

Objektno-orijentisano programiranje – Prvi kolokvijum

Zadatak : Obuke – termin 3

Napraviti Java konzolnu aplikaciju za evidenciju online obuka na sajtu Coursera. Coursera u ponudi ima različite vrste obuka i za svaku od njih znamo oblast kojoj pripada i profesore koji predaju. Profesori i polaznici se registruju na obuke.

Prvi deo – Implementacija UML dijagrama klasa

1. (6p) Implementirati sve elemente na priloženom dijagramu uz sledeće napomene :
 - za sva polja dodati set i get metode sa proverama koje se traže u zadatku
 - metode iz klase *Object*, *equals* i *toString* nadjačati prema proceni na mestima gde je to neophodno za realizaciju zahtevanih funkcionalnosti,
 - proizvoljno se mogu dodati konstruktori, metode kojih nema na dijagramu i implementacije ugrađenih interfejsa (na primer *java.lang.Comparable*)
 2. Klasa *Polaznik* predstavlja polaznike obuka. Za svakog polaznika čuva se ime, prezime i email. Ne može se desiti da imamo dva polaznika sa istim email-om.
 - (1p) Polaznika ispisujemo tako što ispišemo ime i prezime i u zagradama njegov email. (Na primer : Milka Canić [mcanic@raf.rs]).
 3. Klasa *Profesor* nam opisuje osobe koje učestvuju u izvođenju obuka. Za svakog profesora čuvamo ime i prezime, id, kao i koju oblast predaje.
 4. Za svaku *obuku* znamo naziv koji je jedinstven, sve profesore koji učestvuju u izvođenju, minimalan broj polaznika kao i oblast kojoj pripada obuka.
- Projekat* je vrsta obuke za koju se čuva maksimalan broj polaznika, a *Kurs* može da se održava onlajn i može imati druge kurseve kao preduslove koje polaznik mora da prođe pre datog kursa.
6. Interfejs *Registracija* ima metodu *registruj* koja kao argument ima obuku za koju se vrši registracija. Metoda vraća *false* ako nije moguća registracija. Možemo registrovati profesora i polaznika.
- (2p) Da bi profesor bio registrovan mora da bude iz iste oblasti kao i obuka, da nije već registrovan za obuku, kao i da nije registrovan na više od 3 obuke do sada. Za projekte važi da broj predavača ne sme biti veći od maksimalnog broja polaznika. Ukoliko je moguća registracija, profesoru se dodaje nova obuka, a obuci predavač.
- (2p) Da bismo polaznika registrovali na obuku potrebno je da polaznik ima validan email (da nije null i da sadrži znak '@') i da nije već prošao (bio registrovan) tu obuku. Na kurs možemo registrovati polaznike koji su prošli sve kurseve koji su preduslovi tog kursa, dok za projekat imamo maksimalan broj polaznika koji mora biti ispoštovan.
- (2p) Prilikom registracije polaznika na obuku, polazniku i obuci se dodaje nova instanca klase *ObukaPolaznika* sa jedinstvenim registracionim brojem koji se generiše na osnovu brojača *poslednjiBroj* koji počinje od 0.
- (2p) Metoda *registrovan* vraća *true* ako je polaznik odnosno profesor već registrovan na prosleđenu obuku, odnosno *false* ako nije.

(1p) *sveObukePolaznika(Polaznik p)* - metoda treba da ispiše sve obuke za koje je registrovan prosleđeni polaznik.

