Kolekcije

Pojam kolekcije

- Kolekcije se upotrebljavaju za upravljenje grupama objekata i imaju više funkcionalnosti nego nizovi
- Veličina niza se mora unapred definisati dok to nije slučaj sa kolekcijama
- Klase kolekcija nalaze se u prostoru imena System.Collections
- Klase kolekcija su definisane na bazi jasno definisanih interfejsa
- Negeneričke kolekcije rade sa tipom podataka object

Primeri negeneričkih kolekcija

- Klase
- ArrayList
- Queue
- Stack
- Hashtable
- Neophodno kastovanje članova kolekcije u njihov stvaran tip
- Ove kolekcije mogu miksovati različite tipove podataka
- Nisu type-safe

Kolekcija ArrayList

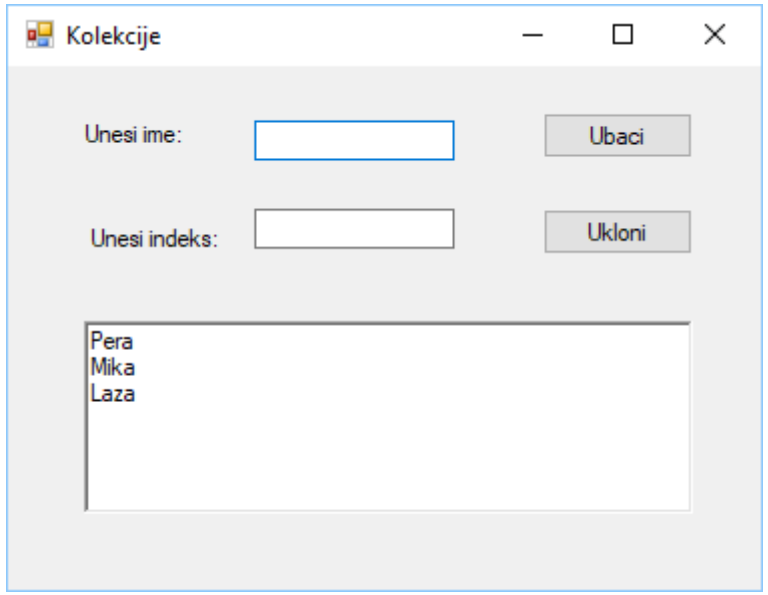
- Elementima liste pristupa se preko indeksa kao i kod niza
- Za razliku od niza nije neophodno unapred poznavati broj elemenata
- Metoda ***Add(object)*** dodaje objekat na kraj liste
- Metoda ***Clear()*** briše sve elemente iz liste
- Metoda ***Insert(pozicija, vrednost)*** ubacuje objekat ***vrednost*** na pozicuju ***pozicija***
- Metoda ***RemoveAt(index)*** briše elemenat sa indeksom index iz liste

```
public class ArrayList : IList, ICollection, IEnumerable, ICloneable
```

Kolekcija ArrayList

- Metoda ***Remove(object)*** uklanja prvo pojavljivanje specifičnog objekta iz kolekcije
- Svojstvo **Count** daje broj članova kolekcije
- Metoda **Contains**(object) određuje da li se određeni element nalazi u kolekciji
- Metoda **Sort**() sortira elemente kolekcije
- Metoda **Reverse**() prikazuje emente kolekcijeu inverznom redosledu
- Metoda **ToArray**() kopira elemente liste u jednodimenzionalan niz

Primer upotrebe kolekcija



Definisanje i štampanje kolekcije

```
public partial class Form1 : Form
{
    private ArrayList listaImena = new ArrayList();
    public void Stampaj(ArrayList al)
    {
            richTextBox1.Clear();
            foreach (string clan in al)
             {
                 richTextBox1.AppendText(clan + "\n");
             }
    }
```

Load procedura forme

```
private void Form1_Load(object sender, EventArgs e)
{
    listaImena.Add("Pera");
    listaImena.Add("Mika");
    listaImena.Add("Laza");
    Stampaj(listaImena);
}
```

