4.3. PRIMERI LISTE (*LISTBOX*) , PADAJUĆE LISTE (*COMBOBOX*)

Objekat lista (***ListBox***) obezbeđuje prikazivanje stavki u obliku liste (spisak predmeta, lista proizvoda, adresar, ...). Ovaj objekat može sadržati samo tekstualne članove. Jedna od najbitnijih osobina je ta da se mogu selektovati (označiti) jedna (svojstvo ***SelectionMode*** postavljeno je na ***One***) ili više stavki iz liste (svojstvo ***SelectionMode*** postavljeno je na ***MultiSimple*** ili ***MultiExtended***).

Sadržaj liste se postavlja posredstvom svojstva ***Items***. Redni broj odabrane stavke je svojstvo ***ItemIndex*** i ono je tipa ***Integer***, pri čemu se prvi red u listi indeksira sa ***0***.

Lista može prikazivati stavke u više kolona, pri čemu se taj broj definiše postavljanjem vrednosti u svojstvu ***Columns***.

Lista se definiše tako što se iz palete alata izabere ikonica . Kada je lista dodata u formu, podešavanja svojstava (***properties***) se izvode na potpuno isti način kao i sa objektima pomenutim u prethodnom poglavlju.

Za razliku od liste, padajuća lista, kombinuje mogućnost unosa teksta (***Edit***) i izbora jedne od više ponuđenih stavki u listi: Bitna karakteristika za ovaj objekat je ta da se ne mogu se izabrati više od jedne ponuđene stavke. Ovaj objekat se sastoji od polja za unos, koje na desnoj strani ima dugme sa strelicom na dole, koji omogućava, da se klikom na njega, otvori spisak ponuđenih stavki čiji oblik može podesiti zadavanjem vrednosti za svojstvo ***DropDownStyle***.

Postoje tri moguće vrednosti svojstva ***DropDownStyle*** padajuće liste, koje se definišu podešavanjem:

- ***Simple***: ne vide se stavke liste, već mora da se unose vrednosti u polje za unos, da bi se pretražile stavke liste.
- ***Drop – Down***: kombinacija unosa teksta za pretraživanje stavke liste ili klika na dugme za otvaranje sadržaja liste.
- ***Drop – Down – List***: onemogućen unos teksta već se izbor stavke vrši klikom na dugme za otvaranje sadržaja liste.

U narednim primerima, prikazaće se postupak formiranja forme sa listama i padajućom listom.

4.3.1. PRIMER FORME SA LISTAMA

Za ovaj primer potrebno je napraviti win aplikaciju, koja će imati jednu listu, dva ***Textbox*** – a i dva komandna dugmeta, slika 22. Korisnik unosi neki podatak u ***Textbox*** (1), klikom na komandno dugme (2) taj podatak se prenosi u listu (3) i prazni se ***Textbox*** (1). Prilikom prenosa stavke iz *Textbox* – a (1) proveriti da li stavka, koja se prenosi, postoji u listi (3) i ako postoji daje se poruka: ***″Stavka je već u LISTI!!!″***, odnosno proverava se da mora da postoji neka vrednost u ***Textbox*** – u , ako je ***Textbox*** prazan daje se poruka "***Mora se uneti bar neki podatak!***". Kada korisnik izabere stavku u listi (3) ta stavka se prikazuje u ***Textbox*** – u (4). Kada korisnik klikne na komandno dugme (5), izabrana stavka se uklanja iz liste, a ako ništa nije izabrano ili je lista prazna generiše se i prikazuje poruka o tome i to:

- ako nije izabran red u listi, poruka je ″Da bi se uklonila stavka iz liste mora se prvo nešto u listi izabrati!!!″.
- ako je lista prazna , poruka je ″Lista je PRAZNA!!!″.

Svojstva forme i pripadajućih objekata sa slike 22, dati su u tabeli T – 6.

Slika 22. Forma za rad sa objektom lista – ***ListBox***

Tabela T – 6. Podešena svojstva fome i njenih objekata za primer sa slike 22

Svojstvo	**Vrednost**
	Forma (0)
(Name)	**PrimerListBox**
AutoScaleMode	**None**
BackColor	**ActiveCaption** (paleta boja ***System***)
Font	**Tahoma, Regular, 14pt**
ForeColor	**DarkBlue** (paleta boja ***Web***)
Size	**690,490**
StartPosition	**CenterScreen**
Text	**Primer rada sa ListBox – om**

	Labela (L1)
Location	**12, 25**
Text	**Uneti stavku**

	***Textbox* (1)**
(Name)	**ZaUnos**
Location	**12, 51**
Size	**613,30**

	Labela (L2)
Location	**12, 143**
Text	**Unete stavke**

Svojstvo	**Vrednost**
	Lista (3)
(Name)	**Lista**
Location	**12, 172**
Size	**608, 165**

	Labela (L3)
Location	**12, 347**
Text	**Izabrana stavka**

	***Textbox* (4)**
(Name)	**ZaPrikaz**
Location	**12, 377**
Size	**613,30**

	Button (2)
(Name)	**DodatiUListu**
Location	**12, 96**
Size	**612, 38**
Text	**Prebaciti u ListBox**

	Button(5)
(Name)	**Ukloni**
Location	**626, 172**
Size	**43, 165**
Text	**–>**

Prvi kod koji se, u ovom poglavlju objašnjava, je programiranje događaja ***click***, dugmeta (2) – ***DodatiUListu***, slika 22., i dat je u PK – 6. Otvaranje prozora za pisanje koda se ostvaruje, (1) ako se klikne 2x levim tasterom miša na dugme ***DodatiUListu*** ili se izabere prozor ***Events*** i klikne dva puta na događaj ***Click***.

