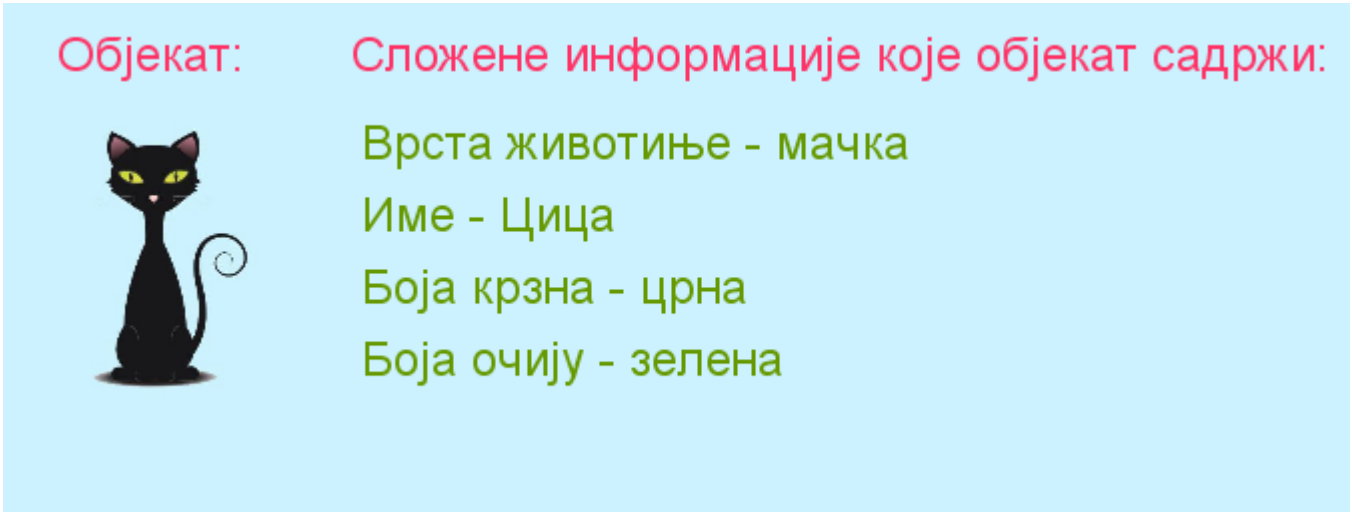
Опис класе

Класа представља уопштени шаблон на основу кога се израђују објекти. Објекат је нека променљива која садржи податке који садрже сложене информације.

