Kladionica

Enum Kontinent

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Kladionica
{
    public enum Kontinent
    {
        EU,
        AZ,
        AF,
        SA,
        JA,
        AU
    }
}
```

Klasa Zemlja

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Kladionica
{
    public class Zemlja
    {
        private string naziv;

        public string Naziv
        {
            get { return naziv; }
            set { naziv = value; }
        }

        private double kvota;

        public double Kvota
        {
            get { return kvota; }
            set { kvota = value; }
        }
        private Kontinent kontinent;

        public Zemlja(string naziv, double kvota, Kontinent kontinent)
        {
            this.naziv = naziv;
            this.kvota = kvota;
            this.kontinent = kontinent;
        }
    }
```

```
}
```

Klasa Osoba

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Kladionica
{
    public class Osoba
    {
        protected string ime;
        protected string prezime;
        private DateTime datumRodjenja;

        public int GodinaRodjenja
        {
            get { return datumRodjenja.Year; }
        }

        public Osoba(string ime, string prezime, DateTime datumRodjenja)
        {
            this.ime = ime;
            this.prezime = prezime;
            this.datumRodjenja = datumRodjenja;
        }
        public virtual string DajPodatke()
        {
            return string.Format("{0} {1} {2}", ime, prezime,
datumRodjenja.ToShortDateString());
        }
    }
}
```

Klasa Igrač

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Kladionica
{
    public class Igrac : Osoba
    {
        private double iznosUplate;

        public double IznosUplate
        {
            get { return iznosUplate; }
            set { iznosUplate = value; }
        }

        private Zemlja odabranaZemlja;
```

```
        public Zemlja OdabranaZemlja
        {
            get { return odabranaZemlja; }
            set { odabranaZemlja = value; }
        }

        public Igrac(string ime, string prezime, DateTime datumRodjenja, double
iznosUplate, Zemlja odabranaZemlja) : base(ime, prezime, datumRodjenja)
        {
            this.iznosUplate = iznosUplate;
            this.odabranaZemlja = odabranaZemlja;
        }

        public override string DajPodatke()
        {
            return string.Format("{0} {1} {2}", odabranaZemlja.Naziv,
iznosUplate*odabranaZemlja.Kvota, base.DajPodatke());
        }
    }
}
```

Klasa Uplatno Mesto

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Kladionica
{
    public class UplatnoMesto
    {
        private string adresa;
        private string grad;
        public List<Igrac> spisakIgraca;

        public List<Igrac> SpisakIgraca
        {
            get { return spisakIgraca; }
            set { spisakIgraca = value; }
        }

        public UplatnoMesto(string adresa, string grad)
        {
            this.adresa = adresa;
            this.grad = grad;
            spisakIgraca = new List<Igrac>();
        }

        public string DajPodatke()
        {
            string ulicaBezBroja = adresa.Replace("0", "");
            ulicaBezBroja.Replace("1", "");
            ulicaBezBroja.Replace("2", "");
            ulicaBezBroja.Replace("3", "");
            ulicaBezBroja.Replace("4", "");
            ulicaBezBroja.Replace("5", "");
```

```
            ulicaBezBroja.Replace("6", "");
            ulicaBezBroja.Replace("7", "");
            ulicaBezBroja.Replace("8", "");
            ulicaBezBroja.Replace("9", "");

            string gradBezSamoglasnika = grad.Replace("a", "");
            gradBezSamoglasnika.Replace("e", "");
            gradBezSamoglasnika.Replace("i", "");
            gradBezSamoglasnika.Replace("o", "");
            gradBezSamoglasnika.Replace("u", "");

            double ukupnaUplata = 0;
            for (int i = 0; i < spisakIgraca.Count; i++)
            {
                ukupnaUplata += spisakIgraca[i].IznosUplate;
            }

            return string.Format("{0}, {1}, Ukupna uplata: {2}RSD", ulicaBezBroja,
gradBezSamoglasnika, ukupnaUplata.ToString());
        }