PK – 6. Programski kod za događaj ***Click*** objekta ***button: DodatiUListu***

```
1   private void DodatiUListu_Click(object sender, EventArgs e)
2   {
3            if (string.IsNullOrWhiteSpace(ZaUnos.Text))
4               MessageBox.Show("Mora se uneti bar neki podatak!");
5            else
6            {
7               string stavka = ZaUnos.Text.Trim();
8               if (Lista.Items.Contains(stavka))
9                  MessageBox.Show("Stavka je već u LISTI!!!");
10              else
11              {
12                 Lista.Items.Add(stavka);
13                 ZaUnos.Text = string.Empty;
14              }
15           }
16  }
```

Procedura PK – 6. je. u stvari, razgranata struktura koja prvoverava da li u *textbox* – u, ***ZaUnos***, postoji uneti tekst. Funkcija koja ispituje da li je unet tekst, odnosno da li je string prazan – ***Null*** ili su unete praznine ***WhiteSpace***, je ***`string.IsNullOrWhiteSpace(String_koji_se_Proverava)`***, linija 3. U liniji 4, je prikazana sintaksa prikazivanje poruke, ukoliko je uslov iz linije 3 ispunjen, u suprotnom, se izvršava blok naredbi linije 7 – 14.

U liniji 7., je prikazan način dodeljivanja tipa i vrednosti promenljivoj *stavka*. Kao što može da se primeti, prvo se definiše tip promenljive (***`string`***), zatim naziv (***`stavka`***) i posle toga se promenljivoj dodeljuje vrednost koja odgovara definisanom tipu. Ovde treba voditi računa, jer vrednost koja se dodeljuje mora biti istog tipa kao što je definisan tip promenljive, u suprotnom javiće se greška suntakse. U liniji 7., je iskorišćena funkcija ***Trim***(), koja briše praznine iz stringa. Njena sintaksa je *String_Promenljiva.**Trim***(). U ovom slučaju *String_Promenljiva* je `ZaUnos.Text`.

U liniji 8., proverava se da li, u listi – ***Lista***, postoji vrednost promenljive *stavka*. Funkcija koja se za to koristi je ***`Lista.Items.Contains(stavka)`***. Ukoliko je stavka uneta ova funkcija vraća ***True*** i prikazuje se poruka da je stavka već uneta, u suprotnom se stavka dodaje u listi – ***Lista*** funkcijom ***Add***, linija 12. Sintaksa ove funkcije `Lista.Items.Add(stavka)`, a opšti oblik je: *ListBox.Items.Add*(*string_promenljiva*). "Pražnjenje" *textbox* – a se izvodi komandom `string.Empty`, linija 13.

Preuzimanje vrednosti iz liste, a se izvodi pomoću događaja ***SelectedIndexChange***, PK – 7. U liniji 3., se ispituje da li je selektovan red u listi. Naime, ukoliko je selektovan red svojstvo ***`SelectedIndex`*** vraća vrednost od 0 do ukupnog broja stavki u listi (***`Lista.Items.Count`***). Ukoliko ništa nije selektovano ***SelectedIndex*** vraća ***–1***. Ukoliko je selektovan red, vrednosti svojstva Text, u *textbox* – u: ***ZaPrikaz***, se dodeljuje selektovani red funkcijom ***Lista.SelectedItem.ToString()***, linija 4.

PK – 7. Programski kod za događaj ***SelectedIndexChange*** objekta ***ListBox: Lista***

```
1   private void Lista_SelectedIndexChanged(object sender, EventArgs e)
2   {
3            if (Lista.SelectedIndex != -1)
4               ZaPrikaz.Text = Lista.SelectedItem.ToString();
5   }
```

Za ovaj primer, ostaje još da se isprogramira dugme ***Ukloni***, PK – 8. U ovoj proceduri se prvo ispituje da li je lista prazna broj stavki u listi, ***`Lista.Items.Count`*** – linija 3, mora da bude različit od 0. Zatim, se proverava da li je izabran red u listi. Ako je izabran ***`Lista.SelectedIndex`*** – linija 5, vraća vrednost između 0 i ukupnog broja stavki u listi (***`Lista.Items.Count`***), u suprotnom vrednost koja se vraća je –1. Vrednost izabranog reda se učitava u promenljivu komandom: ***`Lista.SelectedItem.ToString()`*** – linija 9. Uklanjanje izabranog reda iz liste se izvodi komandom ***`Lista.Items.Remove(stavka)`*** – linija 10.

