Međuispit iz Programskih paradigmi i jezika

29. travnja 2014. godine

Ispit nosi ukupno 30 bodova i piše se 70 minuta.
Na pitanja je potrebno odgovoriti kratko i precizno.

1. zadatak (5 bodova)

Objasnite što je to **višeobličje**. U programskom jeziku po vlastitom odabiru napišite program kojim se na bilo koji način **implementira i koristi** višeobličje (komentarima naznačite gdje se implementira, a gdje koristi).

2. zadatak (15 bodova)

Tvrtka provodi evidenciju ulazaka i izlazaka ovlaštenih osoba pomoću sustava za nadzor pristupa. Pristup prostorijama tvrtke imaju zaposlenici i gosti koji su evidentirani u sustavu. Zaposlenici su opisani svojim imenom, prezimenom i OIB-om, a gosti imenom, prezimenom i datumom rođenja. Nadzor se obavlja evidencijom ulazaka i izlazaka na kontrolnim mjestima koja se razlikuju po nazivu i lokaciji. Evidencija se sastoji od datuma i vremena, tipa evidencije (ulazak/izlazak) i osobe koja je evidentirana.

1) (10 bodova) Potrebno je ostvariti gore opisani model. Napisati kôd u programskom jeziku C#.
2) Nad modelom napisati LINQ upite koji vraćaju:
   a) (2 boda) zaposlenike kojima nikad nije bio evidentiran ni jedan ulazak niti izlazak.
   b) (3 boda) sva kontrolna mjesta s ukupnim brojem ulazaka (za svako mjesto posebno) poredanih silazno prema broju evidentiranih ulazaka.

Napomena: Prije LINQ upita napisati kôd kojim se instanciraju generičke liste koje sadrže podatke nad kojima upiti rade. Kôd za punjenje lista nije potrebno pisati (punjenje naznačiti komentarima).

3. zadatak (10 bodova)

Potrebno je napisati metode proširenja `IsNullOrEmpty` i `Clone` koje kao argument primaju `ICollection<T>`.

- Metoda `IsNullOrEmpty` vraća `true` ako joj je predana prazna kolekcija ili `NULL`-pokazivač.
- `Clone` vraća kolekciju koja se sastoji od kloniranih elemenata originalne (kao parametar predane) kolekcije (drugim riječima, obavlja duboku kopiju - „deep copy“). Elementi ulazne kolekcije moraju implementirati sučelje `ICloneable`.

Napomena: Metode proširenja je potrebno smjestiti u za to prikladne klase. Glavni program i poziv metoda nije potrebno pisati. Sučelje `ICloneable` *se sastoji od jedne metode:* `Object Clone();`

Rješenja:

2.

Model:

```
public class Tvrtka {
     public string Naziv { get; set; }
     public List<Zaposlenik> Zaposlenici { get; set; }
     public List<Evidencija> Evidencije { get; set; }
     public List<KontrolnoMjesto> Ulazi { get; set; }
 }

 public class Zaposlenik : Osoba {

     public string OIB { get; set; }
 }

 public class Gost: Osoba {

     public DateTime DatumRodjenja { get; set; }
 }

 public class Osoba {
     public string Prezime { get; set; }
     public string Ime { get; set; }
 }

 public enum TipEvidencije {
     Ulazak, Izlazak
 }

 public class Evidencija {
     public Osoba Osoba { get; set; }
     public KontrolnoMjesto Mjesto { get; set; }
     public DateTime Vrijeme { get; set; }
     public TipEvidencije Tip { get; set; }

 }

 public class KontrolnoMjesto {
     public string Naziv { get; set; }
     public string Lokacija { get; set; }
 }
```

LINQ:

1)

```
var nikadEvidentirani = tvrtka.Zaposlenici.
       Except(tvrtka.Evidencije.Select(e => e.Osoba as Zaposlenik).Distinct());
```

2)

```
var maxUlaz = tvrtka.Evidencije.Where(e => e.Tip == TipEvidencije.Ulazak).
       GroupBy(g => g.Mjesto).Select(g => new { Mjesto = g.Key, Broj = g.Count()
}).OrderByDescending(g => g.Broj);
```

3.

```
public static bool IsNullOrEmpty<T>(this ICollection<T> collection) {
            return collection == null || collection.Count == 0;
}

public static ICollection<T> Clone<T>(ICollection<T> collection) where T : ICloneable {
            if (!IsNullOrEmpty(collection)) {
                return collection.Select(item => (T)item.Clone()).ToList();
            } else {
                return null;
            }
}
```