        public void DodajIgraca(Igrac noviIgrac)
        {
            if ((DateTime.Now.Year - noviIgrac.GodinaRodjenja >= 18) &&
(noviIgrac.IznosUplate > 50 && noviIgrac.IznosUplate < 100000))
            {
                bool pronasaoIgraca = false;
                int indeksIgraca = 0;

                for (int i = 0; i < spisakIgraca.Count; i++)
                {
                    if (noviIgrac.DajPodatke() == spisakIgraca[i].DajPodatke())
                    {
                        if (noviIgrac.OdabranaZemlja == spisakIgraca[i].OdabranaZemlja)
                        {
                            pronasaoIgraca = true;
                            indeksIgraca = i;
                        }
                    }
                }

                if (pronasaoIgraca)
                {
                    spisakIgraca[indeksIgraca] = noviIgrac;
                }
                else
                {
                    spisakIgraca.Add(noviIgrac);
                }
            }
        }

        public Igrac UcitajIgraca()
        {
            Console.Write("Molimo Vas da unesete ime igraca: ");
            string ime = Console.ReadLine();

            Console.Write("Molimo Vas da unesete prezime igraca: ");
```

```
            string prezime = Console.ReadLine();

            Console.Write("Molimo Vas da unesete datum rodjenja: ");
            string godinaRodjenja = Console.ReadLine();
            string[] nizPodataka = godinaRodjenja.Split('/');

            bool uneoPogresno = true;
            double iznosUplate = 0;

            while (uneoPogresno)
            {
                Console.Write("Molimo Vas da unesete iznos uplate: ");
                string iznos = Console.ReadLine();

                try
                {
                    iznosUplate = Double.Parse(iznos);
                    uneoPogresno = false;
                }
                catch (FormatException)
                {
                    Console.WriteLine("Pogresan unos iznosa uplate. Molimo Vas da unesete
cifru!");
                }
            }

            Console.Write("Molimo Vas da unesete naziv zemlje: ");
            string naziv = Console.ReadLine();

            Console.Write("Molimo Vas da unesete kvotu: ");
            string kvota = Console.ReadLine();

            Console.Write("Molimo Vas da unesete naziv kontinenta: ");
            string kontinent = Console.ReadLine();

            Kontinent odabraniKontinent;

            switch (kontinent)
            {
                case "EU":
                    odabraniKontinent = Kontinent.EU;
                    break;
                case "AZ":
                    odabraniKontinent = Kontinent.AZ;
                    break;
                case "AF":
                    odabraniKontinent = Kontinent.AF;
                    break;
                case "SA":
                    odabraniKontinent = Kontinent.SA;
                    break;
                case "JA":
                    odabraniKontinent = Kontinent.JA;
                    break;
                case "AU":
                    odabraniKontinent = Kontinent.AU;
                    break;
                default :
```

```
                    odabraniKontinent = Kontinent.EU;
                    break;
            }