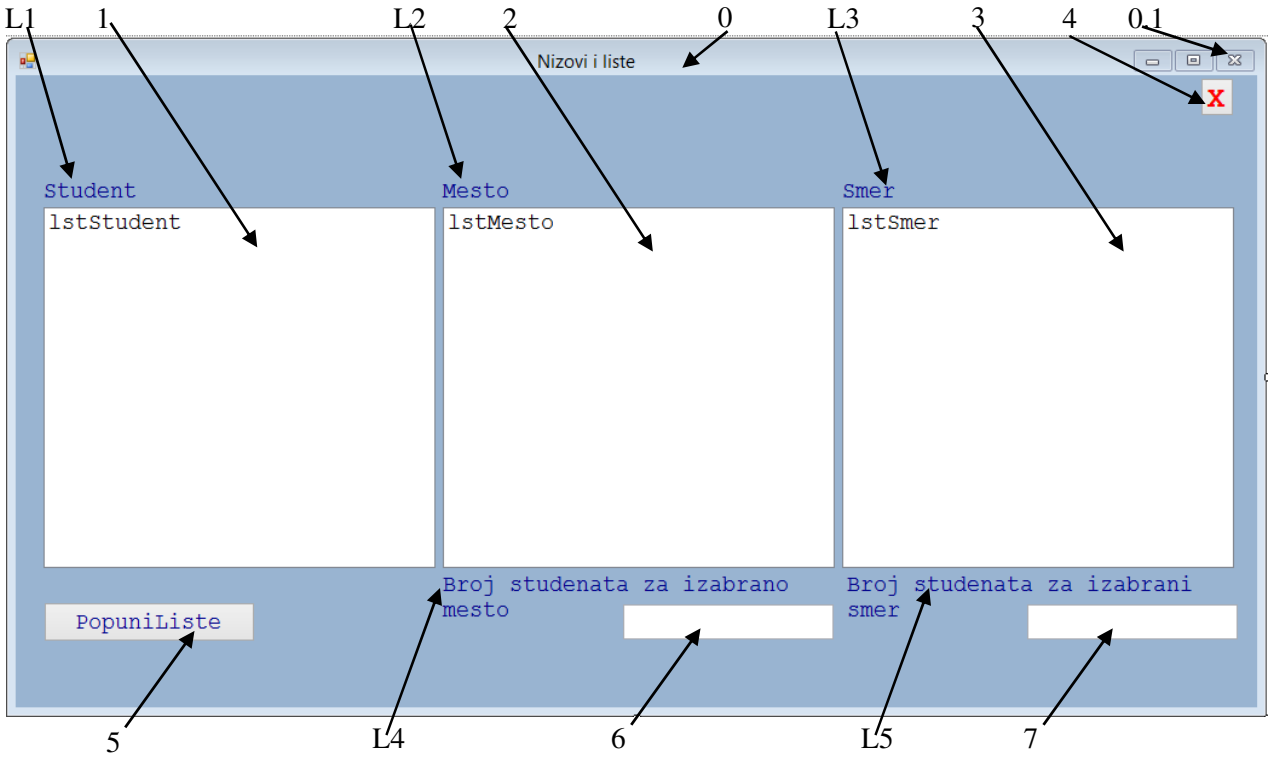
PK – 8. Programski kod za događaj ***Click*** objekta ***button: Ukloni***

```
1   private void Ukloni_Click(object sender, EventArgs e)
2   {
3       if (Lista.Items.Count != 0)
4       {
5            if (Lista.SelectedIndex == -1)
6                MessageBox.Show("Da bi se uklonila stavka iz liste mora se prvo nešto u
                                  listi izabrati!!!");
7            else
8            {
9               string stavka = Lista.SelectedItem.ToString();
10              Lista.Items.Remove(stavka);
11              ZaPrikaz.Text = string.Empty;
12           }
13      }
14      else
15           MessageBox.Show("Lista je PRAZNA!!!");
16  }
```

4.3.2. PRIMER NIZ I LISTE

U ovom poglavlju, cilj je da se, steknu znanja za rad sa nizovima, punjenjem liste sa vrednostima elemenata niza, rad u ciklusima, kao i programski menjanje svojstava liste. U tu svrhu, daje se sledeći primer sa definisanim projektnim zahtevima.

Napraviti Win formu, slika 23., koja će sadržati tri liste: ***lstStudent***, ***lstMesto*** i ***lstSmer***; pet labela sa tekstom: "*Student*", "*Mesto*", "*Smer*", "*Broj studenata za izabrano mesto*" i "*Broj studenata za izabrani smer*"; dva komandna dugmeta: za ubacivanje stavki u liste (***PopuniListe***) i za zatvaranje forme (***Izlaz***), kao i dva *textbox* a: za prikaz broja studenata koji dolaze iz izabranog mesta u listi ***lstMesto*** (***BrojStudMesto***) i za prikaz broja studenata koji su upisali izabrani smer u listi ***lstSmer*** (***BrojStudSmer***). Vrednosti pojedinih svojstava forme i objekata u formi dati su u tabeli T–7.



