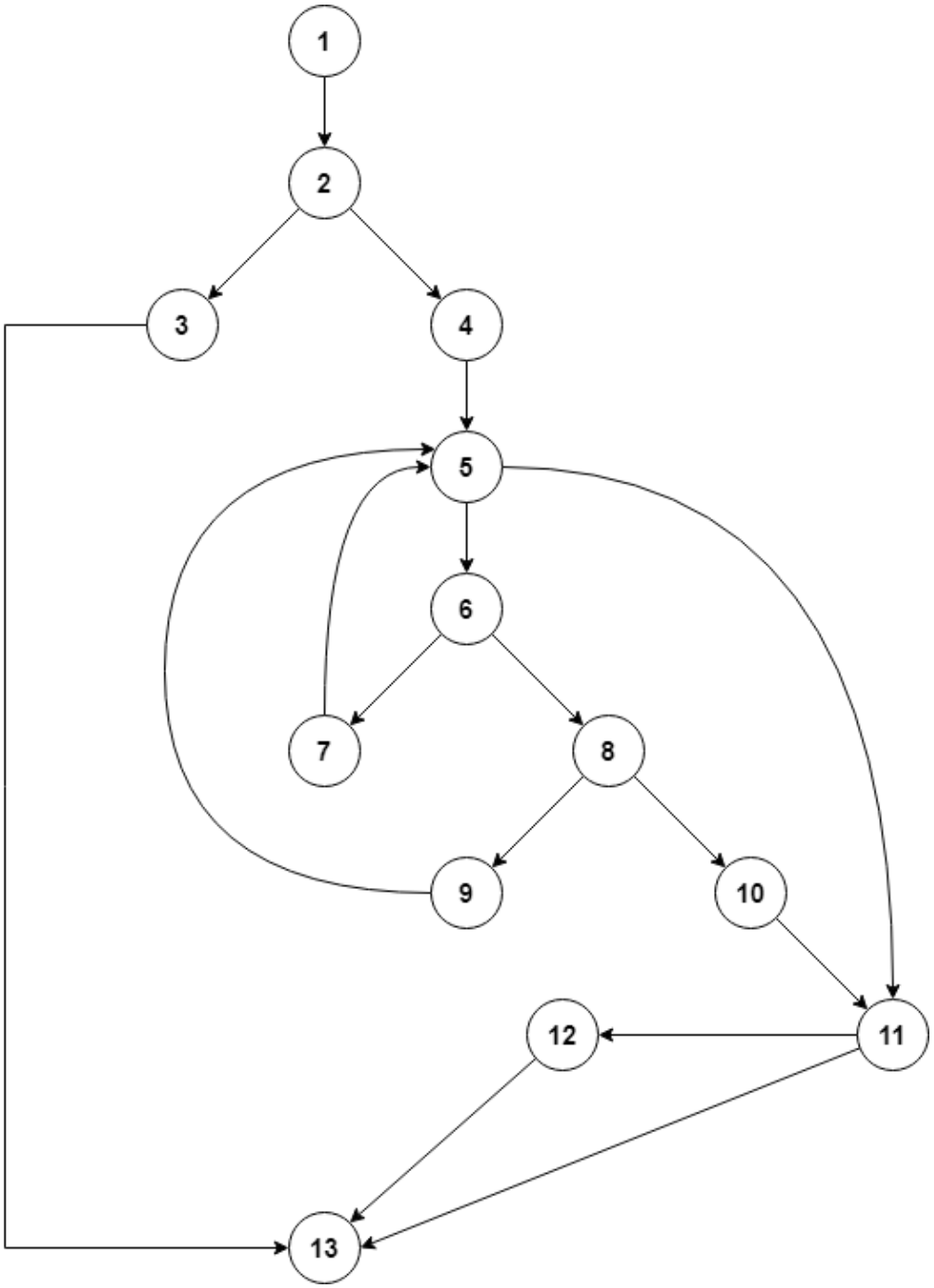


```
101     public void DaLiJeImeIspravno(string ime)
102     {
103         if (ime.Length == 0)
104             throw new ArgumentException("Ime ne može biti prazna riječ!");
105
106         bool samoCrtice = true;
107         int brojCrtica = 0;
108
109         foreach (char c in ime.ToCharArray())
110         {
111             if (!Char.IsLetter(c))
112             {
113                 if (c == '-')
114                 {
115                     brojCrtica++;
116                     continue;
117                 }
118                 else
119                 {
120                     samoCrtice = false;
121                     break;
122                 }
123             }
124         }
125
126         if (ime.Length < 3 || ime.Length > 20 || !samoCrtice || brojCrtica > 1)
127             throw new ArgumentOutOfRangeException("Upisano ime nije validno!");
128     }
129
```

Kontrolni graf za metodu DaLiJeImeIspravno:



Graf programskog toka za metodu DaLiJeImeIspravno:



Obuhvat iskaza/linija (Line coverage)

Za obuhvat svih iskaza/linija potrebna su 4 puta:

- Put 1: 1 → 2 → 3 → 13
- Put 2: 1 → 2 → 4 → 5 → 6 → 7 → 5 → 11 → 13
- Put 3: 1 → 2 → 4 → 5 → 6 → 8 → 9 → 5 → 11 → 12 → 13
- Put 4: 1 → 2 → 4 → 5 → 6 → 8 → 10 → 11 → 12 → 13

Testovi:

Za put 1:

```
[TestMethod]
[ExpectedException(typeof(ArgumentException))]
| 0 references | Please sign-in to New Relic CodeStream to see Code Level Metrics
public void DaLiJeImeIspravno_PraznoIme_BacaIzuzetak()
{
    Glasac g = new Glasac();
    string ime = "";
    g.DaLiJeImeIspravno(ime);
    g = new Glasac(ime, "Kršlak", new DateTime(2001, 11, 23));
}
```

Za put 2:

```
[TestMethod]
| 0 references | Please sign-in to New Relic CodeStream to see Code Level Metrics
public void DaLiJeImeIspravno_Ime_NemaIzuzetka()
{
    Glasac g = new Glasac("Dalila", "Kršlak", new DateTime(2001, 11, 23));
    g.DaLiJeImeIspravno(g.Ime);
}
```

Za put 3:

```
[TestMethod]
[ExpectedException(typeof(ArgumentOutOfRangeException))]
| 0 references | Please sign-in to New Relic CodeStream to see Code Level Metrics
public void DaLiJeImeIspravno_ImeSaCrticama_BacaIzuzetak()
{
    Glasac g = new Glasac("Dalila-Daki-Dals", "Kršlak", new DateTime(2001, 11, 23));
    g.DaLiJeImeIspravno(g.Ime);
}
```

Za put 4:

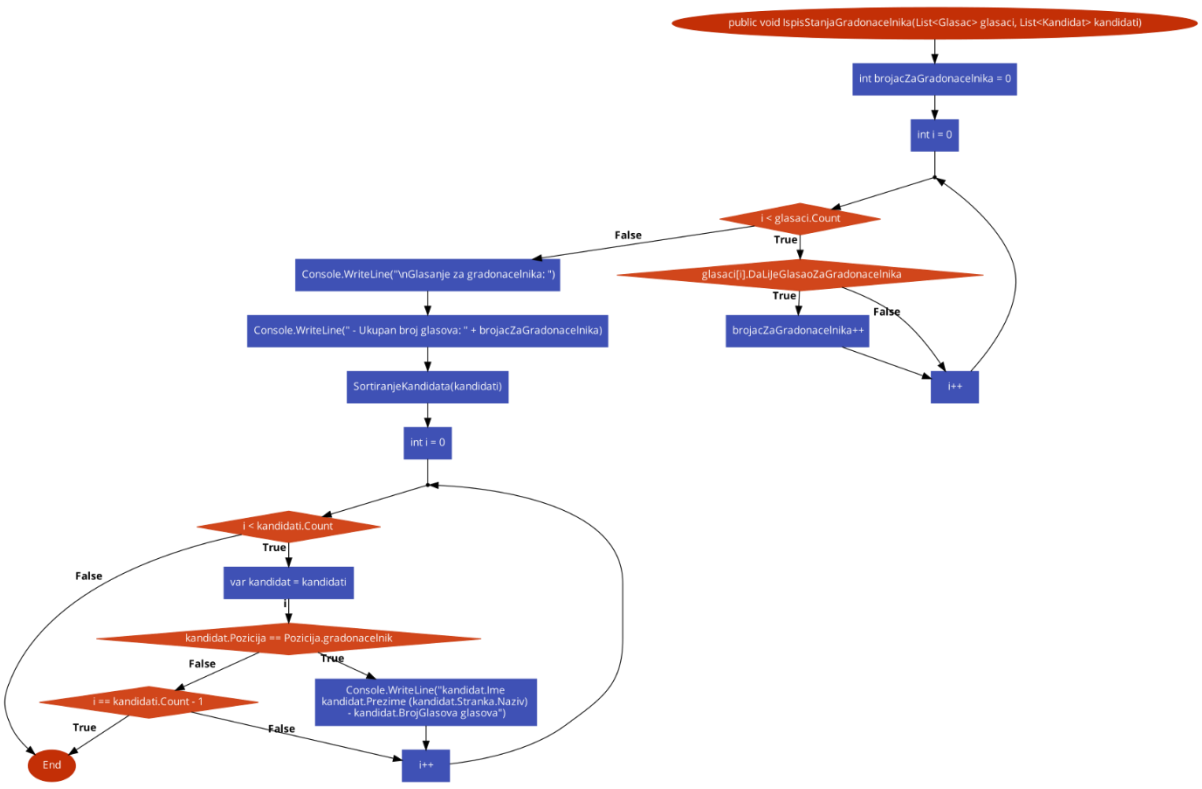
```
[TestMethod]
[ExpectedException(typeof(ArgumentOutOfRangeException))]
| 0 references | Please sign-in to New Relic CodeStream to see Code Level Metrics
public void DaLiJeImeIspravno_ZnakoviUImenu_BacaIzuzetak()
{
    Glasac g = new Glasac("Dalila2311", "Kršlak", new DateTime(2001, 11, 23));
    g.DaLiJeImeIspravno(g.Ime);
}
```