            return new Igrac(ime, prezime, new DateTime(Int32.Parse(nizPodataka[2]),
Int32.Parse(nizPodataka[1]), Int32.Parse(nizPodataka[0])), iznosUplate, new Zemlja(naziv,
Double.Parse(kvota), odabraniKontinent));
        }
    }
}
```

Klasa Kladionica

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Kladionica;

namespace Kladionica1
{
    public class Kladionica
    {
        private string takmicenje;
        public List<UplatnoMesto> spisakUplatnihMesta;
        private int brojIgracaSaViseUplata;

        public Kladionica(string takmicenje)
        {
            this.takmicenje = takmicenje;
            spisakUplatnihMesta = new List<UplatnoMesto>();
            brojIgracaSaViseUplata = 0;
        }

        public string DajPodatke()
        {
            int ukupanBrojIgraca = 0;

            for (int i = 0; i < spisakUplatnihMesta.Count; i++)
            {
                ukupanBrojIgraca += spisakUplatnihMesta[i].SpisakIgraca.Count;
            }

            string ispis = string.Format("Kladionica: {0}, Ukupan broj igraca: {1}/n",
takmicenje, ukupanBrojIgraca);

            for (int i = 0; i < spisakUplatnihMesta.Count; i++)
            {
                ispis += string.Format("/t{0}/n", spisakUplatnihMesta[i].DajPodatke());

                for (int j = 0; j < spisakUplatnihMesta[i].SpisakIgraca.Count; j++)
                {
                    ispis += string.Format("/t/t{0}/n",
spisakUplatnihMesta[i].SpisakIgraca[j].DajPodatke());
                }
            }
            return ispis;
```

```
        }

        public UplatnoMesto UcitajUplatnoMesto()
        {
            Console.WriteLine("Molimo Vas da unesete adresu uplatnog mesta: ");
            string adresa = Console.ReadLine();

            Console.WriteLine("Molimo Vas da unesete grad u kome je uplatno mesto: ");
            string grad = Console.ReadLine();

            UplatnoMesto novoUplatnoMesto = new UplatnoMesto(adresa, grad);

            Console.Write("Molimo Vas unesite broj igraca koje zelite uneti: ");
            int brojIgraca = Int32.Parse(Console.ReadLine());

            for (int i = 0; i < brojIgraca; i++)
            {
                novoUplatnoMesto.DodajIgraca(novoUplatnoMesto.UcitajIgraca());
            }
            return novoUplatnoMesto;
        }

        static void Main(string[] args)
        {
            Console.WriteLine("Molimo Vas da unesete takmicenje: ");
            string takmicenje = Console.ReadLine();

            Kladionica kladionica = new Kladionica(takmicenje);

            kladionica.spisakUplatnihMesta.Add(kladionica.UcitajUplatnoMesto());
            kladionica.spisakUplatnihMesta.Add(kladionica.UcitajUplatnoMesto());

            Console.WriteLine(kladionica.DajPodatke());
            Console.WriteLine("Broj igraca sa vise uplata: {0}",
kladionica.brojIgracaSaViseUplata);
        }
    }
}
```

Festival

Klasa Žanr

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Festival
{
    public class Zanr
    {
        private string naziv;

        public Zanr(string naziv)
        {
            this.naziv = naziv;
        }

        public string DajPodatke()
        {
            return string.Format("{0}{1}", naziv[0], naziv[naziv.Length - 1]);
        }
    }
}
```

Klasa Film

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Festival
{
    public class Film
    {
        private string naziv;
        private Zanr zanr;

        public Zanr Zanr
        {
            get { return zanr; }
            set { zanr = value; }
        }
        private double duzina;

        public double Duzina
        {
            get { return duzina; }
            set { duzina = value; }
        }

        public Film(string naziv, Zanr zanr, double duzina)
        {
            this.naziv = naziv;
```

```
            this.zanr = zanr;
            this.duzina = duzina;
        }

        public string DajPodatke()
        {
            return string.Format("{0}, {1}min", naziv, duzina.ToString());
        }
    }
}
```

Klasa Dnevni Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Festival
{
    public class DnevniProgram
    {
        private DateTime datum;
        private List<Film> spisakFilmova;
        private int maksimalanBrojFilmova;

        public DnevniProgram(DateTime datum)
        {
            this.datum = datum;
            spisakFilmova = new List<Film>();
            maksimalanBrojFilmova = 0;
        }

        public string DajPodatke()
        {
            double ukupnaDuzina = 0;

            for (int i = 0; i < spisakFilmova.Count; i++)
            {
                ukupnaDuzina += spisakFilmova[i].Duzina;
            }

            List<Zanr> spisakZanrova = new List<Zanr>();

            foreach (Film film in spisakFilmova)
            {
                if (!spisakZanrova.Contains(film.Zanr))
                {
                    spisakZanrova.Add(film.Zanr);
                }
            }

            string ispis = string.Format("Datum: {0}, Ukupno trajanje: {1}/n", datum,
ukupnaDuzina);

            foreach (Zanr zanr in spisakZanrova)
            {
```

```
                ispis += string.Format("/t{0}/n", zanr.DajPodatke());

                foreach (Film film in spisakFilmova)
                {
                    if (film.Zanr == zanr)
                    {
                        ispis += string.Format("/t/t{0}/n", film.DajPodatke());
                    }
                }
            }