Slika 23. Traženi izgled forme

Zahtevi koji treba da se ispune su sledeći:

1. ***Close*** (**0.1**), slika 23., u liniji naslova – standardne forme da bude onemogućeno;
2. Liste se popunjavaju iz statički definisanog trodimenzionalnog niza koji treba da bude dostupan svim funkcijama i procedurama u projektu, klikom na dugme ***PopuniListe***;
3. Zatvaranje forme – prekid rada sa programom se ostvaruje klikom na dugme ***Izlaz*** ili pritiskom tatsera ***Esc***;
4. Kada se selektuje student iz liste ***lstStudent*** potrebno je da se automatski selektuje mesto odakle je student, u listi ***lstMesto*** i koji smer je upisao u ***lstSmer***;
5. Kada se u listi ***lstMesto*** selektuje mesto, potrebno je da se u ***lstStudent*** selektuju svi studenti iz tog mesta, a da se listi ***lstSmer*** deselektuje i da se prikaže u *textbox* u ***BrojStudMesto*** prikaže broj studenata koji dolaze iz selektovanog mesta;
6. Kada se u listi ***lstSmer*** selektuje smer, potrebno je da se u ***lstStudent*** selektuju svi studenti koji su na tom smeru, a da se listi ***lstMesto*** deselektuje i da se prikaže u *textbox* u ***BrojStudSmer*** prikaže broj studenata koji su upisali taj smer.

Tabela T – 7. *Podešena svojstva fome i njenih objekata za primer sa slike 23*

Svojstvo	**Vrednost**
	Forma (0)
(Name)	**NizLista**
AutoScaleMode	**None**
BackColor	**ActiveCaption** (paleta boja ***System***)
Font	**Courier New, Regular, 14pt**
ForeColor	**DarkBlue** (paleta boja ***Web***)
Size	**1042,561**
StartPosition	**CenterScreen**
Text	**Nizovi i liste**

	Labela (L1)
Location	**19,85**
Text	**Student**

	Labela (L2)
Location	**349, 85**
Text	**Mesto**

	Labela (L3)
Location	**679, 85**
Text	**Smer**

	Labela (L4)
Location	**349, 410**
Size	**328, 58**
Text	**Broj studenata za izabrano mesto**

	Labela (L5)
Location	**683, 410**
Size	**328, 58**
Text	**Broj studenata za izabrani smer**

	Listbox (1)
(Name)	**lstStudent**
Location	**23, 109**
Size	**324, 298**

Svojstvo	**Vrednost**
	Listbox (2)
(Name)	**lstMesto**
Location	**353, 109**
Size	**324, 298**

	Listbox (3)
(Name)	**lstSmer**
Location	**683, 109**
Size	**324, 298**

	Button(4)
(Name)	**Izlaz**
Font	**Courier New, Bold, 18pt**
ForeColor	**Red**
Location	**979, 2**
Size	**28, 32**
Text	**X**

	Button(5)
(Name)	**PopuniListe**
Location	**23, 435**
Size	**324, 33**
Text	**Popuni Liste**

	***Textbox*(6)**
(Name)	**BrojStudMesto**
Location	**502, 437**
Size	**174, 29**

	***Textbox*(7)**
(Name)	**BrojStudSmer**
Location	**836, 437**
Size	**174, 29**

U ***C#***, nije predviđena mogućnost da se dugmad, u title bar – u, za minimiziranje, uvećavanje i zatvaranje prozora, posebno "isključuju". Zato će se prvi projektni zahtev, iz ovog primera rešiti programski. Rešenje je dato u PK – 9.

PK – 9. Programski kod za isključivanje dugmeta ***Close*** u ***title bar*** – u .

```
1  protected override CreateParams CreateParams
2  {
3         get
4         {
5                CreateParams param = base.CreateParams;
6                param.ClassStyle |= 0x200;
7                return param;
8         }
9  }
```

Iz koda PK – 9., vidi se da je formirana klasa ***param***, linija 4., na bazi standardne klase ***CreateParams***. U ovoj klasi su definisani osnovni stilovi izgleda formi i unosom odgovarajućeg koda, linija 6., može se izvršiti podešavanje forme. U liniji 6., je ukucan kod koji onemogućava isključivanje forme klikom na ***Close*** u ***task bar*** – u.

Drugi zahtev je bio da se liste popunjavaju klikom na dugme ***PopuniListe*** (5) i to pomoću niza, čiji elementi moraju da budu dostupni svakoj proceduri, odnosno funkciji u projektu. Da bi se ovaj zahtev ostvario, potrebno je taj niz definisati u glavnoj klasi (`public partial class NizLista : Form`) – PK 10., a najbolje mesto za to je odmah ispod finkcije:

```
public NizLista()
{
       InitializeComponent();
}
```