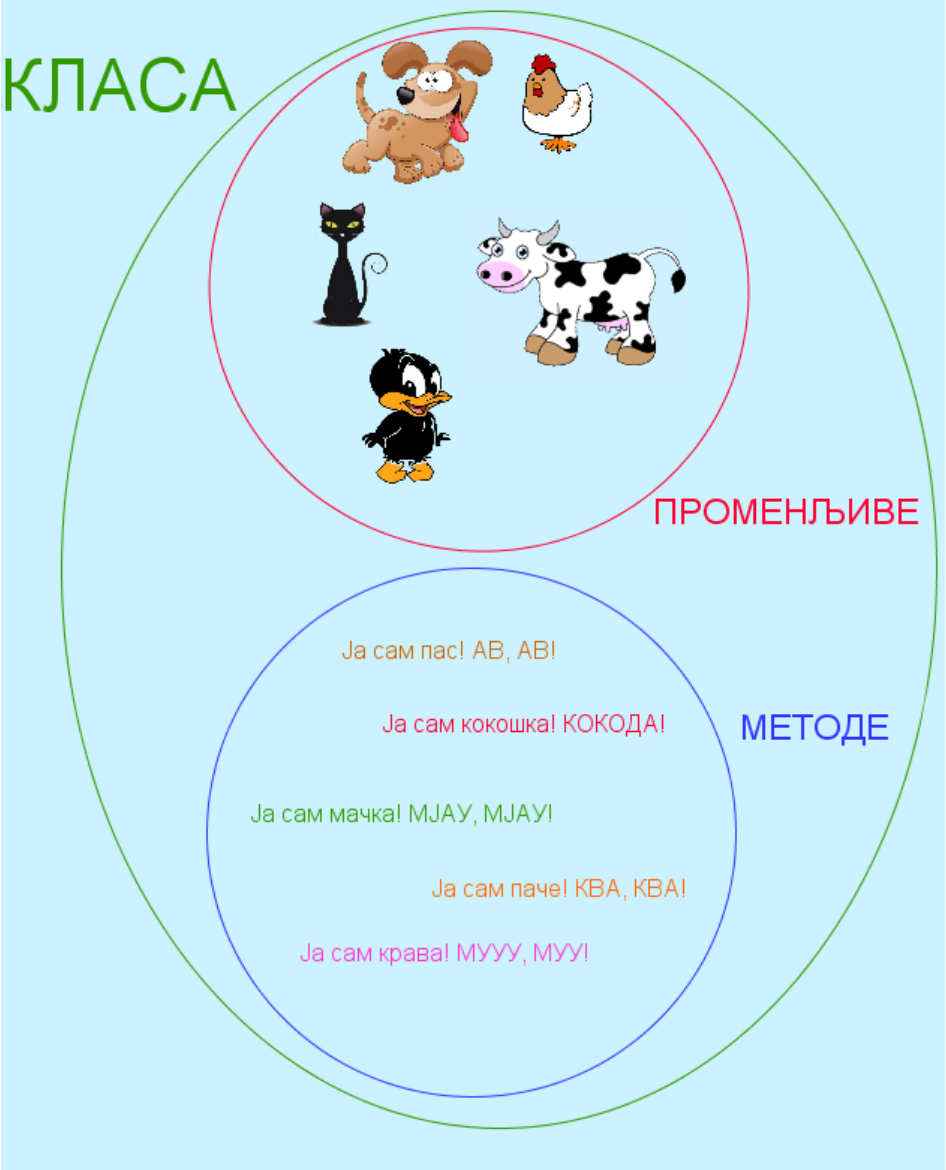


Објекат и структура објекта

Уколико кућне љубимце посматрамо као објекте, онда ће њихова класа бити **Ljubimac**. Љубимце разликујемо по врсти, боји, начину оглашавања.

Најчешћи кућни љубимци су мачке, пси и папагаји. Мачке међусобно можемо разликовати по врсти и боји крзна, псе и папагаје такође, а једни од других се могу разликовати и по начину оглашавања. Заједничке карактеристике љубимаца су врста, боја крзна или перја. Сваком љубимцу можемо придружити неке од ових особина. Ове карактеристике се називају **атрибути**. То ће бити атрибути (променљиве) класе **Ljubimac.**

Које ће радње кућни љубимац обављати зависи од тога које особине има тај кућни љубимац. Радње које кућни љубимац обавља су заправо функције. Функције које се врше над атрибутима класе називамо **методи класе**. Метод који можемо дефинисати за кућног љубимца може бити начин оглашавања. Сваки љубимац одређене врсте има јединствен начин оглашавања.



Структура класе

Дефинисање класе: Користи се кључна реч ***class*** коју прати име класе и у оквиру витичастих заграда детаљи дефиниције. Детаљи дефиниције су атрибути (односно промењљиве) и методе.

```
class ImeKlase
{

Članovi (promenljive i metode)

}
```

Методе дефинишу операције које се могу применити над објектима класе, обично оперишу над промењљивим из класе. Једна од метода класе је конструктор. Конструктор је метод класе за који важе следеће особине:

1. **има исто име као класа**
2. **нема повратну вредност.**

Конструктор нам је неопходан кад год желимо да креирамо објекат одређене класе. Објекат увек морамо креирати да бисмо могли да радимо са њим. Погледајмо следећи пример:

```
class Ljubimac
{
    //Ovo su atributi klase Ljubimac.
    public String vrsta;
    public String ime;

    //Ovo je konstruktor. Moramo napraviti ljubimca da bismo mogli na njega da
primenjujemo metode.
    public Ljubimac(string ime, string vrsta)
    {
        this.ime = ime;
        this.vrsta = vrsta;

    }

    //Ovo je metod klase Ljubimac. Kao argument uzima vrstu ljubimca.
    public string predstavljanje(string vrsta)
    {
        return "Ja sam " + vrsta;
    }
}
```

Под дефиницијом класе подразумева се навођење свих чланова класе. Кад је класа дефинисана, могу се дефинисати и њени објекти. Дефиниција класе у потпуности одређује класу, у нашем примеру, потпуно одређује љубимца: врсту љубимца, његово име и његово представљање (то је метод).

