

Pomoć: Ispravan je samo jedan odgovor.

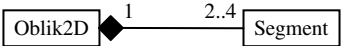
1. (1 bod) Koja temeljna značajka ili značajke **najviše razlikuju** moderan način oblikovanja programske potpore od tradicijskog?
 - A. Uvođenje analize zahtjeva.
 - B. *Uvođenje i analiza modela sustava.**
 - C. Evaluacija sustava.
 - D. Specifikacija sustava.
 - E. Sve gore navedeno.

2. (1 bod) Za spiralni model inženjerstva zahtjeva **ne** vrijedi:
 - A. U kasnijim iteracijama izrađuju se prototipovi radi validacije.
 - B. Koriste se iteracije ciklusa svih generičkih aktivnosti.
 - C. Rane iteracije imaju fokus na poslovnom modelu zahtjeva.
 - D. *Zahtjevi sustava određuju se u ranijim iteracijama.**
 - E. Zahtjevi se kroz iteracije specificiraju s različitom razinom detalja.

3. (1 bod) Usporedbom najznačajnijih modela procesa programskog inženjerstva odredi jedini ispravan odgovor.
 - A. RUP (engl. *Rational Unified Process*) je najstariji model.
 - B. Najviše novog kôda generira se u modelu zasnovanom na komponentama.
 - C. Implementacija promjena najlakša je u vodopadnom modelu.
 - D. Evolucijski model generira najbolju strukturu.
 - E. *Sve gornje tvrdnje su neistinite.**

4. (1 bod) Provjera sustava simulacijama i testiranjem:
 - A. Dokazuje da pogrešaka nema.
 - B. Identificira rubne slučajeve (engl. *corner cases*).
 - C. *Dokazuje moguće postojanje pogreške (engl. *bug*).**
 - D. Smanjuje vrijeme stavljanja programskog produkta na tržište (engl. *time to market*).
 - E. Sve gore navedeno.

5. (1 bod) U okviru generičke aktivnosti procesa programskog inženjerstva "Oblikovanje i implementacija", u dijelu "Oblikovanje programskog produkta", postoji niz podaktivnosti i rezultata tih podaktivnosti. Što neposredno prethodi podaktivnosti "Oblikovanje struktura podataka"?
 - A. *Oblikovanje i specifikacija komponenata.**
 - B. Oblikovanje i specifikacija arhitekture.
 - C. Oblikovanje i specifikacija algoritama.
 - D. Oblikovanje i specifikacija sučelja.
 - E. Apstraktna specifikacija programske potpore.

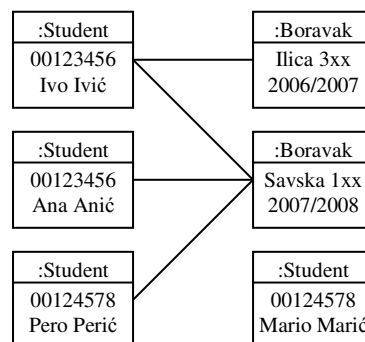
6. (1 bod) Prednosti pristupa oblikovanju programske potpore "odozdo prema gore" **najviše** se ogledaju u:
- A. U dobroj strukturi programske potpore.
 - B. ***Većem skupu modula za ponovno korištenje.**
 - C. Malom prostoru oblikovanja.
 - D. Jednostavnijem dokumentu oblikovanja arhitekture programske potpore.
 - E. Mogućnosti rane analize i uočavanje pogrešaka u oblikovanju.
7. (1 bod) Što **nije potrebno** uključiti u dokument oblikovanja arhitekture programske potpore?
- A. Referenciju prema dokumentu zahtjeva.
 - B. Opis s najviše razine promatranja.
 - C. Alternativna rješenja.
 - D. Prioriteti koji su vodili proces oblikovanja.
 - E. ***Kod s komentarima.**
8. (1 bod) Što **nije** dio UML skupa dijagrama statičkih pogleda?
- A. Dijagram razreda.
 - B. Dijagram objekata.
 - C. Dijagram obrazaca uporabe
 - D. ***Sekvencijski dijagram.**
 - E. Dijagram komponenata.
9. (1 bod) UML dijagram obrazaca uporabe može koristiti odnos <<extend>> između dva obrasca uporabe. Što od navedenog **nije** dio takvog odnosa?
- A. ***Naziv proširenja na odnosu <<extend>>.**
 - B. Uvjet izvođenja proširenja na odnosu <<extend>>.
 - C. Proširujući obrazac uporabe.
 - D. Bazni obrazac uporabe.
 - E. Točka proširenja (engl. *extension point*) u baznom obrascu uporabe.
10. (1 bod) Što definira veza prikazana na UML dijagramu:
- 
- ```

classDiagram
 class Oblik2D
 class Segment
 Oblik2D "1" *-- "2..4" Segment

