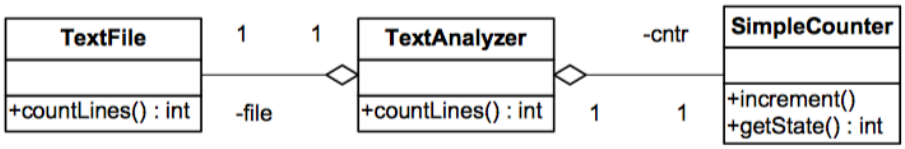
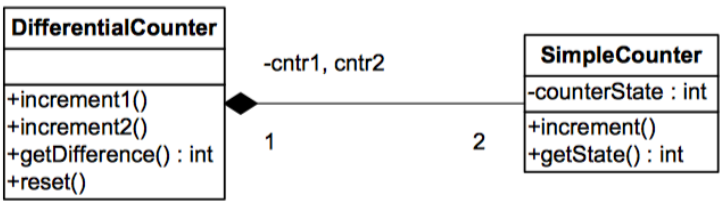
Usmeni - Blašković

1. **Navesti vrste veza između razreda.**
   Ovisnost - semantička relacija između dva elementa u kojoj promjena jednog (nezavisnog) elemnta može utjecati na semantiku drugog (zavisnog).
   Asocijacija - strukturana veza koja opisuje odnos između objekata. Može biti usmjerena, imati ime, uloge elemenata koje povezuje i kardinalnost veze. Posebna vrsta asocijacije je agregacija koja predstavlja strukturnu vezu cjeline i njenih dijelova.
   Generalizacija - veza među klasama kod koje jedna klasa dijeli strukturu i/ili ponašanje druge klase. Podklasa superklase nasljeđuje atribute, metode i odnose.

2. **Što su kompozicija i agregacija?**
   Agregacija - određuje "dio_od" vezu među razredima. Povezuje dio s cjelinom na način da se dio može izostaviti iz cjeline.



   Kompozicija - isto kao i agregacija, povezuje dio s cjelinom tako da se dio ne može izostaviti od cjeline.



3. **Što su varijable (od čega se sastoje, vidljivost, tipovi podataka)?**
   Varijabla je spremnik u radnoj memoriji kojem je dodijeljeno ime (identifikator) i vrsta podatka koji se u nju sprema.
   Java razlikuje varijable primitivne i referentne vrste. Vrijednost primitivne varijable je podatak primitivne vrste (boolean, int, double…), izravno je spremljen u varijablu i vrijednost varijable može se jednostavno izmijeniti. U varijablu referentne vrste sprema se referenca na objekt, vrijednost joj nikad nije sam objekt već referenca na njega.
   Lokalna varijabla obično se deklarira unutar glavnog bloka metode, njena vidljivost proteže se od mjesta deklaracije do kraja bloka u kojem je deklarirana.
   Java raspolaže ograničenim skupom primitivnih vrsta podataka (boolean, byte, short, char, int, long, float, double).

4. **Razlike i sličnosti objektno orijentiranog, funkcijskog, logičkog, imperativnog jezika.**
   Imperativni jezici - slijedno izvođenje naredbi, tok kontrole određen programskim brojilom, funkcijski pristup razvoju sustava, prednosti: blisko ljudskom poimanju sustavnog obavljanja posla prema naputku, blisko načinu rada računala, postižu se dobre performanse, nedostaci: u jednom koraku izvodi se jedna naredba, moguć (neželjen) pristup podatku i promjena

vrijednosti izvan kontrole programa, npr. C

Objektni jezici - enkapsulacija podatka/stanja programa u objekt i pristup podatku samo preko operacija definiranih u objektu, klase, nasljeđivanje, prednosti: podudarnost jezika i postupa razvoja (objektni pristup), izgradnja sustava odozdo prema gore od ponovno uporabivih dijelova (objekata), riješen problem imperativnog programiranja, nedostaci: slijed izvođenja naredbi: tok kontrole, kao kod imperativnih, npr. Java

Funkcijski jezici - izračunavanje (matematičkih) funkcija, vrijednosti iz jedne preslikavaju se u drugu domenu čime se izbjegava promjena vrijednosti podatka (stanja), deklarativni pristup ("što treba raditi") umjesto imperativnog ("kako treba raditi"), npr. Erlang

Logički jezici - matematičkom logikom specificira se problem, program je popis pravila temeljem kojih se izvodi zaključak, deklarativni pristup, npr. Prolog

5. Razlika između informacije i podatka - objasniti na primjeru govornog poziva.

Znanje o govornom pozivu: čemu služi i kako se odvija - definicija poziva. Informacija potrebna za govorni poziv: što predočuje - informacija o korisniku (adresa, pozivni broj), stanje (dostupan/nedostupan, slobodan/zauzet/blokiran), informacij o "mreži" (sredstva za ostvarivanje poziva raspoloživa). Podaci koji nastaju tijekom govornog poziva: kako su strukturirani i kakve su vrijednosti - podaci o korisniku (adresa, pozivni broj, stanje korisnika), definiraju informaciju.