Dodavanje člana u kolekciju

```
private void button1_Click(object sender, EventArgs e)
{
    string ime = textBox1.Text.Trim();
    if (ime.Length > 1)
    {
        string b0 = ime.Substring(0, 1).ToUpper();
        string b1 = ime.Substring(1).ToLower();
        ime = b0 + b1;

        if (!listaImena.Contains(ime))
        {
            listaImena.Add(ime);
            Stampaj(listaImena);
        }
        else
        {
            MessageBox.Show("Clan se vec nalazi u kolekciji");
        }
    }
    else
    {
        MessageBox.Show("Ime mora sadrzati bar 2 slova");
    }
    textBox1.Clear();
    textBox1.Focus();
}
```

Uklanjanje člana iz kolekcije

```
private void button2_Click(object sender, EventArgs e)
{
    int indeks;
    if (!int.TryParse(textBox2.Text, out indeks))
    {
        MessageBox.Show("Unesite ceo broj");
        return;
    }

    if (indeks >= 0 && indeks < listaImena.Count)
    {
        listaImena.RemoveAt(indeks);
        Stampaj(listaImena);
    }
    else
    {
        MessageBox.Show("Indeks van opsega");
    }
}
```

Pitanje 1

Svi članovi kolekcije ArrayList su tipa:

a. int
b. object
c. string

Odgovor: b

Pitanje 2

Broj članova neke negeneričke kolekcije dobija se korišćenjem:

a. svojstva Count
b. svojstva Length
c. metode Count()
d. metode Number()

Odgovor: a

Pitanje 3

Sve članove ArrayList kolekcije moguće je prikazati foreach petljom:

a. Da
b. Ne

Odgovor: a

Pitanje 4

Uklanjanje člana ArrayList kolekcije sa određene pozicije vrši se korišćenjem metode:

a. Remove()
b. RemoveAt()
c. DefeteFrom()

Odgovor: b

Generičke liste

Generičke liste

List<T>

```
using System.Collections.Generic;
```

```
private void button1_Click(object sender, EventArgs e)
{
    List<int> celobrojnaLista = new List<int>();
    celobrojnaLista.Add(1);
    celobrojnaLista.Add(2);
    celobrojnaLista.Add(55);

    for (int i = 0; i < celobrojnaLista.Count; i++)
    {
        richTextBox1.AppendText(celobrojnaLista[i] + "\n");
    }

    richTextBox1.AppendText("Novi nacin stampanja\n");

    foreach (int i in celobrojnaLista)
    {
        richTextBox1.AppendText(i + "\n");
    }

}
```