```
- A. U sustavu se u bilo kojem trenutku nalazi najviše jedan objekt instanciran iz razreda Oblik2D,
  - B. **\*Svaki objekt razreda Oblik2D je povezan s dva do četiri objekta razreda Segment (nakon inicijalizacije).**
  - C. U sustavu se u bilo kojem trenutku može nalaziti samostalno od dva do četiri objekata nastalih iz razreda Segment.
  - D. Svaki objekt razreda Segment nasljeđuje atribut razreda Oblik2D.
  - E. Ako se obriše jedan objekt razreda Segment, mora se obrisati i pripadajući objekt razreda Oblik2D.

11. (1 bod) Dijagram objekata na slici nastao je iz dijagrama razreda. Kakva je brojnost (višestrukost) u dijagramu razreda [Student]---[Boravak] ? Svaka višestrukost veza prikazana na dijagramu vrijedi i općenitije, tj. ako npr. vrijedi 3---x tada vrijedi i N---x.).

- A. \*1..N---\*
- B. 1---1
- C. 1..N---1..N
- D. \*---1
- E. N---N



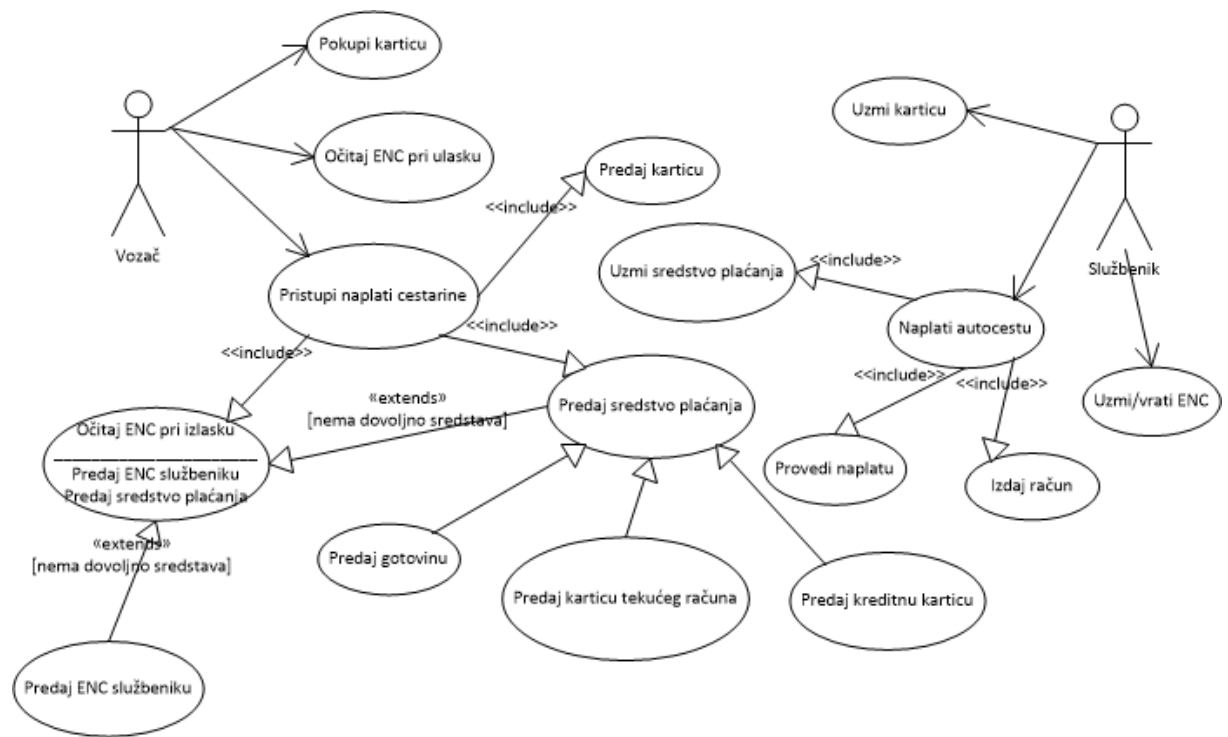
12. (1 bod) Poredaj od najviše razine apstrakcije prema najnižoj.