PK – 10. Programski kod za definisanje trodimenzionalnog niza ***studenti*** .

```
1  string[,] studenti = new string[,]
2  {
3         {"001/2019 Marko Markovic","Trstenik","IT"},
4         {"031/2019 Janko Jankovic","Krusevac","DS"},
5         {"017/2019 Marina Marinovic","Krusevac","IT"},
6         {"025/2019 Jovana Jovic","Kraljevo","MAS"},
7         {"035/2019 Stanko Stankic","Trstenik","DS"},
8         {"074/2019 Perica Peric","Vrnjacka Banja","DS"},
9         {"019/2019 Evgenije Trubic","Vrnjacka Banja","IT"},
10 };
```

U liniji 1., programskog koda PK – 10., definisan je niz pod nazivom *studenti* tipa podataka ***string*** i to sa više kolona. Način definisanja prikazan u ovom kodu omogućava da se unese proizvoljan broj redova i proizvoljan broj kolona – dinamička definicija niza. Drugim rečima, u linijama od 3 do 9 unose se 7 redova sa tri kolone, koje predstavljaju podatke o studentima (student – indeks kolone **0**; mesto – indeks kolone **1** i smer – indeks kolone **2**). Redovi su definisani sadržajem između vitičastih zagrada, a odvojeni su zarezom(**,**):

`{"001/2019 Marko Markovic","Trstenik","IT"},`

dok su kolone predstavljene podacima između znaka navoda (") i odvojene su, takođe, zarezom (**,**):

`"001/2019 Marko Markovic","Trstenik","IT"`

Naravno, kao u programskim jezicima C/C++, indeksi reda, odnosno kolone uvek počinju od 0 i pristupa se elementu niza preko indeksa reda i kolone, ***studenti[i, j]***, gde je ***i*** – indeks reda, a ***j*** – indeks kolone. Recimo, podatak o broju indeksa i imenu i prezimenu studenta u ***i*** – tom redu, bi se dobio pozivanjem vrednosti niza ***i*** – tog reda i ***0*** – te kolone: ***studenti[i,0]***.

Dimenzije niza se mogu dobiti pomoću funkcije `studenti.GetLength(param)`. Da bi se dobio ukupan broj **redova** višedimenzionalnog niza vrednost za ***param*** je 0, a za ukupan broj **kolona** vrednost je 1.

Pošto je definisan niz studenti, ostaje još da se programira dugme (5) – ***PopuniListe***. Ideja je da se prvo isprazne sve tri liste – što se postiže pozivanjem funkcije ***...Items.Clear()*** (linije od 3 do 4 u PK – 11.), a zatim se kroz ciklus, linije od 6 – 4 u PK – 11., dodaju stavke u liste, pri čemu se stavke u listu ***lstStudent*** dodaju bez provere da li se stavka nalazi u listi, a za liste ***lstMesto*** i ***lstSmer*** mora da se proveri da li je vrednost, iz niza, koja se unosi u te liste već u listi i ako nije da se doda. Provera da li je neka stavka već u listi ostvaruje se pozivanjem funkcije ***...Items.Contains(stavka_koja_se_proverava)***, linije od 12 do 17 i od 18 do 23. Kod za ispunjenje ovog projektnog zahteva je dat u PK – 11.

PK – 11. Programski kod za definisanje trodimenzionalnog niza ***studenti*** .

```
1  private void PopuniListe_Click(object sender, EventArgs e)
2  {
3         lstStudent.Items.Clear();
4         lstMesto.Items.Clear();
5         lstSmer.Items.Clear();
6         for (int i = 0; i < studenti.GetLength(0); i++)
7         {
8             string ws = studenti[i, 0];
9             string wm = studenti[i, 1];
10            string wsm = studenti[i, 2];
11            lstStudent.Items.Add(ws);
12            if (lstMesto.Items.Contains(wm))
13            { }
14            else
15            {
16                lstMesto.Items.Add(wm);
17            }
18            if (lstSmer.Items.Contains(wsm))
19            { }
20            else
21            {
22                lstMesto.Items.Add(wsm);
23            }
24        }
25 };
```

U linijima od 8 do 10 je prikazano kako se u promenljive učitavaju vrednosti iz niza ***studenti***.

Zatvaranje forme, što je treći zahtev ovog projekta, je već objašnjen u prethodnim primerima, tako da se ovde daju samo kodovi, PK – 12. i PK – 13., za njegovu realizaciju.

PK – 12. Programski kod za zatvaranje forme kada se pritisne taster ***Esc*** – ***KeyDown***

```
1  private void NizLista_KeyDown(object sender, KeyEventArgs e)
2  {
3      if (e.KeyCode == Keys.Escape)
4          Application.Exit();
5  }
```

PK – 13. Programski kod za zatvaranje forme kada se pritisne dugme (4) – ***Izlaz***

```
1  private void Izlaz_Click(object sender, KeyEventArgs e)
2  {
3      Application.Exit();
4  }
```

Sledeći zahtev se odnosio na to, da kada se izabere student iz liste ***lstStudent***, da se u listi ***lstMesto*** automatski selektuje mesto, odakle je student – kolona 1 iz niza ***studenti***; odnosno u listi ***lstSmer***, smer koji je student upisao – kolona 2 iz niza ***studenti***,. Za ispunjenje ovog zahteva mora da se koriste dva događaja vezano za liste i to ***Click*** i ***SelectedIndexChanged***.