ČLAN 2: Nejra Adilović

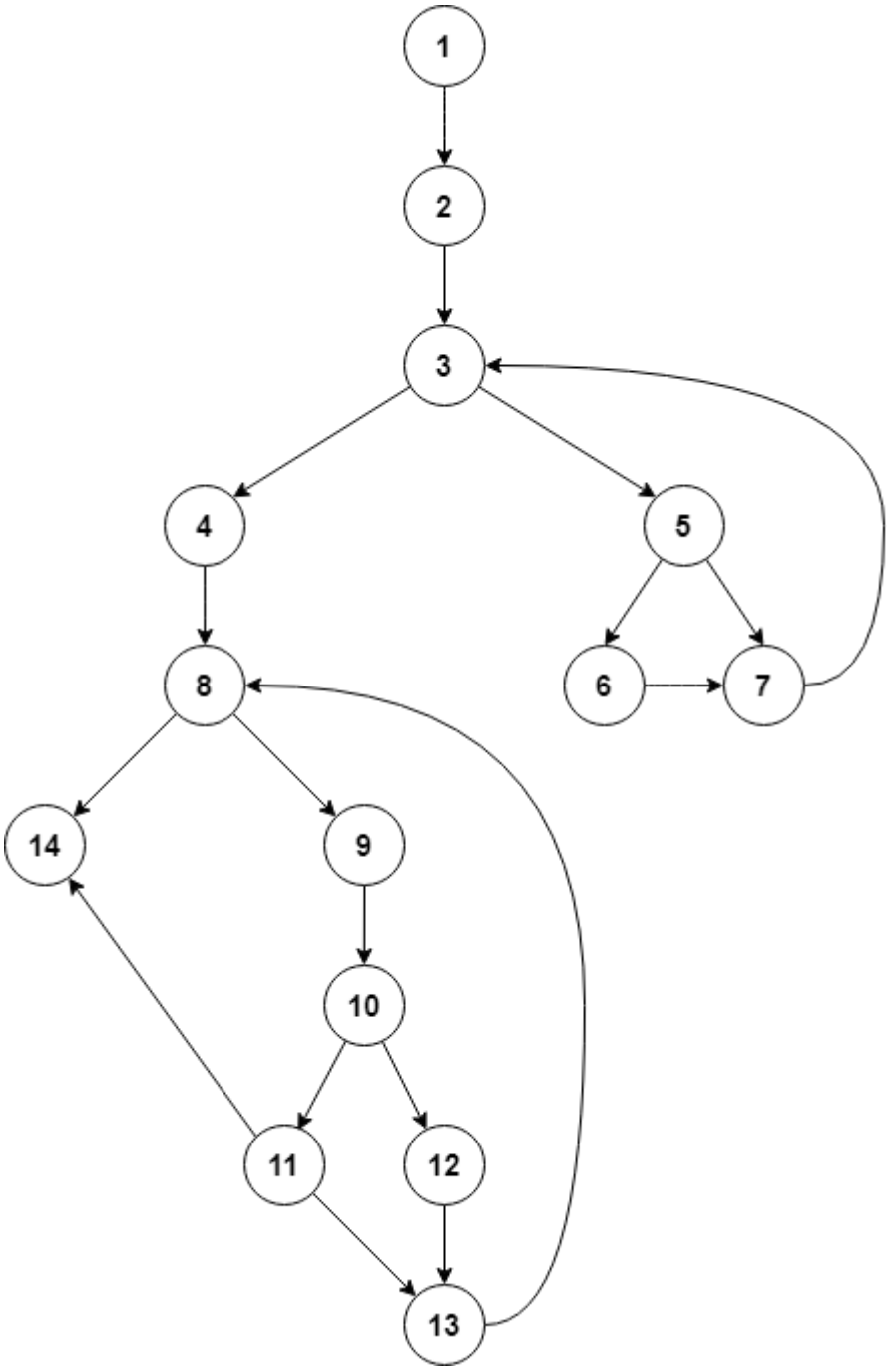
U metrikama programskog rješenja pronađena je metoda koja ima visok stepen održivosti, a da pritom sadrži uslove i petlje. To je metoda IspisStanjaGradonačelnika.

```
2 references | Please sign in to New Relic CodeStream to see Code Level Metrics
public void IspisStanjaGradonacelnika(List<Glasac> glasaci, List<Kandidat> kandidati)
{
    int brojacZaGradonacelnika = 0;
    foreach (Glasac g in glasaci)
    {
        if (g.DaLiJeGlasaoZaGradonacelnika)
            brojacZaGradonacelnika++;
    }
    Console.WriteLine("\nGlasanje za gradonacelnika: ");
    Console.WriteLine(" - Ukupan broj glasova: " + brojacZaGradonacelnika);
    SortiranjeKandidata(kandidati);
    foreach (var kandidat in kandidati)
    {
        if (kandidat.Pozicija == Pozicija.gradonacelnik)
        {
            Console.WriteLine($"     {kandidat.Ime} {kandidat.Prezime} ({kandidat.Stranka.Naziv}) - {kandidat.BrojGlasova} glasova");
        }
    }
}
```