- A) Odgovornosti, metode, operacije
- B. \*Odgovornosti, operacije, metode
- C) Metode, operacije, odgovornosti
- D) Metode, odgovornosti, operacije
- E) Operacije, metode, odgovornosti

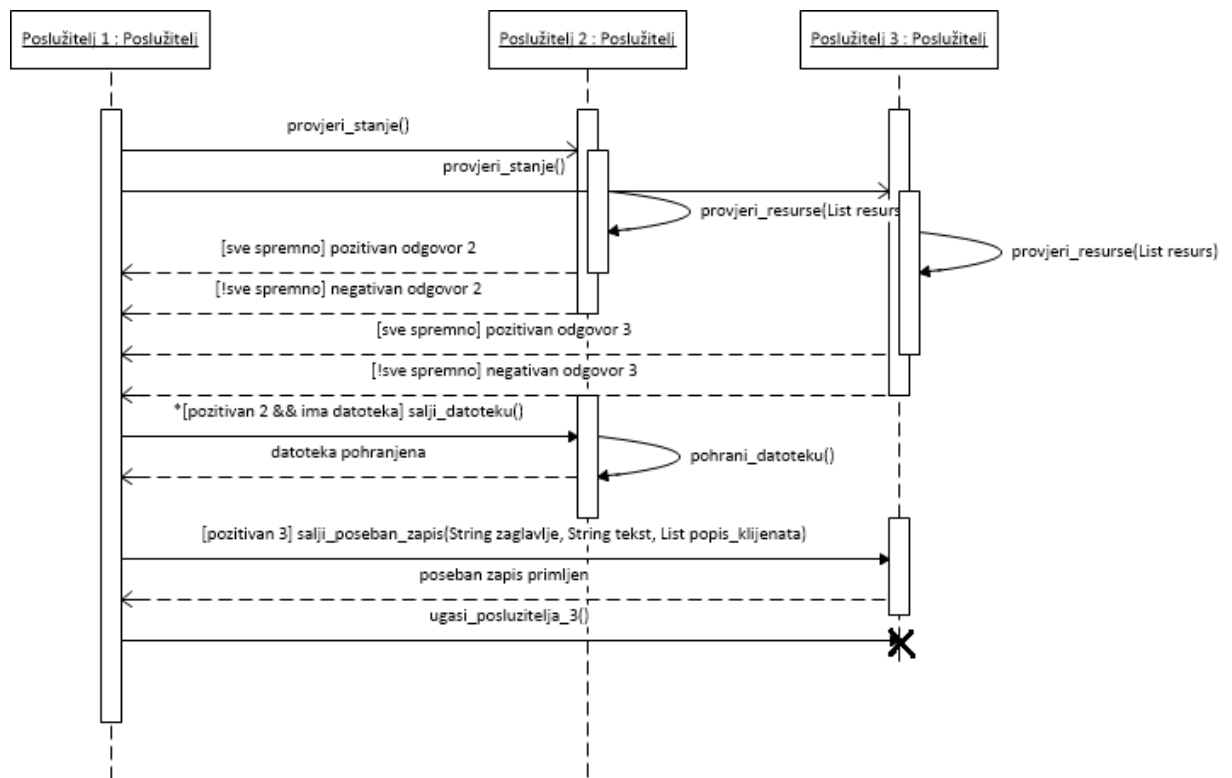
13. (1 bod) U okviru oblikovanja UML dijagrama razreda potrebno je odrediti pridruživanja (engl. *association*) između razreda. Pri tome **najviše** valja koristiti:

- A) Imenice i imeničke izraze u specifikaciji programske potpore.
- B) Akcije i navede ih se kao pridruživanja.
- C) Funkcionalne zahtjeve temeljem analize zahtjeva.
- D) UML dijagrame obrazaca uporabe (engl. *Use Case*).
- E) \*UML sekvencijske dijagrame.