Pošto su 4, 5 i 6 zahtev protivurečni, u smislu da lista ***lstStudent***, u početku, ne sme da dopusti selektovanje više od jednog reda – svojstvo ***SelectionMode*** mora da ima vrednost ***One***, dok kasnije kada se selektuju mesto odnosno smer, to svojstvo mora da ime vrednost ***MultiSimple*** – što omogućava selektovanje više redova iz liste. Ovaj zahtev se uspešno rešava uvođenjem još jedne promenljive (u ovom primeru ***klik***) koja proverava koja lista je dobila fokus i u zavisnosti od vrednosti te promenljive definiše se vrednost svojstva ***SelectionMode***. Pošto ta promenljiva, ***klik***, mora da bude vidljiva za sve procedure i funkcije u projektu i ona se definiše na isti način kao što je definisan niz, samo što je ona skalar i za potrebe ovog projekta ona je definisana iznad definicije niza ***studenti*** komadom:

```
int klik = -1;
```

Za zahtev 4 kodovi za događaje ***Click*** i ***SelectedIndexChanged*** su dati u PK – 13 i PK – 14.

PK – 14. Programski kod definisanje vrednosti promenljive ***klik*** i svojstva ***SelectionMode*** kada lista ***lstStudent*** dobije fokus

```
1  private void lstStudent_Click(object sender, EventArgs e)
2  {
3      klik = 0;
4      lstStudent.SelectionMode = System.Windows.Forms.SelectionMode.One;
5  }
```

U PK – 14., promenljiva ***klik*** dobija vrednost 0, linija 3., i svojstvo ***SelectionMode*** dobija vrednost ***One***, linija 4. Za definisanje vrednostu ***SelectionMode*** koristi se sistemska klasa ....***Windows.Forms.SelectionMode.One***.

PK – 15. Programski kod za događaj ***SelectedIndexChanged*** liste ***lstStudent***.

```
1  private void lstStudent_SelectedIndexChanged(object sender, EventArgs e)
2  {
3      if (lstStudent.SelectedIndex != -1 && klik==0)
4      {
5        int redi = lstStudent.SelectedIndex;
6        string mesto = studenti[redi, 1].Trim();
7        string smer = studenti[redi, 2].Trim();
8        lstMesto.SelectedIndex = -1;
9        lstSmer.SelectedIndex = -1;
10       for (int i = 0; i < lstMesto.Items.Count; i++)
11       {
12           string mmesto = lstMesto.Items[i].ToString().Trim();
13           if (mmesto == mesto)
14              lstMesto.SelectedItem = mmesto;
15       }
16       for (int i = 0; i < lstSmer.Items.Count; i++)
17       {
18           if (lstSmer.Items[i].ToString() == smer)
19             lstSmer.SelectedItem = smer;
20       }
21     }
22 }
```

U liniji 3., PK – 15., proverava se da li je uopšte selektovano nešto u listi i da li je vrednost promenljive ***klik*** postavljena na 0. Ukoliko je ispunjen taj uslov u promenljivu, ***redi***, učitava se broj selektovanog reda (***lstStudent.SelectedIndex***), linija 5., a zatim se iz niza studenti u promenljive mesto i smer, učitavaju vrednosti iz

niza studenti i to za mesto iz kolone sa indeksom 1, a za smer sa indeksom 2, linije 6 – 7. Zatim se vrše poništavanje selekcije, u listama ***lstMesto*** i ***lstSmer***, postavljanjem vrednosti svojstva ***SelectedIndex*** na –1, linije 8 – 9. Na kraju se u ciklusu, u ovom primeru ***for***, prolazi kroz liste ***lstMesto***, od linije 10 do linije 15, a za listu ***lstSmer***, od linije 16 do linije 20. Broj stavki u listi se dobija svojstvom ***Count***, linija 10 (***lstMesto.Items.Count***) , odnosno 16 (***lstSmer.Items.Count***). Dodeljivanje vrednosti promenljivoj iz izabranog reda liste prikazano je u liniji 12 (***string*** *mmesto* ***= lstMesto.Items[i].ToString().Trim();***), a poređenje vrednosti izabranog reda sa nekom promenljivom, u liniji 18 (***lstSmer.Items[i].ToString()*** ***==*** *smer*). Programsko selektovanje reda u listi moguće je uraditi na dva načina. U kodu PK – 15., prikazan je način gde se izabrani red izjednačava sa odgovarajućom promenljivom, linija 14 (***lstMesto.SelectedItem*** ***=*** *mmesto*;), odnosno linija 15 (***lstSmer.SelectedItem*** ***=*** *smer****;***). U kodovima PK – 17, odnosno PK – 18., to se izvodi preko indeksa reda.

Peti i šesti zahtev su realizovani na sličan način kao što je dato za zahtev četiri, tako da se u nastavku daju kodovi za događaje ***Click*** i ***SelectedIndexChanged***.