Kontrolni graf za metodu IspisStanjaGradonačelnika:



Graf programskog toka za metodu IspisStanjaGradonačelnika:



Obuhvat iskaza/linija (Line coverage)

Za obuhvat svih iskaza/linija potrebna su 4 puta:

- Put 1: 1 → 2 → 3 → 4 → 8 → 14
- Put 2: 1 → 2 → 3 → 5 → 6 → 7 → 3 → 4 → 8 → 9 → 10 → 11 → 14
- Put 3: 1 → 2 → 3 → 5 → 6 → 7 → 3 → 4 → 8 → 9 → 10 → 11 → 13→ 8 → 14
- Put 4: 1 → 2 → 3 → 5 → 7 → 3 → 4 → 8 → 9 → 10 → 12 → 13 → 8 → 14

Testovi:

Za put 1:

```
[TestMethod]
| 0 references | New Relic CodeStream Code Level Metrics Loading...
public void TestIspisStanjaGradonacelnika_Put1()
{
    var glas = new Glas();
    var glasaci = new List<Glasac>();
    var kandidati = new List<Kandidat>();
    var ocekivaniIzlaz = new StringBuilder();
    ocekivaniIzlaz.AppendLine("Glasanje za gradonacelnika: ");
    ocekivaniIzlaz.AppendLine(" - Ukupan broj glasova: 0");
    ocekivaniIzlaz.AppendLine(" - Prva tri mjesta: ");

    var izlazKonzole = IspisKonzole(() => glas.IspisStanjaGradonacelnika(glasaci, kandidati));

    StringAssert.Contains(izlazKonzole, ocekivaniIzlaz.ToString());
}
```

Za put 2:

```
[TestMethod]
| 0 references | New Relic CodeStream Code Level Metrics Loading...
public void TestIspisStanjaGradonacelnika_Put2()
{
    var glas = new Glas();
    var glasaci = new List<Glasac>
    {
        new Glasac { DaLiJeGlasaoZaGradonacelnika = true },
        new Glasac { DaLiJeGlasaoZaGradonacelnika = true },
            new Glasac { DaLiJeGlasaoZaGradonacelnika = true }
    };

    var kandidati = new List<Kandidat>
    {
        new Kandidat { Pozicija = Pozicija.nacelnik, BrojGlasova = 3, Ime = "K3", Prezime = "Prezime3", Stranka = new Stranka { Naziv = "StrankaB" } }
    };

    var ocekivaniIzlaz = new StringBuilder();
    ocekivaniIzlaz.AppendLine("Glasanje za gradonacelnika: ");
    ocekivaniIzlaz.AppendLine(" - Ukupan broj glasova: 3");
    ocekivaniIzlaz.AppendLine(" - Prva tri mjesta: ");

    var izlazKonzole = IspisKonzole(() => glas.IspisStanjaGradonacelnika(glasaci, kandidati));

    StringAssert.Contains(izlazKonzole, ocekivaniIzlaz.ToString());
}
```

Za put 3:

```
[TestMethod]
| 0 references | New Relic CodeStream Code Level Metrics Loading...
public void TestIspisStanjaGradonacelnika_Put3()
{
    var glas = new Glas();
    var glasaci = new List<Glasac>
    {
        new Glasac { DaLiJeGlasaoZaGradonacelnika = true },
        new Glasac { DaLiJeGlasaoZaGradonacelnika = true },
            new Glasac { DaLiJeGlasaoZaGradonacelnika = true }
    };

    var kandidati = new List<Kandidat>
    {
        new Kandidat { Pozicija = Pozicija.vijecnik, BrojGlasova = 5, Ime = "K1", Prezime = "Prezime1", Stranka = new Stranka { Naziv = "StrankaA" } },
        new Kandidat { Pozicija = Pozicija.nacelnik, BrojGlasova = 4, Ime = "K2", Prezime = "Prezime2", Stranka = new Stranka { Naziv = "StrankaA" } },
        new Kandidat { Pozicija = Pozicija.nacelnik, BrojGlasova = 3, Ime = "K3", Prezime = "Prezime3", Stranka = new Stranka { Naziv = "StrankaB" } }
    };

    var ocekivaniIzlaz = new StringBuilder();
    ocekivaniIzlaz.AppendLine("Glasanje za gradonacelnika: ");
    ocekivaniIzlaz.AppendLine(" - Ukupan broj glasova: 3");
    ocekivaniIzlaz.AppendLine(" - Prva tri mjesta: ");

    var izlazKonzole = IspisKonzole(() => glas.IspisStanjaGradonacelnika(glasaci, kandidati));

    StringAssert.Contains(izlazKonzole, ocekivaniIzlaz.ToString());
}
```