Пре позива метода везаног за класу неопходно је помоћу конструктора креирати одређени објекат на који ће се тај метод применити. Дакле, прво креирамо нашег љубимца, то ће бити објекат типа **Ljubimac**. Објекат креирамо помоћу оператора ***new*** иза кога наводимо конструктор са одговарајућим аргументима.

```
//Pravimo ljubimca pomoću konstruktora iz klase
Ljubimac ljubimac = new Ljubimac(ime, vrsta);
```

Уколико у класи нисмо дефинисали конструктор, користимо подразумевани конструктор који увек постоји, а потом додељујемо вредности одређеним атрибутима.

```
//Pomoću podrazumevanog konstruktora pravimo ljubimca
Ljubimac ljubimac = new Ljubimac();
//Dodeljujemo mu ime
ljubimac.ime = "Pera";
//Dodeljujemo mu vrstu
ljubimac.vrsta = "macka";
```

Одређени метод позивамо помоћу оператора тачка.

```
ljubimac.predstavljanje(vrsta);
```

Математички пример за класу и методе класе.

Направићемо класу Sfera. Атрибут класе Sfera биће полупречник. Дефинисаћемо два метода ове класе - први ће рачунати запремину, а други површину.

```
class Sfera
{
    //Atributi klase Sfera
    public Double poluprecnik;

    //Definišemo konstruktor
    public Sfera(double poluprecnik)
    {
      //Pravimo sferu da bismo na nju primenili metode
        this.poluprecnik = poluprecnik;
    }

    //Metod klase Sfera, računa površinu
    public Double povrsina(double poluprecnik)
    {
      //Matematičkoj konstanti PI pristupamo pomoću operatora tačka. Math je definisana
klasa u C#
        return 4 * poluprecnik * poluprecnik * Math.PI;
    }

    //Metod klase Sfera, računa zapreminu
    public Double zapremina(double poluprecnik)
    {

        return (4 / 3) * poluprecnik * poluprecnik * poluprecnik * Math.PI;
    }
}
```

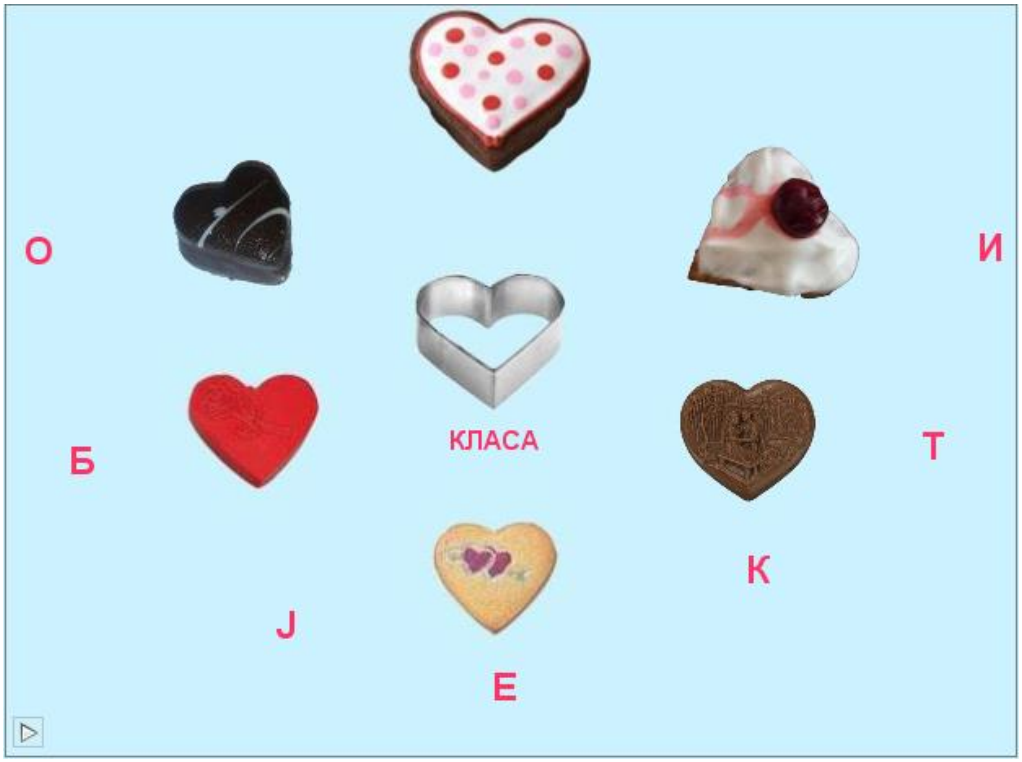
Као и у претходном примеру, прво помоћу конструктора правимо објекат типа **Sfera.**

```
Sfera loptica = new Sfera(r);
```