PK – 16. Programski kod definisanje vrednosti promenljive ***klik*** i svojstva ***SelectionMode*** kada lista ***lstMesto*** dobije fokus

```
1  private void lstMesto_Click(object sender, EventArgs e)
2  {
3      klik = -1;
4      lstStudent.SelectionMode = System.Windows.Forms.SelectionMode.MultiSimple;
5  }
```

Za kod PK – 16., je karakteristično to da se svojstvo ***SelectionMode*** listi ***lstStudent*** postavlja na MultiSimple, odnosno omogućava se selektovanje više redova u listi, linija 4.

PK – 17. Programski kod za događaj ***SelectedIndexChanged*** liste ***lstMesto***.

```
1  private void lstMesto_SelectedIndexChanged(object sender, EventArgs e)
2  {
3      if (lstMesto.SelectedIndex != -1 && klik==-1)
4      {
8        lstStudent.SelectedIndex = -1;
9        lstSmer.SelectedIndex = -1;
10       int ukbroj = 0;
11       int redi = lstMesto.SelectedIndex;
13       string mesto = lstMesto.Items[redi].ToString().Trim();
14       for (int i = 0; i < studenti.GetLength(0); i++)
15       {
16           if (mesto == studenti[i, 1].Trim())
17           {
18               for (int j = 0; j < lstStudent.Items.Count; j++)
19                  if (j == i)
20                  {
21                     lstStudent.SetSelected(j, true);
22                     ukbroj += 1;
23                  }
24       }
25       BrojStudMesto.Text = ukbroj.ToString();
26     }
27 }
```

Da bi se izračunao ukupan broj studenata koji dolaze iz izabranog mesta u listi ***lstMesto***, uvodi se celobrojna (***int***) promenljiva ***ukbroj*** i inicijalizuje se sa vrednošću 0, linija 10. U promenljivu, ***redi***, smešta se indeks selektovanog reda liste ***lstMesto***, linija 11. i učitava se vrednost tog reda u promenljivu ***mesto***, linija 12. U ciklusu ***for***, od 14 do 24., prolazi se kroz niz ***studenti***, upoređuje se vrednost promenljive mesto sa vrednošću druge kolone niza studenti, linija 16., i ako su vrednosti iste prolazi se kroz celu listu ***lstStudent***, linije 18 do 23., upoređuju se indeksi ***i*** i ***j***, linija

19. i ukoliko se poklapaju selektuje se red u listi ***lstStudent***, linija 21. i uvećava se promenljiva ***ukbroj*** za jedan, linija 22. Na kraju, određen broj studenata za izabrano mesto se prikazuje u *textbox* – u, ***BrojStudMesto***, u liniji 25.

Na potpuno isti način je rešen i zahtev 6, tako da se u nastavku daju samo kodovi u PK – 18. i PK – 19.

PK – 18. Programski kod definisanje vrednosti promenljive ***klik*** i svojstva ***SelectionMode*** kada lista ***lstSmer*** dobije fokus

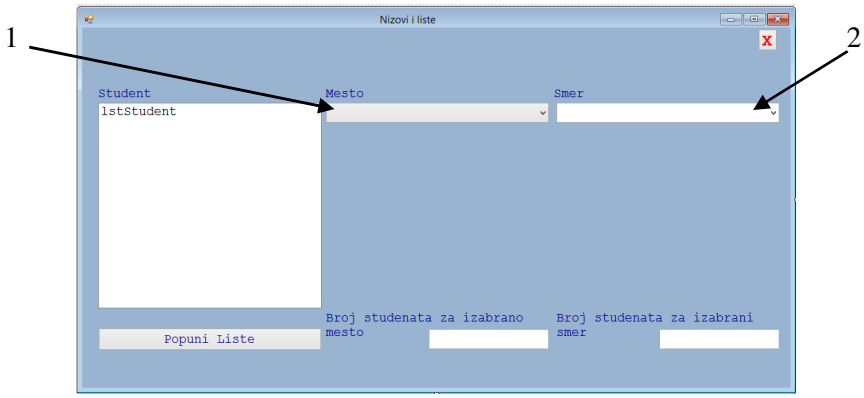
```
1  private void lstSmer_Click(object sender, EventArgs e)
2  {
3      klik = -1;
4      lstStudent.SelectionMode = System.Windows.Forms.SelectionMode.MultiSimple;
5  }
```

PK – 19. Programski kod za događaj ***SelectedIndexChanged*** liste ***lstSmer***.

```
1  private void lstSmer_SelectedIndexChanged(object sender, EventArgs e)
2  {
3      if (lstSmer.SelectedIndex != -1 && klik==-1)
4      {
8        lstStudent.SelectedIndex = -1;
9        lstSmer.SelectedIndex = -1;
10       int ukbroj = 0;
11       int redi = lstSmer.SelectedIndex;
13       string smer = lstSmer.Items[redi].ToString().Trim();
14       for (int i = 0; i < studenti.GetLength(0); i++)
15       {
16           if (smer == studenti[i, 2].Trim())
17           {
18               for (int j = 0; j < lstStudent.Items.Count; j++)
19                  if (j == i)
20                  {
21                     lstStudent.SetSelected(j, true);
22                     ukbroj += 1;
23                  }
24       }
25       BrojStudSmer.Text = ukbroj.ToString();
26     }
27 }
```

4.3.3. PRIMER NIZ I PADAJUĆA LISTA

Rad sa padajućom listom je u potpunosti isti kao i sa listama, slika 24., tako da se ovde daju, samo izmenjene procedure,u odnosu na prethodni primer.