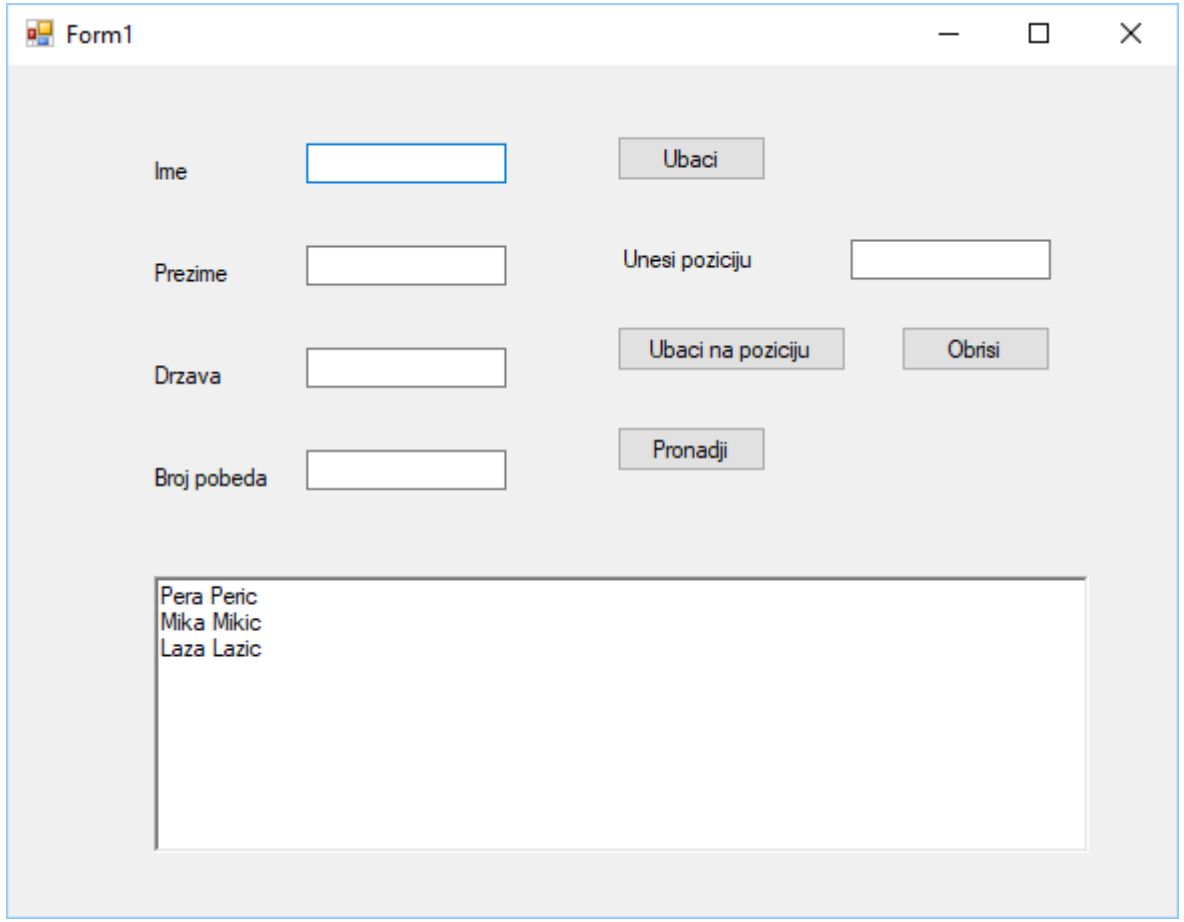
Klasa Trkac

```
public class Trkac
    {
        public string Ime { get; set; }
        public string Prezime { get; set; }
        public string Drzava { get; set; }
        public int BrojPobeda { get; set; }

        public override string ToString()
        {
            return Ime + " " + Prezime + "\n";
        }

    }
```

GUI aplikacije



Štampanje liste

```
private List<Trkac> lista= new List<Trkac>();
```

```
private void StampajListu(List<Trkac> trkaci)
{
    richTextBox1.Clear();
    foreach (Trkac t in trkaci)
    {
        richTextBox1.AppendText(t.ToString());
    }
}
```

Load događaj forme

```
private void Form1_Load(object sender, EventArgs e)
{
    Trkac t1 = new Trkac { Ime = "Pera", Prezime = "Peric", BrojPobeda = 12, Drzava = "Srbija" };
    lista.Add(t1);

    Trkac t2 = new Trkac { Ime = "Mika", Prezime = "Mikic", BrojPobeda = 10, Drzava = "Srbija" };
    lista.Add(t2);

    Trkac t3 = new Trkac { Ime = "Laza", Prezime = "Lazic", BrojPobeda = 8, Drzava = "Srbija" };

    lista.Add(t3);

    StampajListu(lista);
}
```

Resetovanje korisničkog interfejsa

```
private void Resetuj()
{
    textBoxIme.Clear();
    textBoxPrezime.Clear();
    textBoxDrzava.Clear();
    textBoxBrojPobeda.Clear();
}
```

Metoda za validaciju

```
public bool Validacija()
{
    if (textBoxIme.Text.Trim().Length < 2)
    {
        MessageBox.Show("Ime mora imati najmanje 2 karaktera");
        textBoxIme.Focus();
        return false;
    }
    if (textBoxPrezime.Text.Trim().Length < 2)
    {
        MessageBox.Show("Prezime mora imati najmanje 2 karaktera");
        textBoxPrezime.Focus();
        return false;
    }
    if (textBoxDrzava.Text.Trim().Length < 2)
    {
        MessageBox.Show("Drzava mora imati najmanje 2 karaktera");
        textBoxDrzava.Focus();
        return false;
    }

    if (!int.TryParse(textBoxBrojPobeda.Text.Trim(), out int _ ))
    {
        MessageBox.Show("Broj pobeda mora biti ceo broj");
        textBoxBrojPobeda.Clear();
        textBoxBrojPobeda.Focus();
        return false;
    }
    return true;
}
```