Методе позивамо помоћу оператора тачка.

```
povrsina = loptica.povrsina(r);
zapremina = loptica.zapremina(r);
```

Напомена: Класа заправо представља шаблон на основу кога се израђују различити објекти. Она сама по себи не представља конкретан објекат, већ описује оно што објекат представља и ради. Она је калуп који нам служи за креирање једног или више различитих објеката као што нам калуп за колаче омогућава да направимо више различитих колача истог облика.



Сликовит приказ класе и објеката

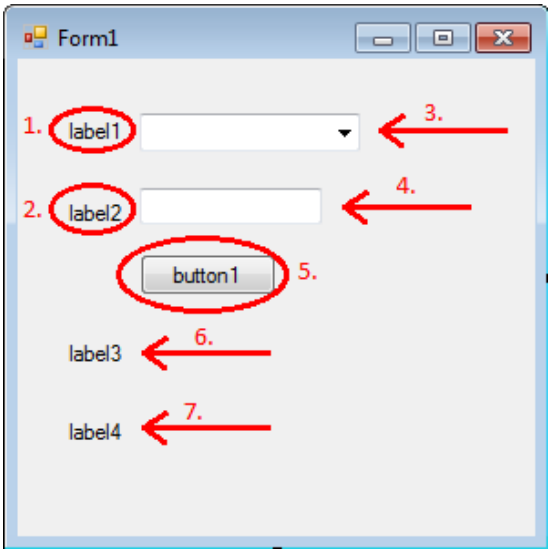
Класа је скуп објеката који имају заједничку структуру и понашање. Класа представља нови, комплексни тип података. Заправо, сви типови података са којима смо до сада радили су класе. Дакле, Integer, Boolean, Float, Double, String и тако даље представљају посебне класе које у себи имају дефинисане одређене особине и методе, операције које се над њима могу обављати. Имена класа пишу се великим почетним словом. Сада ћемо на једном примеру објаснити како се креира нова класа у C#.

Пример 1. **Направити класу Ljubimac која ће имати две класне променљиве - ime и vrsta. Унутар класе креирати две методе - методу predstavljanje, која ће саопштавати која животиња је у питању и њено име, и методу oglasavanje. Потом, написати програм у коме ће се креирати објекти класе и позивати методе.**

Решење:

Креираћемо нов пројекат под именом Zivotinje.