            return ispis;
        }

        public void DodajFilm(Film noviFilm)
        {
            if (spisakFilmova.Count < maksimalanBrojFilmova)
            {
                int brojFilmovaZanra = 0;

                foreach (Film film in spisakFilmova)
                {
                    if (film.Zanr == noviFilm.Zanr)
                    {
                        brojFilmovaZanra++;
                    }
                }

                if (brojFilmovaZanra <= 4)
                {
                    spisakFilmova.Add(noviFilm);
                }
            }
        }
    }
}
```

Klasa Festival

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Festival;

namespace Festival1
{
    class Festival
    {
        private string naziv;
        private List<DnevniProgram> programFestivala;
        private int ukupanBrojZanrova;

        public Festival(string naziv)
        {
            this.naziv = naziv;
            programFestivala = new List<DnevniProgram>();
```

```
            ukupanBrojZanrova = 0;
        }

        public Film UcitajFilm()
        {
            Console.Write("Molimo Vas da unesete naziv filma: ");
            string naziv = Console.ReadLine();

            Console.Write("Molimo Vas da unesete zanr filma: ");
            string zanr = Console.ReadLine();

            bool pogresanUnos = true;
            double duzina = 0;

            while (pogresanUnos)
            {
                Console.Write("Molimo Vas da unesete duzinu filma: ");
                string duzinaString = Console.ReadLine();
                try
                {
                    duzina = Double.Parse(duzinaString);
                    pogresanUnos = false;
                }
                catch (FormatException)
                {
                    Console.WriteLine("Pogresan unos duzine filma. Molimo Vas da unesete
cifru!");
                }
            }

            return new Film(naziv, new Zanr(zanr), duzina);
        }

        public DnevniProgram UcitajDnevniProgram()
        {
            Console.WriteLine("Molimo Vas da unesete datum: ");
            string datum = Console.ReadLine();
            string[] nizPodataka = datum.Split('/');

            DnevniProgram dnevniProgram = new DnevniProgram(new
DateTime(Int32.Parse(nizPodataka[2]), Int32.Parse(nizPodataka[1]),
Int32.Parse(nizPodataka[0])));

            bool pogresanUnos = true;
            int maksimalanBrojFilmova = 0;

            while (pogresanUnos)
            {
                Console.Write("Molimo Vas da unesete unesete maksimalan broj filmova: ");
                string maksimalniBroj = Console.ReadLine();

                try
                {
                    maksimalanBrojFilmova = Int32.Parse(maksimalniBroj);
                    pogresanUnos = false;
                }
                catch (FormatException)
                {
```

```
                    Console.WriteLine("Pogresan unos maksimalnog broja filmova. Molimo
Vas da unesete cifru!");
                }
            }

            for (int i = 0; i < maksimalanBrojFilmova; i++)
            {
                dnevniProgram.DodajFilm(UcitajFilm());
            }
            return dnevniProgram;
        }

        static void Main(string[] args)
        {
            Console.Write("Molimo Vas da unesete naziv festivala: ");
            string nazivFestivala = Console.ReadLine();

            Festival festival = new Festival(nazivFestivala);

            festival.programFestivala.Add(festival.UcitajDnevniProgram());
            festival.programFestivala.Add(festival.UcitajDnevniProgram());

            string ispis = string.Format("Naziv festivala: {0}/n", festival.naziv);

            foreach (DnevniProgram dnevniProgram in festival.programFestivala)
            {
                ispis += string.Format("/t{0}/n", dnevniProgram.DajPodatke());
            }

            ispis += string.Format("Ukupan broj zanrova: {0}/n",
festival.ukupanBrojZanrova);

            Console.WriteLine(ispis);
        }
    }
}
```

Aerodorom

Enum Klasa

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Aerodrom
{
    public enum Klasa
    {
        Ekonomska,
        Biznis
    }
}
```

Klasa Sedište

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Aerodrom
{
    public class Sediste
    {
        private int broj;
        private Klasa klasa;

        public int Broj
        {
            get { return broj; }
            set { broj = value; }
        }

        public Sediste(int broj, Klasa klasa)
        {
            this.broj = broj;
            this.klasa = klasa;
        }
    }
}
```