String metoda

```
public string VelikoPrvoSlovo(string rec)
{
    rec = rec.Trim().ToLower();
    if (rec.Length > 1)
    {
        return rec.Substring(0, 1).ToUpper() + rec.Substring(1);
    }
    return string.Empty;
}
```

Metoda KreirajTrkaca()

```
public Trkac KreirajTrkaca()
{
    if (Validacija())
    {
        string ime = VelikoPrvoSlovo(textBoxIme.Text);
        string prezime = VelikoPrvoSlovo(textBoxPrezime.Text);
        string drzava = VelikoPrvoSlovo(textBoxDrzava.Text);
        int brojPobeda = int.Parse(textBoxBrojPobeda.Text.Trim());

        Trkac t = new Trkac
        {
            Ime =ime, Prezime=prezime, Drzava=drzava, BrojPobeda = brojPobeda
        };
        return t;
    }
    else
    {
        return null;
    }
}
```

Ubacivanje u Listu-1

```
private void button1_Click(object sender, EventArgs e)
{
    Trkac t = KreirajTrkaca();
    if (t != null)
    {
        lista.Add(t);
        StampajListu(lista);
        Resetuj();
    }
}
```

Definisanje jednakosti dva objekta

```
class Trkac : IEquatable<Trkac>
{
    ....
 public bool Equals(Trkac other)
 {
    if (Ime.ToLower() == other.Ime.ToLower()
        &&
        Prezime.ToLower() == other.Prezime.ToLower()
        )
    {
        return true;
    }
    return false;
 }
}
```

Ubacivanje u Listu-2

```
private void button1_Click(object sender, EventArgs e)
{
    Trkac t = KreirajTrkaca();
    if (t != null)
    {
        if (lista.Contains(t))
        {
            MessageBox.Show("Trkac se vec nalazi u listi", "Poruka");
        }
        else
        {
            lista.Add(t);
            StampajListu(lista);
        }
        Resetuj();
    }
}
```

Ubacivanje elementa na poziciju

```
private void button2_Click(object sender, EventArgs e)
{
    Trkac t = KreirajTrkaca();
    if (t != null)
    {

        if (!int.TryParse(textBoxPozicija.Text, out int pozicija))
        {
            MessageBox.Show("Pozicija u listi mor biti ceo broj", "Poruka");
            return;
        }
        if (pozicija >= 0 && pozicija <= lista.Count)
        {
            lista.Insert(pozicija, t);
            StampajListu(lista);
        }
        else
        {
            MessageBox.Show("Prekoracili ste poziciju", "Poruka");
        }
        Resetuj();
        textBoxPozicija.Clear();
    }

}
```

Uklanjanje elementa iz liste

```
private void button3_Click(object sender, EventArgs e)
{
    int pozicija;

    if (!int.TryParse(textBoxPozicija.Text, out pozicija))
    {
        MessageBox.Show("Pozicija u listi mora biti ceo broj", "Poruka");
        textBoxPozicija.Focus();
        return;
    }
    if (pozicija >= 0 && pozicija < lista.Count)
    {
        lista.RemoveAt(pozicija);
        StampajListu(lista);
    }
    else
    {
        MessageBox.Show("Ne postoji clan liste", "Poruka");
    }
}
```

Pristup elementima generičke liste

```
private void button4_Click(object sender, EventArgs e)
{
    if (!int.TryParse(textBoxPozicija.Text, out int pozicija))
    {
        MessageBox.Show("Pozicija u listi mora biti ceo broj", "Poruka");
        textBoxPozicija.Focus();
        return;
    }
    if (pozicija > -1 && pozicija < listaTrkaca.Count)
    {
    Trkac t = listaTrkaca[pozicija];
    textBoxIme.Text = t.Ime;
    textBoxPrezime.Text = t.Prezime;
    textBoxDrzava.Text = t.Drzava;
    textBoxBrojPobeda.Text = t.BrojPobeda.ToString();
    }
    else
    {
        MessageBox.Show("Prekoracili ste poslednju poziciju", "Poruka");
    }
}
```

Pitanje 1

Generički celobrojna lista pod nazivom **intLista** se instancira na sledeći način:

a. int<List> intLista = new int<List>();
b. List<int> intLista = new List<int>();
c. List[int] intLista = new List[int];

