Računarski fakultet, Univerzitet Union

Objektno-orijentisano programiranje - prvi kolokvijum

Zadatak: Ocenjivanje testova - termin 4

Napisati Java konzolnu aplikaciju za pregled rešenja testova sa različitim vrstama pitanja. Čuvaju se odgovori na pitanja i potrebno je proveriti tačnost odgovora i odrediti broj osvojenih poena. Mogu se ocenjivati pojedinačna pitanja i ceo test koji sadrži više pitanja.

Prvi deo - implementacija UML djagrama klasa

1. (5p) implementirati sve elemente na priloženom dijagramu uz sledeće napomene:
- za sva polja dodati set/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)

Pitanja imaju tekst pitanja i broj poena koje pitanje nosi. Postoje dve vrsta pitanja, pitanje sa ponuđenim odgovorima i pitanje sa slobodnim tekstom kao odgovorom. Čuvaju se pojedinačni odgovori na konkretno pitanja, a odgovori mogu biti grupisani u rešenje testa. Pojedinačni odgovori i rešenja testa mogu da se ocenjuju.

2. Svaka vrsta pitanja sadrži potrebne informacija za ocenjivanje. Pitanje kod koga je odgovor slobodan tekst sadrže ključne reči, a pitanja sa ponuđenim odgovorima sadrže indekse tačnih odgovora u listi ponuđenih. Prilikom postavljanja vrednosti za polja u klasama pitanja treba voditi računa o sledećem:

- (2p) metoda *dodajPonudjeniOdgovor* u klasi *PitanjaPonudjeniOdgovori* uzima kao argument tekst ponuđenog odgovora i informaciju da li je taj ponuđeni odgovor tačan. Ponuđeni odgovor se dodaje u listu ponudjenih odgovora, a ako je u pitanju tačan odgovor njegov indeks se čuva u listi tačnih odgovora. Odgovor se ne može dodati ako već postoji u listi i još jedno ograničenje je da broj tačnih odgovora ne može biti veći od broja poena koje pitanje nosi.

- (1p) prilikom setovanja kolekcije ključnih reči u klasi *PitanjeSlobodanOdgovor* implementirati proveru da broj ključnih reči mora biti jednak broju poena koje nosi pitanje i ne dozvoliti postavljanje vrednosti polja ako ovaj uslov nije zadovoljen.

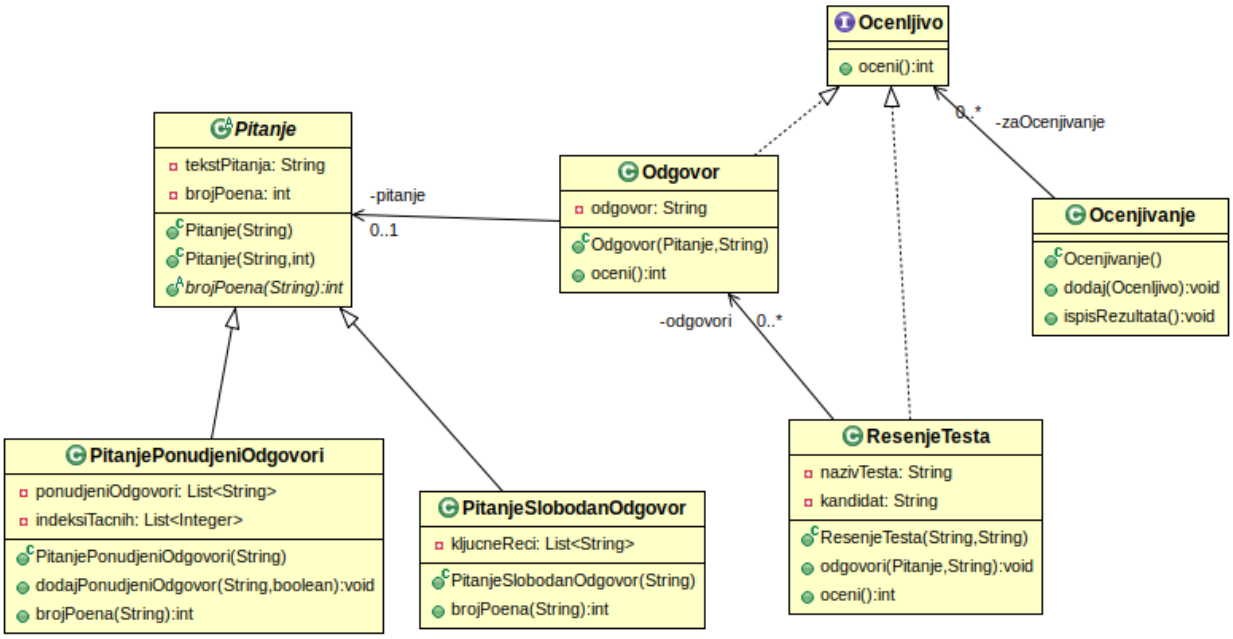
3. Određivanje broja poena koje se dodeljuju odgovoru zavisi od vrste pitanja, odgovor se prosleđuje kao argument metode *brojPoena(String)*.