Klasa Osoba

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Aerodrom
{
    public class Osoba
    {
```

```
        protected string ime;

        public string Ime
        {
            get { return ime; }
        }
        protected string prezime;

        public string Prezime
        {
            get { return prezime; }
        }

        public Osoba(string ime, string prezime)
        {
            this.ime = ime;
            this.prezime = prezime;
        }

        public virtual string DajPodatke()
        {
            return string.Format("{0} {1}", ime, prezime);
        }
    }
}
```

Klasa Putnik

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Aerodrom
{
    public class Putnik : Osoba
    {
        private Sediste sediste;

        public Sediste Sediste
        {
            get { return sediste; }
        }

        public Putnik(string ime, string prezime, Sediste sediste) : base(ime, prezime)
        {
            this.sediste = sediste;
        }

        public override string DajPodatke()
        {
            return string.Format("{0} {1}", sediste.Broj, base.DajPodatke());
        }
    }
}
```

Klasa Let

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Aerodrom
{
    public class Let
    {
        private string relacija;
        private DateTime datum;
        private List<Putnik> spisakPutnika;

        public List<Putnik> SpisakPutnika
        {
            get { return spisakPutnika; }
            set { spisakPutnika = value; }
        }

        public Let(string relacija, DateTime datum)
        {
            this.relacija = relacija;
            this.datum = datum;
            spisakPutnika = new List<Putnik>();
        }

        public string DajPodatke()
        {
            string relacijaBezSamoglasnika = relacija.Replace("a", "");
            relacijaBezSamoglasnika.Replace("e", "");
            relacijaBezSamoglasnika.Replace("i", "");
            relacijaBezSamoglasnika.Replace("o", "");
            relacijaBezSamoglasnika.Replace("u", "");

            string[] relacijaPodeljena = relacijaBezSamoglasnika.Split('/');
            string odrediste = relacijaPodeljena[0];
            string destinacija = relacijaPodeljena[1];
            string odredisteSkraceno = string.Format("{0}{1}", odrediste[0].ToString(),
odrediste[odrediste.Length-1].ToString());
            string destinacijaSkraceno = string.Format("{0}{1}",
destinacija[0].ToString(), destinacija[destinacija.Length - 1].ToString());

            return string.Format("{0} {1}-{2}", datum.ToShortDateString(),
odredisteSkraceno, destinacijaSkraceno);
        }

        public void DodajPutnika(Putnik noviPutnik)
        {
            bool zauzetoSediste = false;

            for (int i = 0; i < spisakPutnika.Count; i++)
            {
                if (spisakPutnika[i].Sediste.Broj == noviPutnik.Sediste.Broj)
                {
                    zauzetoSediste = true;
```

```
                }
            }

            if (!zauzetoSediste && spisakPutnika.Count < 100)
            {
                bool istiPutnik = false;
                int redniBrojUSpisku = 0;