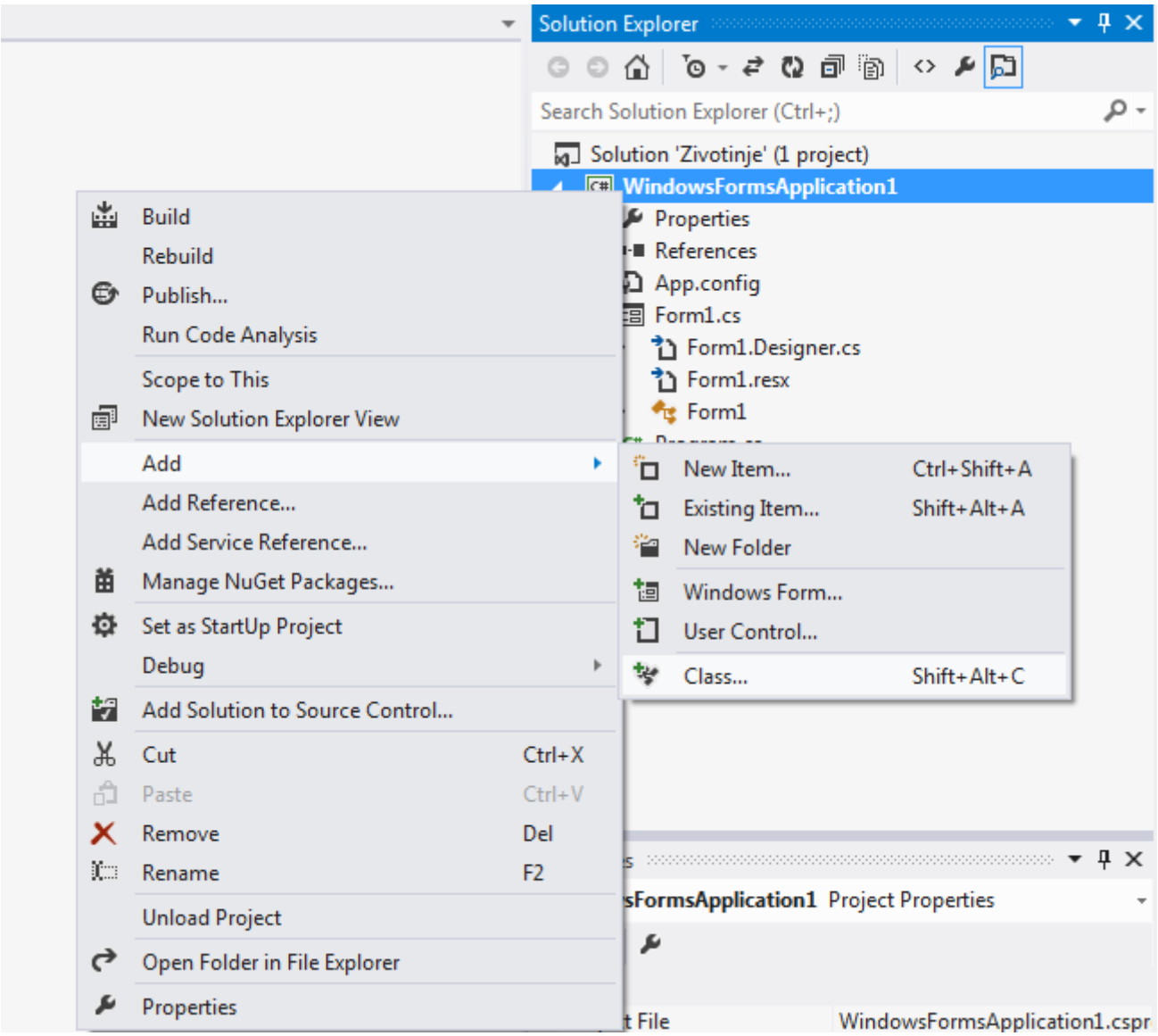
Правимо форму облика:



Слика 9.1. Изглед форме пре преименовања у примеру 1

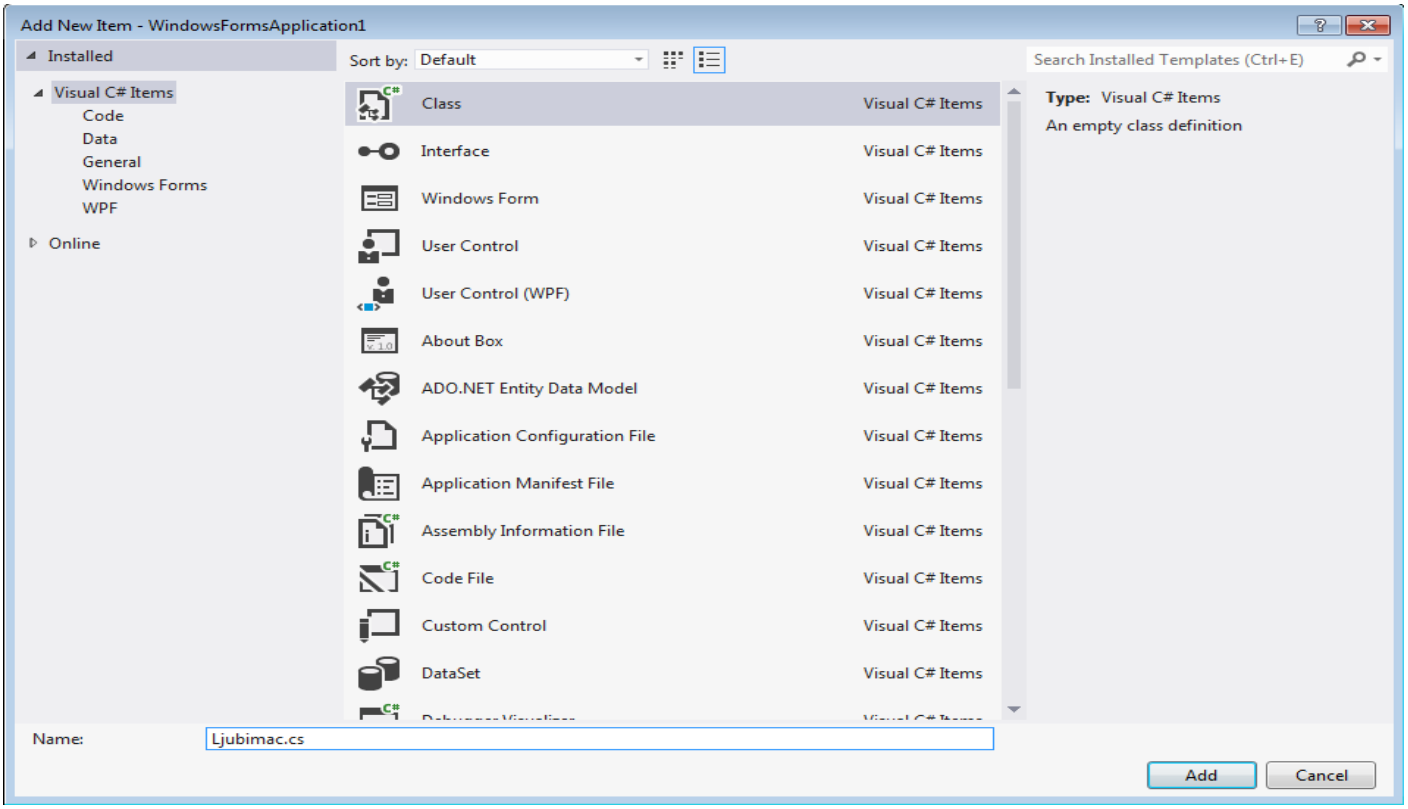
1. label1 - преименоваћемо у Zivotinja
2. label2 - преименоваћемо у ime
3. comboBox1 - овде ћемо додати неколико домаћих животиња
4. textBox1 - овде ћемо уписати име љубимца
5. button1 - преименујемо дугме у Predstavi se
6. label3 - овде ће се исписати врста и име љубимца
7. label4 - овде ће се исписати на који начин се љубимац оглашава

Сада ћемо креирати класу **Ljubimac**. Она мора бити креирана унутар пројекта на коме радимо. То ћемо постићи на следећи начин. Унутар радног окружења, у горњем десном углу налази се Solution Explorer. Десним кликом на име пројекта на коме радимо отвара нам се мени са опцијама. Бирамо опцију Add Class као на наредној слици.



Слика 9.2. Додавање класе у пројекат на коме радимо

Сада дајемо име класи и након тога, притиском на дугме Add креира се класа и отвара нам се прозор где пишемо код за класу



Слика 9.3. Креирање нове класе

Сада ћемо навести код за класу и методе класе.

```
class Ljubimac
{
    //Atributi klase Ljubimac
    public String ime;
    public String vrsta;
    //Konstruktor. Moramo napraviti ljubimca da bismo na njega mogli primenjivati
metode
    public Ljubimac(string ime, string vrsta)
    {
        this.ime = ime;
        this.vrsta = vrsta;
    }
    //Ovo je metod klase Ljubimac. Kao argument uzima vrstu i ime ljubimca.
    public string predstavljanje(string ime, string vrsta)
    {
        //Metod vraća vrstu i ime ljubimca
        return "Ja sam " + vrsta + " i zovem se " + ime;
    }

    //Metod klase Ljubimac. Kao argument uzima vrstu ljubimca.
    public string oglasavanje(string vrsta)
    {
        //U zavisnosti koju smo vrstu ljubimca odabrali on će se oglašavati na
različit način
        if (vrsta.Equals("Macka"))
```