Za put 4:

```
[TestMethod]
0 references | New Relic CodeStream Code Level Metrics Loading...
public void TestIspisStanjaGradonacelnika_Put4()
{
    var glas = new Glas();
    var glasaci = new List<Glasac>
    {
        new Glasac { DaLiJeGlasaoZaGradonacelnika = false },
        new Glasac { DaLiJeGlasaoZaGradonacelnika = false },
           new Glasac { DaLiJeGlasaoZaGradonacelnika = false }
    };

    var kandidati = new List<Kandidat>
    {
        new Kandidat { Pozicija = Pozicija.gradonacelnik, BrojGlasova = 5, Ime = "K1", Prezime = "Prezime1", Stranka = new Stranka { Naziv = "StrankaA" } },
        new Kandidat { Pozicija = Pozicija.gradonacelnik, BrojGlasova = 5, Ime = "K1", Prezime = "Prezime1", Stranka = new Stranka { Naziv = "StrankaA" } },
        new Kandidat { Pozicija = Pozicija.gradonacelnik, BrojGlasova = 4, Ime = "K2", Prezime = "Prezime2", Stranka = new Stranka { Naziv = "StrankaA" } }
    };

    var ocekivaniIzlaz = new StringBuilder();
    ocekivaniIzlaz.AppendLine("Glasanje za gradonacelnika: ");
    ocekivaniIzlaz.AppendLine(" - Ukupan broj glasova: 0");
    ocekivaniIzlaz.AppendLine(" - Prva tri mjesta: ");

    var izlazKonzole = IspisKonzole(() => glas.IspisStanjaGradonacelnika(glasaci, kandidati));

    StringAssert.Contains(izlazKonzole, ocekivaniIzlaz.ToString());
}
```

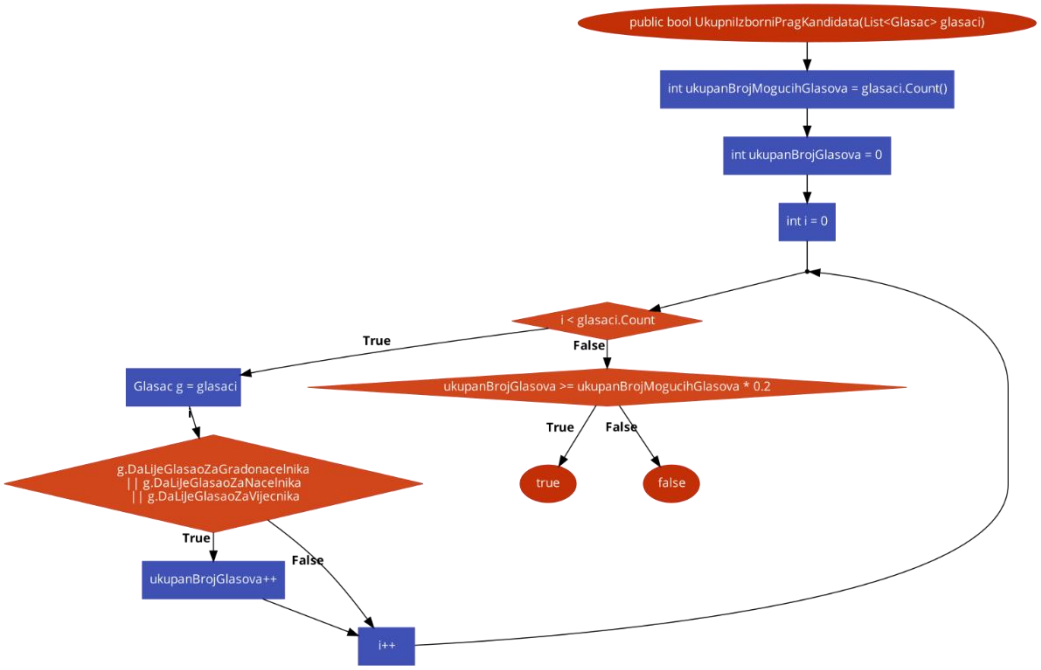
ČLAN 3: Berina Zejnilović

U metrikama programskog rješenja pronađena je metoda koja ima visok stepen održivosti, a da pritom sadrži uslove i petlje. To je metoda UkupniIzborniPragKandidata.