                for (int i = 0; i < spisakPutnika.Count; i++)
                {
                    if (spisakPutnika[i].Ime == noviPutnik.Ime &&
spisakPutnika[i].Prezime == noviPutnik.Prezime)
                    {
                        istiPutnik = true;
                        redniBrojUSpisku = i;
                    }
                }
                if (istiPutnik)
                {
                    spisakPutnika[redniBrojUSpisku] = noviPutnik;
                }
                else
                {
                    spisakPutnika.Add(noviPutnik);
                }
            }
        }
    }
}
```

Klasa Aerodrom

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Aerodrom;

namespace Aerodrom1
{
    public class Aerodrom
    {
        public string naziv;
        public List<Let> spisakLetova;
        public int brojPutnikaUBiznisKlasi;

        public Aerodrom(string naziv)
        {
            this.naziv = naziv;
            spisakLetova = new List<Let>();
            brojPutnikaUBiznisKlasi = 0;
        }

        public string DajPodatke()
        {
            int ukupanBrojPutnika = 0;
```

```
            for (int i = 0; i < spisakLetova.Count; i++)
            {
                ukupanBrojPutnika += spisakLetova[i].SpisakPutnika.Count;
                    }

            string ispis = string.Format("Naziv aerodroma: {0}, Ukupan broj putnika:
{1}/n", naziv, ukupanBrojPutnika);

            for (int i = 0; i < spisakLetova.Count; i++)
            {
                ispis += string.Format("/t{0}/n", spisakLetova[i].DajPodatke());
                for (int j = 0; j < spisakLetova[i].SpisakPutnika.Count; j++)
                {
                    if (spisakLetova[i].SpisakPutnika[j].Sediste.Klasa == Klasa.Biznis)
                    {
                        ispis += string.Format("/t/t{0}/n",
spisakLetova[i].SpisakPutnika[j].DajPodatke());
                    }
                }
            }
        }

        public Putnik UcitajPutnika()
        {
            Console.Write("Molimo Vas da unesete ime putnika: ");
            string ime = Console.ReadLine();

            Console.Write("Molimo Vas da unesete prezime putnika: ");
            string prezime = Console.ReadLine();

            bool pogresanUnos = true;
            int brojSedista = 0;

            while (pogresanUnos)
            {
                Console.Write("Molimo Vas da unesete broj sedista: ");
                string broj = Console.ReadLine();

                try
                {
                    brojSedista = Int32.Parse(broj);
                    pogresanUnos = false;
                }
                catch (FormatException)
                {
                    Console.WriteLine("Pogresan unos broja sedista. Molimo Vas da unesete
cifru!");
                }
            }

            Console.Write("Molimo Vas da unesete klasu: ");
            string klasa = Console.ReadLine();

            Klasa odabranaKlasa;

            switch (klasa)
            {
                case "Ekonomska":
```

```
                    odabranaKlasa = Klasa.Ekonomska;
                    break;
                case "Biznis" :
                    odabranaKlasa = Klasa.Biznis;
                    break;
                default:
                    odabranaKlasa = Klasa.Ekonomska;
                    break;
            }
            return new Putnik (ime, prezime, new Sediste(brojSedista, odabranaKlasa));
        }

        public Let UcitajLet()
        {
            Console.Write("Molimo Vas da unesete relaciju (u formatu odrediste-relacija):
");
            string relacija = Console.ReadLine();

            Console.Write("Molimo Vas da unesete datum: ");
            string datumString = Console.ReadLine();
            string[] nizStringova = datumString.Split('/');

            Let let = new Let(relacija, new DateTime(Int32.Parse(nizStringova[2]),
Int32.Parse(nizStringova[1]), Int32.Parse(nizStringova[0])));

            Console.WriteLine("Molimo Vas da unesete ukupan broj putnika koje zelite
uneti: ");
            int ukupanBrojPutnika = Int32.Parse(Console.ReadLine());

            for (int i = 0; i < ukupanBrojPutnika; i++)
            {
                let.DodajPutnika(UcitajPutnika());
            }
            return let;
        }

        static void Main(string[] args)
        {
            Console.WriteLine("Molimo Vas da unesete naziv aerodroma: ");
            string nazivAerodroma = Console.ReadLine();

            Aerodrom aerodrom = new Aerodrom(nazivAerodroma);

            aerodrom.spisakLetova.Add(aerodrom.UcitajLet());
            aerodrom.spisakLetova.Add(aerodrom.UcitajLet());

            Console.WriteLine(aerodrom.DajPodatke());
            Console.WriteLine("Broj putnika u biznis klasi: {0}",
aerodrom.brojPutnikaUBiznisKlasi);
        }
    }
}
```

Fakultet

Enum Status

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Fakultet
{
    public enum Status
    {
        Redovan,
        NaDaljinu,
        Diplomirao
    }
}
```

Klasa Osoba

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Fakultet
{
    public class Osoba
    {
        protected string ime;
        protected string prezime;

        public Osoba(string ime, string prezime)
        {
            this.ime = ime;
            this.prezime = prezime;
        }

        public virtual string DajPodatke()
        {
            return string.Format("{0} {1}", ime, prezime);
        }
    }
}
```