Odgovor: b

Pitanje 2

Da li je unutar generičke liste, koja nije tipa object, dozvoljeno čuvanje podataka različitog tipa:

a. Da
b. Ne

Odgovor: b

Pitanje 3

Drugom članu generičke liste tipa string pod nazivom **stringLista** pristupa se korišćenjem sledeće linije koda:

a. string a= stringLista(1);
b. string a= stringLista[1];
c. string a= stringLista.IndexOf(1);

Odgovor: b

Generički rečnici

Dictionary

- Dictionary je kolekcija koja pridružuje ključ (key) nekoj vrednosti (value)
- Dictionary<TKey, TValue>
- Pronalaženje vrednosti korišćenjem njemu pridruženog ključa je veoma brzo
- Elementi rečnika su članovi KeyValuePair<TKey,Tvalue> strukture

Svojstva i metode kolekcije Dictionary<Tkey, Tvalue>

Method or property	Purpose
Count	Public property that gets the number of elements in the Dictionary.
Item( )	The indexer for the Dictionary.
Keys	Public property that gets a collection containing the keys in the Dictionary. (See also Values, later in this table.)
Values	Public property that gets a collection containing the values in the Dictionary. (See also Keys, earlier in this table.)
Add( )	Adds an entry with a specified Key and Value.
Clear( )	Removes all objects from the Dictionary.
ContainsKey( )	Determines whether the Dictionary has a specified key.
ContainsValue( )	Determines whether the Dictionary has a specified value.
GetEnumerator( )	Returns an enumerator for the Dictionary.
Remove( )	Removes the entry with the specified Key.

Štampanje sadržaja rečnika

```
public void Stampaj<Tkey,Tvalue>(Dictionary<Tkey, Tvalue> dict)
{
    richTextBox1.Clear();

    foreach (KeyValuePair<Tkey,Tvalue> red in dict)
    {
        string s = string.Format("Kljuc: {0}, Vrednost: {1}\n",red.Key, red.Value);

        richTextBox1.AppendText(s);
    }
}
```

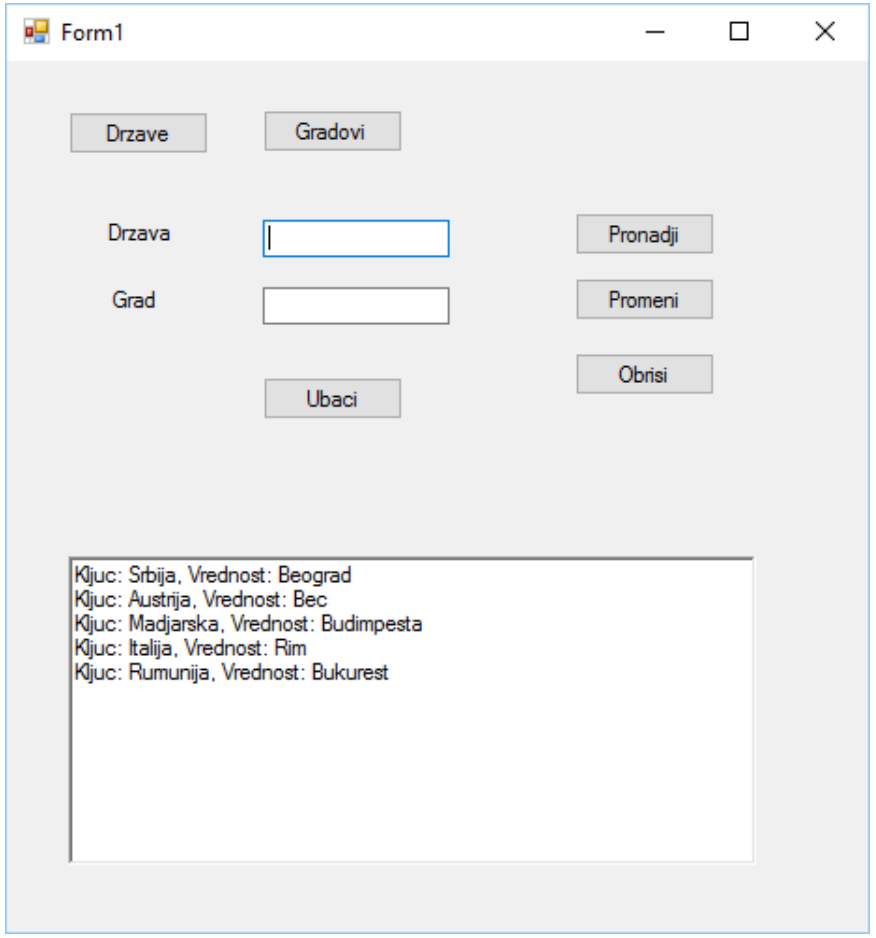
Upotreba kolekcije Dictionary-2

```
private Dictionary<string, string> recnik = new Dictionary<string, string>();
```

```
private void Form1_Load(object sender, EventArgs e)
{
    recnik.Add("Srbija", "Beograd");
    recnik.Add("Austrija", "Bec");
    recnik.Add("Madjarska", "Budimpesta");
    recnik.Add("Italija", "Rim");
    recnik.Add("Rumunija", "Bukurest");

    Stampaj(recnik);
}
```