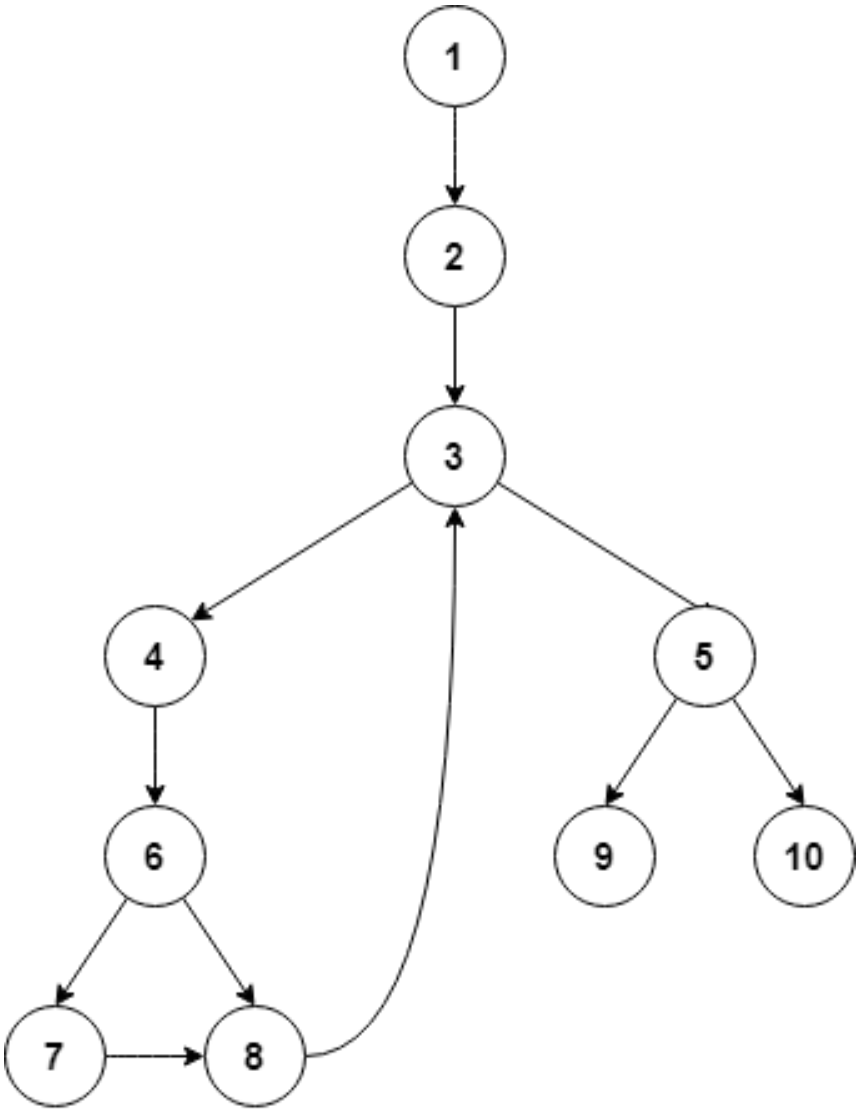
```
3 references | New Relic CodeStream Code Level Metrics Loading...
public bool UkupniIzborniPragKandidata(List<Glasac> glasaci)
{
    int ukupanBrojMogucihGlasova = glasaci.Count();
    int ukupanBrojGlasova = 0;
    foreach (Glasac g in glasaci)
    {
        if (g.DaLiJeGlasaoZaGradonacelnika || g.DaLiJeGlasaoZaNacelnika || g.DaLiJeGlasaoZaVijecnika)
            ukupanBrojGlasova++;
    }

    if (ukupanBrojGlasova >= ukupanBrojMogucihGlasova * 0.2)
        return true;
    return false;
}
```

Kontrolni graf za metodu UkupniIzborniPragKandidata:



Graf programskog toka za metodu UkupniIzborniPragKandidata:



Obuhvat iskaza/linija (Line coverage)

Za obuhvat svih iskaza/linija potrebna su 2 puta:

- Put 1: 1 → 2 → 3 → 4 → 6 → 7 → 8 → 3 → 5 → 10
- Put 2: 1 → 2 → 3 → 4 → 6 → 8 → 3 → 5 → 9

Testovi:

Za put 1:

```
[TestMethod]
| 0 references | Please sign-in to New Relic CodeStream to see Code Level Metrics
public void TestUkupnogIzbornogPragaKandidata2()
{
    var glas = new Glas();
    var glasaci = new List<Glasac>
        {
            new Glasac { DaLiJeGlasaoZaGradonacelnika = false }
        };
    var result = glas.UkupniIzborniPragKandidata(glasaci);
    Assert.IsFalse(result);
}
```

Za put 2:

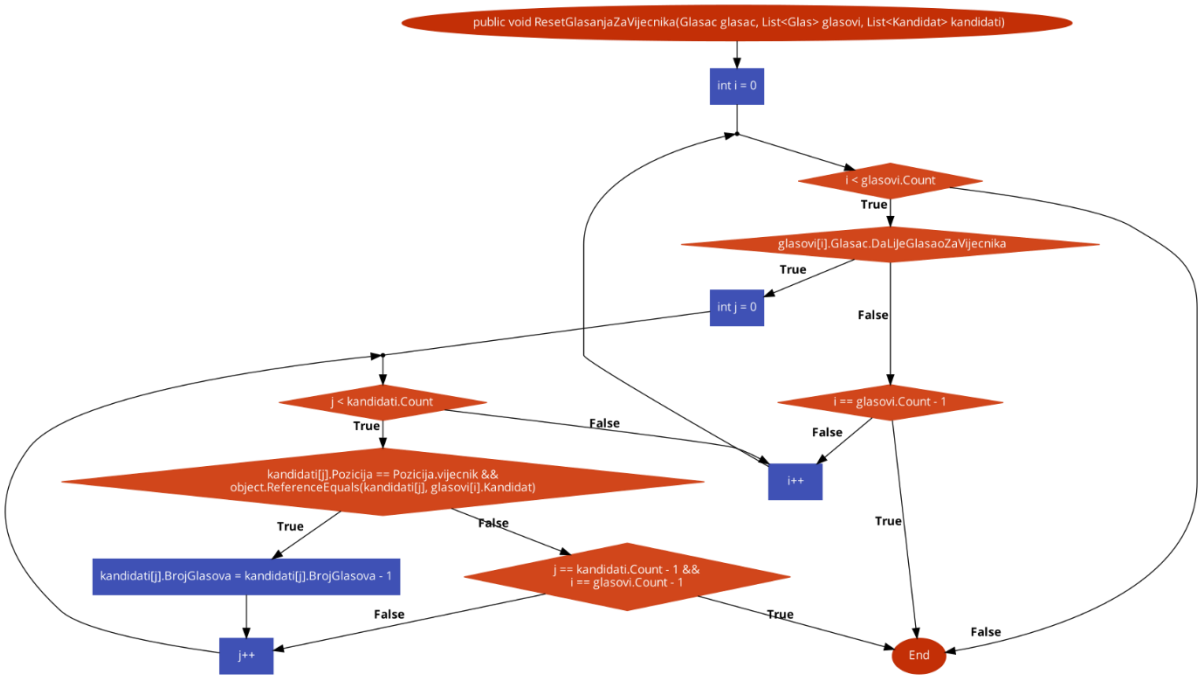
```
[TestMethod]
| 0 references | Please sign-in to New Relic CodeStream to see Code Level Metrics
public void TestUkupnogIzbornogPragaKandidata1()
{
    var glas = new Glas();
    var glasaci = new List<Glasac>
    {
        new Glasac { DaLiJeGlasaoZaGradonacelnika = true },
        new Glasac { DaLiJeGlasaoZaNacelnika = true },
        new Glasac { DaLiJeGlasaoZaVijecnika = true }
    };
    var result = glas.UkupniIzborniPragKandidata(glasaci);
    Assert.IsTrue(result);
}
```

ČLAN 4: Vildana Tabaković

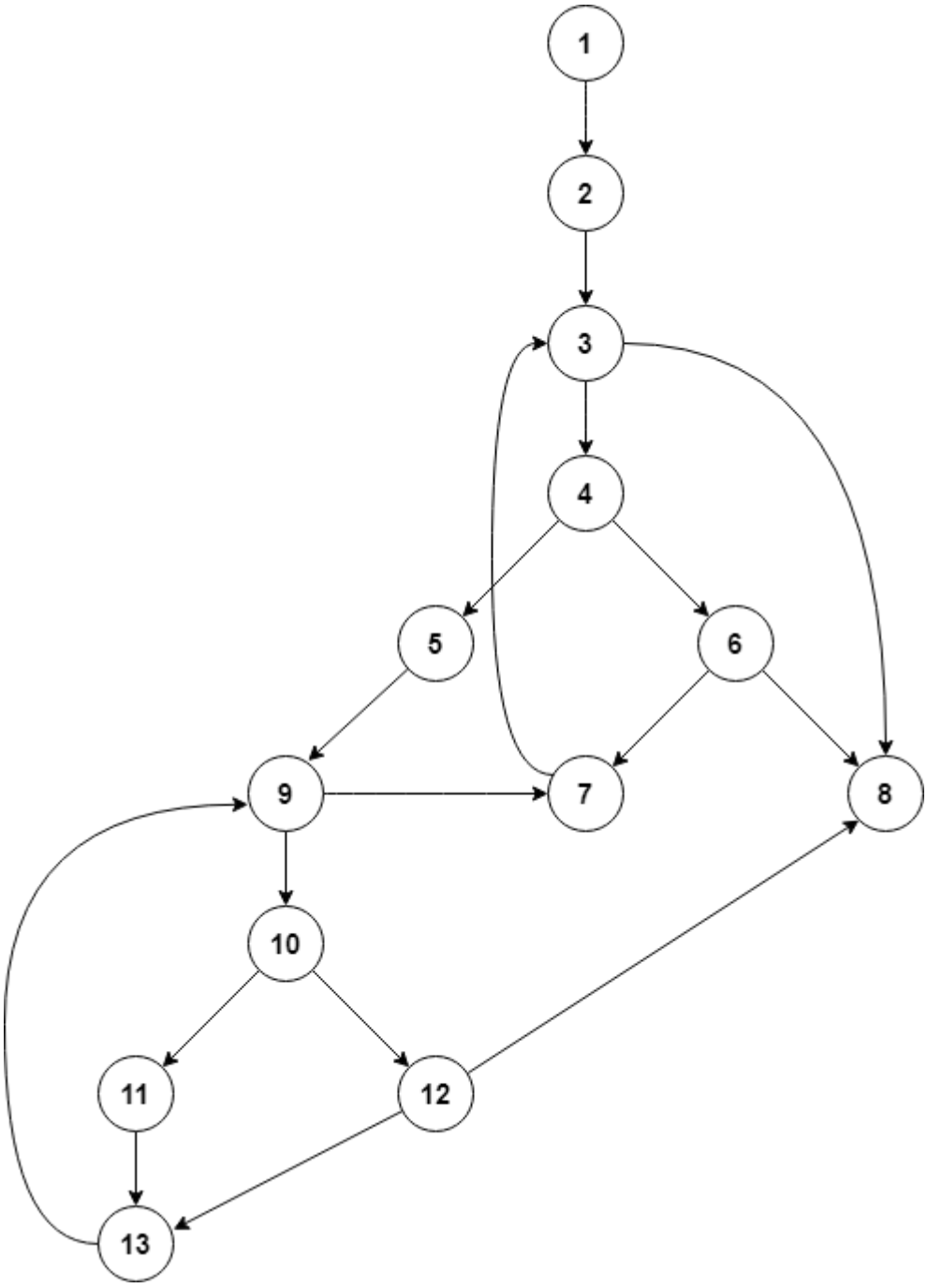
U metrikama programskog rješenja pronađena je metoda koja ima visok stepen održivosti, a da pritom sadrži uslove i petlje. To je metoda ResetGlasanjaZaVijecnika.

```
188     public void ResetGlasanjaZaVijecnika(Glasac glasac, List<Glas> glasovi, List<Kandidat> kandidati)
189     {
190         for (int i = 0; i < glasovi.Count; i++)
191         {
192             if (glasovi[i].Glasac.DaLiJeGlasaoZaVijecnika)
193             {
194                 for (int j = 0; j < kandidati.Count; j++)
195                 {
196                     if (kandidati[j].Pozicija == Pozicija.vijecnik && object.ReferenceEquals(kandidati[j], glasovi[i].Kandidat))
197                     {
198                         kandidati[j].BrojGlasova = kandidati[j].BrojGlasova - 1;
199                     }
200                 }
201             }
202         }
203     }
```