- (3p) metoda *brojPoena(String)* za pitanje sa ponuđenim odgovorima treba da vrati broj tačnih odgovora. Kao argument se prosleđuje string koji sadrži sve odgovore razdvojene zarezom i potrebno je proveriti koliko je muđu tim odgovorima tačnih.
- (2p) metoda *brojPoena(String)* za pitanja sa slobodnim odgovorom treba da proveri koliko ključnih reči se nalazi u odgovoru i taj broj se vraća kao rezultat.

4. Interfejs *Ocenjivo* označava elemente koji mogu da se ocenjuju, to su odgovori na pitanja i rešenje testa.

- (1p) metoda *oceni()* u klasi *Odgovor* treba da vrati broj poena koji odgovor dobija na pitanju (pitanje i odgovor su polja klasa), odnosno rezultat poziva metode *brojPoena* za klasu Pitanje sa prosleđenim odgovorom, metoda *oceni()* za rešenja testa je zbor poena svih odgovora.

5. Ocenjivanje sadrži kolekciju elemenata za ocenjivanje
- (2p) metoda *ispisRezultata()* vrši ocenjivanje svih ocenljivih elemenata i ispisuje na standardni izlaz vrstu elementa koji se ocenjuje (test ili pojedinačan odgovor), za odgovor se ispisuje tekst pitanja, vrsta pitanja, odgovor i broj poena, a za test naziv testa i kandidata, broj ogovorenih pitanja i ukupan broj poena. Za funkcinalnost ispisa implementirti i iskoristiti *toString()* metode objekata iz kojih se uzimaju podaci za ispis.



Drugi deo - proširenje modela i dodatne funkcionalnosti

6. (7p) Dodati još jednu vrstu pitanja, to su pitanja sa spajanjem odgovora iz dve kolone (na primer spajanje sinonima). Kolone ne moraju imati isti broj elemenata što znači da se element iz jedne kolone može upariti sa više elemenata iz druge kolone. Osmisliti način čuvanja tačnih odgovora kao i sam format odgovora koji se prosleđuje kao string i određivanje broja poena, uz ograničenje da dodeljeni broj poena ne sme biti veći od broja poena koje predviđa pitanje.

4. (4p) Proširiti funkcionalnost ocenjivanja rešenja testova tako da se doda informacija o raspodeli poena i ocenama (na primer 30-40 - 6, 41-50-7,...) i ocenu uključiti u ispis testa (dozvoljeno je dodavati nove klase). Uključiti provere da se ne mogu postaviti preklapajući intervali.

5. (3p) Napraviti klasu sa main metodom u kojoj se kreira instanca klase *Ocenjivanje*, zatim po dva pitanja od svake vrste, napraviti rešenja testa koji sadrži odgovore na 5 kreiranih pitanja, i jedan odgovor na jedno pitanje dodati samostalno (kao vrstu ocenljivog). Zatim ispisati rezultate za sve elemente koji se ocenjuju.