Klasa Profesor

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Fakultet
{
    public class Profesor : Osoba
    {
        private int brojRadneKnjizice;

        public Profesor(string ime, string prezime, int brojRadneKnjizice) : base(ime,
prezime)
```

```
        {
            this.brojRadneKnjizice = brojRadneKnjizice;
        }

        public override string DajPodatke()
        {
            return string.Format("{0}, {2}", base.DajPodatke(),
brojRadneKnjizice.ToString());
        }
    }
}
```

Klasa Ispit

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Fakultet
{
    public class Ispit
    {
        private string predmet;

        public string Predmet
        {
            get { return predmet; }
            set { predmet = value; }
        }
        private Profesor profesor;

        public Profesor Profesor
        {
            get { return profesor; }
            set { profesor = value; }
        }
        private DateTime datum;
        private int ocena;

        public int Ocena
        {
            get { return ocena; }
            set { ocena = value; }
        }

        public Ispit(string predmet, Profesor profesor, DateTime datum, int ocena)
        {
            this.predmet = predmet;
            this.profesor = profesor;
            this.datum = datum;
            this.ocena = ocena;
        }

        public string DajPodatke()
        {
            string predmetBezSamoglasnika = predmet.Replace("a", "");
            predmetBezSamoglasnika.Replace("e","");
            predmetBezSamoglasnika.Replace("i", "");
```

```
            predmetBezSamoglasnika.Replace("o", "");
            predmetBezSamoglasnika.Replace("u", "");

            return string.Format("{0}, {1}, {2}", predmetBezSamoglasnika[0].ToString(),
datum.ToShortDateString(), ocena.ToString());
        }
    }
}
```

Klasa Student

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Fakultet
{
    public class Student : Osoba
    {
        private string brojIndeksa;
        private List<Ispit> spisakPolozenihIspita;

        public List<Ispit> SpisakPolozenihIspita
        {
            get { return spisakPolozenihIspita; }
            set { spisakPolozenihIspita = value; }
        }
        private Status status;

        public Student(string ime, string prezime, string brojIndeksa, Status status) :
base(ime, prezime)
        {
            this.brojIndeksa = brojIndeksa;
            spisakPolozenihIspita = new List<Ispit>();
            this.status = status;
        }

        public override string DajPodatke()
        {
            int zbirOcena = 0;
            for (int i = 0; i < spisakPolozenihIspita.Count; i++)
            {
                zbirOcena += spisakPolozenihIspita[i].Ocena;
            }

            double prosecnaOcena = Convert.ToDouble(zbirOcena) /
Convert.ToDouble(spisakPolozenihIspita.Count);

            List<Profesor> spisakProfesora = new List<Profesor>();

            foreach (Ispit ispit in spisakPolozenihIspita)
            {
                if (!spisakProfesora.Contains(ispit.Profesor))
                {
                    spisakProfesora.Add(ispit.Profesor);
                }
            }
```

```
            string ispis = string.Format("{0}, {1}, {3}", brojIndeksa.ToString(),
base.DajPodatke(), prosecnaOcena.ToString());

            foreach (Profesor profesor in spisakProfesora)
            {
                ispis += string.Format("/t{0}/n", profesor.DajPodatke());
                foreach (Ispit ispit in spisakPolozenihIspita)
                {
                    if (ispit.Profesor == profesor)
                    {
                        ispis += string.Format("/t/t{0}/n", ispit.DajPodatke());
                    }
                }
            }

            return ispis;
        }

        public void PolozenIspit(Ispit noviIspit)
        {
            int brojPolozenihIspitaKodProfesora = 0;

            for (int i = 0; i < spisakPolozenihIspita.Count; i++)
            {
                if (spisakPolozenihIspita[i].Profesor == noviIspit.Profesor)
                {
                    brojPolozenihIspitaKodProfesora++;
                }
            }