14. (4 boda) **Modelirajte dijagramom obrazaca uporabe** sustav naplate na autocesti. Sustav naplate koriste vozači i službenici. Svaki vozač može pokupiti točno jednu karticu za autocestu ili očitati točno jedan ENC uređaj pri ulasku na autocestu. Pri izlasku s autoceste, svaki vozač pristupa naplati cestarine. Naplata cestarine uključuje predaju kartice za autocestu i predaju sredstva plaćanja službeniku. Tipovi sredstva plaćanja su gotovina, kartica tekućeg računa i kreditna kartica. U slučaju da je na ulazu na autocestu očitao ENC, vozač kao sredstvo naplate na izlazu ponovno očitava ENC (izlazno očitavanje). U specijalnom slučaju kad na ENC-u nema dovoljno sredstava pri izlasku s autoceste, vozač mora predati ENC službeniku i predati jedno od uobičajena tri sredstva plaćanja službeniku. Svaki službenik može uzeti jednu karticu za autocestu od pojedinog kupca i naplatiti cestarinu. Naplaćivanje cestarine uključuje uzimanje sredstva plaćanja, provedbu naplate i izdavanje računa. Službenik također može uzeti ENC i vratiti ENC vozaču (nakon postupka naplate cestarine).



15. (4 boda) **Modelirajte sekvencijskim dijagramom** mrežnu komunikaciju između tri poslužitelja. Svi poslužitelji se mogu smatrati objektima razreda Poslužitelj. Poslužitelj 1 najprije šalje poruke poslužiteljima 2 i 3 u kojima pita za njihovo stanje. Redoslijed slanja tih dviju poruka poslužiteljima nije bitan i poslužitelj 1 ne čeka na odgovor. Poslužitelji 2 i 3 kad prime poruku od poslužitelja 1 najprije pokreću internu rekurzivnu metodu kojom provjeravaju redom svoje resurse. Pretpostavite da se resursi nalaze u listi koja se smanjuje svakim rekurzivnim pozivom i nalazi se u argumenti interne metode). Ako su svi resursi spremni, tada poslužitelji 2 i 3 šalju povratnu poruku poslužitelju 1 s potvrdnim odgovorom o spremnosti. U protivnom, šalje se negativan odgovor o spremnosti. Ako je poslužitelj 1 primio pozitivan potvrđan o spremnosti od poslužitelja 2, poslužitelj 1 mu šalje  $N$  korisničkih datoteka, jednu po jednu. Poslužitelj 2 sprema lokalno te datoteke tako što poziva interni postupak pohrane za svaku od datoteka. Kad je završio, šalje odgovor poslužitelju 1 da je datoteka pohranjena. Ako je poslužitelj 1 primio pozitivan odgovor od poslužitelja 3, tada poslužitelj 1 šalje poslužitelju 3 poseban zapis koji se sastoji od zaglavlja (String), teksta (String) i popisa klijenata (List). Po primitku posebnog zapisa, poslužitelj 3 vraća odgovor poslužitelju 1 o uspješno primljenom zapisu. Primitkom tog odgovora, poslužitelj 1 šalje signal poslužitelju 3 da ga ugasi (*destroy* signal) te nastavlja sa svojim radom.



16. (4 boda) **Modelirati dijagramom razreda** jednu zgradu. Zgrada se sastoji od podruma, prizemlja i 3 do 5 katova, a na svakom katu nalazi se točno 4 stana. Svaki stan ima jedno zvono. Podrum, prizemlje i katovi su izvedeni iz zajedničkog razreda „Etaža“. Zgrada ima barem jedna ulazna vrata koja se nalaze u prizemlju. Ulaznim vratima je pridruženo tipkalo zvona koje ima operaciju „AktivirajZvono“. Zvono ovisi o tipkalu: nakon što se tipkalo pritisne zvono je aktivirano u odgovarajućem stanu. Na svakoj etaži nalazi se i barem jedna sijalica, a etaža ima operaciju „UpaliSvijetlo“. Sijalice ne pripadaju samo jednoj etaži već se mogu premještati između etaža. Svaka etaža ima atribut: razina (int, *public*). Svaki stan ima attribute: redni broj (int, *public*) i broj soba (int, *private*). Sve operacije razreda su javno dostupne. U izradi dijagrama navedite nazive uloga gdje je to potrebno, te označite vidljivosti svih atributa i operacija pomoću simbola. Dijagram izradite koristeći minimalni potreban broj razreda i njihovih međusobnih odnosa.