Informacija - rezultat obrade podataka na način da dodaje znanje primatelju (Ako je T>=30, vani je vruće).

Podatak - činjenica predočena u formaliziranom obliku (Vani je 30 stupnjeva. T=30C).

6. Razlika između Jave, C-a, XML-a, UML-a i SQL-a.

Java je objektno orijentiran jezik, C je imperativan, UML je specifikacijski, XML je jezik za označavanje (markup), a SQL je deklarativni.

7. Što je polje?

Polje je blok u memoriji u koji je spremljen niz podataka iste vrste. U Javi je polje izvedeno kao objekt, ali posebnog karaktera. Polja su objekti posebne referentne vrste polja i njima se barata putem referenci, kao i ostalim objektima.

8. Što je preopterećenje?

Preopterećenje je postojanje više varijanti metodi istog imena unutar jednog razreda. Potpuni identitet metode sastoji se od njenog imena i popisa vrsta parametara koje prima (potpis metode, signature). Zbog toga mogu postojati dvije metode istog imena, a različitih parametara.

9. Što je sučelje?

Sučelje je, grubo govoreći, protokol ponašanja koji klasa može implementirati. Formalno se u Javi sastoji od metoda bez implementacije te konstanti. Sve deklarirane metode i konstante su u sučelju javne.

10. Što je apstraktna klasa?

Apstraktna klasa je klasa koja se ne može instancirati, sadrži samo popis metoda, najavljeno je njihovo postojanje ali nema implementacije. Razlikuje se od sučelja po tome što neka klasa može implementirati više sučelja ali naslijediti samo jednu apstraktnu klasu.

11. Koje su razlike i sličnosti između sučelja i apstraktne klase?

Zajedničko apstraktnim klasama i sučeljima je da se ne mogu instancirati.
Razlika je da sučelje ne može implementirati svoje metode, dok apstraktna klasa može.

12. Što je enkapsulacija?

Enkapsulacija je odvajanje sučelja objekta i same implementacije objekta. Implementacija objekta skrivena iza sučelja. Okolina (drugi objekti) moraju samo znati što objekt radi, a ne i kako radi. Svrha enkapsulacije je da omogući jednostavne promjene oko implementacije objekta, bez mijenjanja sučelja. Druga svrha je da se spriječi druge objekte da postave neku vrijednost unutar objekta na nevažeću vrijednost i time dovedu objekt u nedefinirano stanje (npr. postavljanje duljine stranice geometrijskog lika na vrijednost -2 ne bi imalo smisla).

13. Što je nasljeđivanje?

Nasljeđivanje je svojstvo definiranja razreda na temelju razreda koji su već definirani. Novi razred "nasljeđuje" sve metode i atribute već definiranog razreda, te dodaje neke nove. Npr. razred Kvadrat bi mogo naslijediti razred Geometrijski lik.

14. Što je polimorfizam?

Polimorfizam je svojstvo da se različiti tipova podataka upotrebljavaju koristeći ista sučelja. Polimorfizam omogućuje metodama razreda koji nasljeđuju da pregaze metode u razredu kojeg nasljeđuju. Npr. ako razred Kvadrat nasljeđuje razred Geometrijski lik, u razredu Kvadrat bi se "pregazile" metode za računanje opsega i površine.

15. Što su konstruktori i destruktori? Tko i kada alocira i oslobađa memoriju prilikom korištenja konstruktora i destruktora?

Konstruktor - posebna inicijalizacijska procedura, svaka klasa sadrži barem jedan, a ako ih je više, razlikuju se po broju i vrstama parametara. Različitim parametrima incijaliziramo objekt u stanje kakvo nam je potrebno. Postoji podrazumijevani konstruktor koji ne prima parametar i implicitno se dodaje klasi ukoliko konstruktor nije definiran, no ukoliko eksplicitno deklariramo novi, onda se ovaj ne dodaje implicitno više.
Destruktor - metoda kojom se oslobađaju resursi koje je zauzimao neki objekt nakon što on više nije potreban - garbage collection.

16. Koja je razlika između Implements i Extends?

Implements - obvezuje koja sve sučelja implementira, odnosno apstraktne specifikacije sučelja.
Extends - proširuje klasu (ne sučelje), odnosno dodajmo drukčiju funkcionalnost već neke prijašnje klase.