Slika 24. Traženi izgled forme

U tabeli 8., data su samo svojstva padajućih listi za mesto (***cmbMesto***) i za smer (***cmbSmer***).

Tabela T – 8. *Podešena svojstva fome i njenih objekata za primer sa slike 24*

Svojstvo	**Vrednost**
	ComboBox (1)
(Name)	**cmbMesto**
AutoCompleteMode	**SuggestAppend**
AutoCompleteSource	**ListItems**
Location	**353, 109**
Size	**323, 29**

Svojstvo	**Vrednost**
	ComboBox (2)
(Name)	**cmbSmer**
AutoCompleteMode	**SuggestAppend**
AutoCompleteSource	**None**
Location	**687, 109**
Size	**323, 29**

PK – 20. Programski kod za događaj ***SelectedIndexChanged*** liste ***lstStudent*** – izmenjena procedura PK – 15.

```
1   private void lstStudent_SelectedIndexChanged(object sender, EventArgs e)
2   {
3       if (lstStudent.SelectedIndex != -1 && klik==0)
4       {
5         int redi = cmbStudent.SelectedIndex;
6         string mesto = studenti[redi, 1].Trim();
7         string smer = studenti[redi, 2].Trim();
8         cmbMesto.SelectedIndex = -1;
9         cmbSmer.SelectedIndex = -1;
10        for (int i = 0; i < cmbMesto.Items.Count; i++)
11        {
12            string mmesto = cmbMesto.Items[i].ToString().Trim();
13            if (mmesto == mesto)
14               cmbMesto.SelectedItem = mmesto;
15        }
16        for (int i = 0; i < cmbSmer.Items.Count; i++)
17        {
18            if (cmbSmer.Items[i].ToString() == smer)
19              cmbSmer.SelectedItem = smer;
20        }
21      }
22  }
```

PK – 21. Programski kod definisanje vrednosti promenljive ***klik*** i svojstva ***SelectionMode*** kada lista ***cmbMesto*** dobije fokus – izmenjena procedura PK – 16.

```
1   private void cmbMesto_Click(object sender, EventArgs e)
2   {
3       klik = -1;
4       lstStudent.SelectionMode = System.Windows.Forms.SelectionMode.MultiSimple;
5   }
```

PK – 22. Programski kod za događaj ***SelectedIndexChanged*** liste ***cmbMesto*** – izmenjena procedura PK – 17.

```
1   private void lstMesto_SelectedIndexChanged(object sender, EventArgs e)
2   {
3       if (cmbMesto.SelectedIndex != -1 && klik==-1)
4       {
8         lstStudent.SelectedIndex = -1;
9         cmbSmer.SelectedIndex = -1;
10        int ukbroj = 0;
11        int redi = cmbMesto.SelectedIndex;
13        string mesto = cmbMesto.Items[redi].ToString().Trim();
14        for (int i = 0; i < studenti.GetLength(0); i++)
```

```
15          {
16              if (mesto == studenti[i, 1].Trim())
17              {
18                  for (int j = 0; j < lstStudent.Items.Count; j++)
19                     if (j == i)
20                     {
21                        lstStudent.SetSelected(j, true);
22                        ukbroj += 1;
23                     }
24          }
25          BrojStudMesto.Text = ukbroj.ToString();
26      }
27  }
```

PK – 23. Programski kod definisanje vrednosti promenljive ***klik*** i svojstva ***SelectionMode*** kada lista ***lstSmer*** dobije fokus – izmenjena procedura PK – 18.

```
1   private void cmbSmer_Click(object sender, EventArgs e)
2   {
3       klik = -1;
4       lstStudent.SelectionMode = System.Windows.Forms.SelectionMode.MultiSimple;
5   }
```

PK – 24. Programski kod za događaj ***SelectedIndexChanged*** liste ***lstSmer*** – izmenjena procedura PK – 19.

```
1   private void cmbSmer_SelectedIndexChanged(object sender, EventArgs e)
2   {
3       if (cmbSmer.SelectedIndex != -1 && klik==-1)
4       {
8          lstStudent.SelectedIndex = -1;
9          cmbSmer.SelectedIndex = -1;
10         int ukbroj = 0;
11         int redi = cmbSmer.SelectedIndex;
13         string smer = cmbSmer.Items[redi].ToString().Trim();
14         for (int i = 0; i < studenti.GetLength(0); i++)
15         {
16             if (smer == studenti[i, 2].Trim())
17             {
18                 for (int j = 0; j < lstStudent.Items.Count; j++)
19                    if (j == i)
20                    {
21                       lstStudent.SetSelected(j, true);
22                       ukbroj += 1;
23                    }
24         }
25         BrojStudSmer.Text = ukbroj.ToString();
26      }
27  }
```