            if (noviIspit.Ocena > 5 && brojPolozenihIspitaKodProfesora <= 2)
            {
                for (int i = 0; i < spisakPolozenihIspita.Count; i++)
                {
                    if (spisakPolozenihIspita[i].Predmet == noviIspit.Predmet)
                    {
                        spisakPolozenihIspita[i] = noviIspit;
                    }
                    else
                    {
                        spisakPolozenihIspita.Add(noviIspit);
                    }
                }
            }
        }
    }
}
```

Klasa Fakultet

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Fakultet;

namespace Fakultet1
{
    public class Fakultet
```

```
    {
        private string naziv;
        private List<Student> spisakStudenata;
        private int brojDesetki;

        public Fakultet(string naziv)
        {
            this.naziv = naziv;
            spisakStudenata = new List<Student>();
            brojDesetki = 0;
        }

        public Ispit UcitajIspit()
        {
            Console.Write("Molimo Vas da unesete naziv predmeta: ");
            string predmet = Console.ReadLine();

            Console.Write("Molimo Vas da unesete ime profesora: ");
            string ime = Console.ReadLine();

            Console.Write("Molimo Vas da unesete prezime profesora: ");
            string prezime = Console.ReadLine();

            Console.Write("Molimo Vas da unesete broj radne knjizice profesora: ");
            string brojRadneKnjizice = Console.ReadLine();

            Console.Write("Molimo Vas da unesete datum polaganja: ");
            string datum = Console.ReadLine();
            string[] nizDatum = datum.Split('/');

            bool pogresanUnos = true;
            int ocena = 0;

            while (pogresanUnos)
            {
                Console.Write("Molimo Vas da unesete ocenu: ");
                string ocenaString = Console.ReadLine();

                try
                {
                    ocena = Int32.Parse(ocenaString);
                    pogresanUnos = false;
                }
                catch (FormatException)
                {
                    Console.Write("Pogresno ste uneli ocenu. Molimo Vas da unesete
cifru!");
                }
            }

            return new Ispit(predmet, new Profesor(ime, prezime,
Int32.Parse(brojRadneKnjizice)), new DateTime(Int32.Parse(nizDatum[2]),
Int32.Parse(nizDatum[1]), Int32.Parse(nizDatum[0])), ocena);
        }

        public Student UcitajStudenta()
        {
            Console.WriteLine("Molimo Vas da unesete ime studenta: ");
```

```
            string ime = Console.ReadLine();

            Console.WriteLine("Molimo Vas da unesete prezime studenta: ");
            string prezime = Console.ReadLine();

            Console.WriteLine("Molimo Vas da unesete broj indeksa studenta: ");
            string brojIndeksa = Console.ReadLine();

            Console.WriteLine("Molimo Vas da unesete status studenta: ");
            string status = Console.ReadLine();

            Status odabraniStatus = new Status();

            switch (status)
            {
                case "Redovan":
                    odabraniStatus = Status.Redovan;
                    break;
                case "NaDaljinu":
                    odabraniStatus = Status.NaDaljinu;
                    break;
                case "Diplomirao":
                    odabraniStatus = Status.Diplomirao;
                    break;
                default:
                    odabraniStatus = Status.Redovan;
                    break;
            }

            Student student = new Student(ime, prezime, brojIndeksa, odabraniStatus);

            Console.Write("Molimo vas da unesete broj ispita koje cete uneti: ");
            int broj = Int32.Parse(Console.ReadLine());

            for (int i = 0; i < broj; i++)
            {
                student.SpisakPolozenihIspita.Add(UcitajIspit());
            }

            return student;
        }

        static void Main(string[] args)
        {
            Console.WriteLine("Molimo Vas da unesete naziv fakulteta: ");
            string nazivFakulteta = Console.ReadLine();

            Fakultet fakultet = new Fakultet(nazivFakulteta);

            fakultet.spisakStudenata.Add(fakultet.UcitajStudenta());
            fakultet.spisakStudenata.Add(fakultet.UcitajStudenta());

            string ispis = string.Format("Naziv fakulteta: {0}/n", nazivFakulteta);

            for (int i = 0; i < fakultet.spisakStudenata.Count; i++)
            {
                ispis += string.Format("{0}/n",
fakultet.spisakStudenata[i].DajPodatke());
```

```
            }

            ispis += string.Format("Broj desetki na fakultetu: {0}",
fakultet.brojDesetki);

            Console.WriteLine(ispis);
        }
    }
}
```