Korisnički interfejs



Pristup ključevima rečnika

```
private void button1_Click(object sender, EventArgs e)
{
    richTextBox1.Clear();

    foreach (string s in recnik.Keys)
    {
        richTextBox1.AppendText(s + "\n");
    }
}
```

Pristup vrednostima rečnika

```
private void button2_Click(object sender, EventArgs e)
{
    richTextBox1.Clear();

    foreach (string s in recnik.Values)
    {
        richTextBox1.AppendText(s + "\n");
    }
}
```

String funkcija

```
public string VelikoPrvoSlovo(string rec)
{
    rec = rec.Trim().ToLower();
    if (rec.Length > 1)
    {
        return rec.Substring(0, 1).ToUpper() + rec.Substring(1);
    }
    return string.Empty;
}
```

Unos reda u rečnik

```
private void buttonUbaci_Click(object sender, EventArgs e)
{
    string drzava = VelikoPrvoSlovo( textBoxDrzava.Text);
    string grad = VelikoPrvoSlovo(textBoxGrad.Text);

    if (!recnik.ContainsKey(drzava))
    {
        recnik.Add(drzava, grad);
        Stampaj(recnik);
    }
    else
    {
        MessageBox.Show("Drzava se vec nalazi u recniku");
    }

    textBoxDrzava.Clear();
    textBoxGrad.Clear();
    textBoxDrzava.Focus();
}
```

Pristup vrednosti na osnovu ključa

```
private void buttonPronadji_Click(object sender, EventArgs e)
{
    string drzava = VelikoPrvoSlovo(textBoxDrzava.Text);

    if (recnik.ContainsKey(drzava))
    {
        textBoxGrad.Text = recnik[drzava];
    }
    else
    {
        MessageBox.Show("Drzava se ne nalazi u recniku");
        textBoxDrzava.Clear();
        textBoxDrzava.Focus();
    }
}
```

Brisanje člana rečnika

```
private void buttonObrisi_Click(object sender, EventArgs e)
{
    string drzava = VelikoPrvoSlovo( textBoxDrzava.Text.Trim());

    if (recnik.ContainsKey(drzava))
    {
        recnik.Remove(drzava);
        Stampaj(recnik);
    }
}
```

Pitanje 1

Red generičkog rečnika Dictionary<int, string> je :

a. vrednost tipa KeyValuePair<int,string>
b. vrednost tipa string
c. vrednost tipa int

Odgovor: a

Pitanje 2

Za brisanje člana sa ključem **k** iz rečnika **dict** koristi se naredba:

a. dict.Delete(k);
b. dict[k].Delete();
c. dict.Remove(k);

Odgovor: c

Pitanje 3

Ako je definisan sledeći kod:

```
Dictionary<int, string> dict = new Dictionary<int, string>();
int a = 0;
string b = "";
if (dict.ContainsKey(a))
    {
        //???
    }
```

koja linija koda se može napisati u bloku if

a. b = dict[a];
b. a = dict[b];
c. a= dict(a);
d. b=dict(a);

Odgovor: a