Kontrolni graf za metodu ResetGlasanjaZaVijecnika:



Graf programskog toka za metodu ResetGlasanjaZaVijecnika:



Obuhvat iskaza/linija (Line coverage)

Za obuhvat svih iskaza/linija potrebna su 3 puta:

- Put 1: 1 → 2 → 3 → 4 → 6 → 8
- Put2: 1 → 2 → 3 → 4 → 5 → 9 → 10 → 12 → 8
- Put3: 1 → 2 → 3 → 4 → 5 → 9 → 10 → 11 → 13 → 9 → 7 → 3 → 8

Testovi:

Za put 1:

```
[TestMethod]
| 0 references | New Relic CodeStream Code Level Metrics Loading...
public void ResetGlasanjaZaVijecnika_NemaGlasovaZaVijecnika_BrojGlasovaSeNeMijenja()
{
    var glasac = new Glasac();
    var vijecnik = new Kandidat("Kandidat", "Prvi", new DateTime(1977, 10, 12), Pozicija.vijecnik, "Opis", null);
    var drugiKandidat = new Kandidat("Kandidat", "Prvi", new DateTime(1977, 10, 12), Pozicija.gradonacelnik, "Opis", null);
    var kandidati = new List<Kandidat> { vijecnik, drugiKandidat };
    var glasovi = new List<Glas> { new Glas(new Glasac(), drugiKandidat) };
    var originalniBrojGlasova = vijecnik.BrojGlasova;

    Glas glas = new Glas();

    glas.ResetGlasanjaZaVijecnika(glasac, glasovi, kandidati);

    Assert.AreEqual(originalniBrojGlasova, vijecnik.BrojGlasova);
}
```

Za put 2:

```
[TestMethod]
| 0 references | New Relic CodeStream Code Level Metrics Loading...
public void ResetGlasanjaZaVijecnika_GlasacNijeGlasaoZaVijecnika_BrojGlasovaSeNeMijenja()
{
    var glasac = new Glasac();
    var kandidati = new List<Kandidat> {
        new Kandidat("Kandidat", "Prvi", new DateTime(1977,10,12), Pozicija.gradonacelnik, "Opis", null)
    };
    var glasovi = new List<Glas> { new Glas(glasac, kandidati[0]) };
    var originalniBrojGlasova = kandidati[0].BrojGlasova;
    Glas glas = new Glas();
    glas.ResetGlasanjaZaVijecnika(glasac, glasovi, kandidati);

    Assert.AreEqual(originalniBrojGlasova, kandidati[0].BrojGlasova);
}
```

Za put 3:

```
[TestMethod]
| 0 references | New Relic CodeStream Code Level Metrics Loading...
public void ResetGlasanjaZaVijecnika_GlasacGlasaoZaVijecnika_BrojGlasovaSeSmanjuje()
{
    var glasac = new Glasac();
    var vijecnik = new Kandidat("Kandidat", "Prvi", new DateTime(1977, 10, 12), Pozicija.vijecnik, "Opis", null);
    var drugiKandidat = new Kandidat("Kandidat", "Prvi", new DateTime(1977, 10, 12), Pozicija.gradonacelnik, "Opis", null);
    var kandidati = new List<Kandidat> { vijecnik, drugiKandidat };
    var glasovi = new List<Glas> { new Glas(glasac, vijecnik), new Glas(new Glasac(), drugiKandidat) };
    Glas glas = new Glas();

    glas.ResetGlasanjaZaVijecnika(glasac, glasovi, kandidati);

    Assert.AreEqual(0, vijecnik.BrojGlasova);
}
```