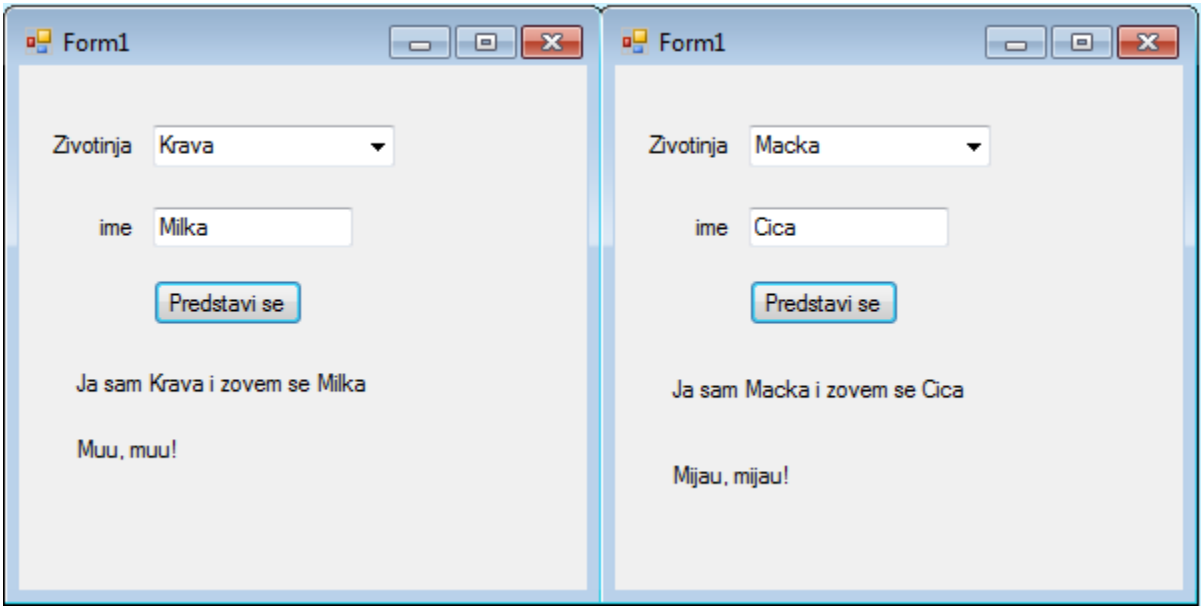
```
            return "Mijau, mijau!";
        else if (vrsta.Equals("Pas"))
            return "Av,av!";
        else if (vrsta.Equals("Kokoska"))
            return "Ko, ko!";
        else if (vrsta.Equals("Krava"))
            return "Muu, muu!";
        else if (vrsta.Equals("Ovca"))
            return "Bee, bee!";
        else if (vrsta.Equals("Patka"))
            return "Kva, kva!";
        else
            return "Niste izabrali ljubimca!";
    }
}
```

Креирали смо класу Ljubimac са методама predstavljanje и oglasavanje. Сада ћемо написати програм који ће се извршавати кликом на дугме. Вратимо се форми и двокликом на дугме Predstavi se отвара нам се прозор где пишемо одговарајући код.

```
public partial class Form1 : Form
{
    public Form1()
    {
        InitializeComponent();
    }

    private void button1_Click(object sender, EventArgs e)
    {
        //Deklaracija promenljivih
        string ime, vrsta;
        //Učitavamo ime iz textBox - a
        ime = textBox1.Text;
        //Učitavamo vrstu iz comboBox - a, biramo vrstu iz padajućeg menija
        vrsta = comboBox1.SelectedItem.ToString();
        //Kreiramo ljubimca pomoću konstruktora iz klase
        Ljubimac lj = new Ljubimac(ime, vrsta);
        //Na kreiranog ljubimca primenjujemo metode iz klase
        label3.Text = lj.predstavljanje(ime, vrsta);
        label4.Text = lj.oglasavanje(vrsta);

    }

    private void Form1_Load(object sender, EventArgs e)
    {
        //Postavljamo početnu vrednost comboBox - a na prvu stavku
        comboBox1.SelectedIndex = 0;
```

```
        }
}
```

Покренимо програм да видимо резултат.



Слика 9.4. Резултати рада програма у примеру 1

Ми ћемо се овде највише бавити прављењем нових метода које ће бити примењиване на већ постојеће класе како бисмо себи олакшали рад са њима. Методе које ћемо правити зваћемо функцијама.