17. Koja je razlika između crne i bijele kutije?

Kod crne kutije nije potrebno poznavati unutarnju strukturu programa, provjeravamo obavlja li program funkciju za koju već znamo da bi trebala obavljati. Testovi se definiraju iz specifikacije, zahtjeva i ostalih opisa sustava.
Za bijelu kutiju potrebno je poznavati unutarnju strukturu programa. xUNIT testiranje je tipičan pristup, od ostalih mogućnosti: statička analiza - testiranja odnosno invarijante, tj. “runtime” analiza. Testovi se definiraju iz programskog koda uz posebnu pažnju na grananja, petlje i ostale kontrolne strukture.

18. Čemu služi XSLT i kako se pretvara za prikaz?

XSLT transformira XML dokument u drugi dokument, izvorno je ciljan za razvoj stylesheeta. Odabire čvorove pomoću XPatha, podržava for petlje, sortiranje čvorova, filtriranje, if i choice, kopiranje elemenata, varijable i predloške s parametrima. Kod Weba XSLT se koristi za pretvaranje XML dokumenata u dokumente XHTML ili HTML koji se mogu prikazati u pregledniku.

19. Objasniti DTD.

DTD je jedan od načina strukturiranja XML dokumenta, drugi je XML Schema, koji je noviji i opširniji. Opisuje jednostavan izgled XML dokumenta. Elementi DTD-a mogu se definirati u posebnoj, tzv. vanjskoj datoteci ili unutar samog XML dokumenta (tzv. unutarnji DTD).

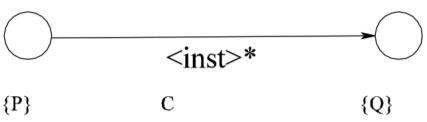
20. Koja je uloga Xpatha i XQueryja?

XPath je jezik za identifikaciju određenih dijelova XML dokumenata, ali nije XML. XSLT pomoću XPath-ovih izraza pronalazi i bira određene elemente ulaznog dokumenta i kopira ih u izlazni dokument ili ih dalje obrađuje. XPointer pomoću XPath-ovih izraza identificira određenu točku u XML dokumentu ili njegov određeni dio na koji upućuje neka XLink veza. XML Schema pomoću XPath-ovih izraza definira ograničenja jedinstvenosti i identiteta. XPath je jezik za biranje čvorova i skupova čvorova u tom stablu, a razlikuje nekoliko vrsta čvorova: korijenski čvor, čvorove elemenata, čvorove teksta, čvorove atributa, čvorove komentara, čvorove naredbi za obradu i čvorove imenika.

XQuery je upitni jezik za odabir odabir i transformaciju dokumenata u XML-u. Koristi se kao upitni jezik u bazama podataka temeljnima na XML-u ili kod upita koji uključuju nekoliko XML dokumenata u datotečnom sustavu. XQuery zapravao proširuje mogućnosti XPatha tj. svaki XPath izraz je ujedno i XQuery izraz. XQuery u svojim izrazima koristi XPath, ali treba paziti na inačice XPatha jer XQuery koristi inačicu 2 XPatha. Dok za razliku od njega XSLT koristi inačicu 1 XPatha.

21. Vrste semantike i opisi. Floyd-Hoare - što je i na kojim se principima temelji?

Aksiomatska semantika daje značenje frazama u jeziku iskazujući ih preko logičkih aksioma. Primjer je Floyd-Hoare logika:



P, Q su tvrdnje (assert).
C je instrukcija.
Ako je tvrdnja P točna prije izvršenja C onda je tvrdnja Q točna ili je program u blokadi.

Denotacijska semantika frazu u jeziku preslikava u opis ili denotaciju (najčešće psudokod ili matematički formalizam) - možemo ju okvirno opisati kao prevođenje fraze iz polaznog jezika u matematički formalizam. Primjer: algoritam implementiran u Javi prikažemo pseudokodom.

Operacijska semantika definira apstraktnu mašinu (npr. konačni diskretni automat) i značenje fraze opisuje kroz prijelaze u apstraktnoj mašini. Primjer: instrukcija c:=a+b.

22. Logging. Zašto ga koristimo i kome najviše služi?

Logging je tehnika skupljanja / evidentiranja podataka o radu programa. Koristi se u razvoju (omogućuje fleksibilnost pronalaska pogrešaka) i u isporučenom sustavu (dijagnostika grešaka).

23. Test prihvatljivosti, alfa i beta testiranje?

Testiranje prihvatljivosti mora odgovoriti na pitanje: prihvaća li krajnji korisnik programsku

potporu kao upotrebljiv proizvod.

Alfa testiranje - programska potpora u alfa statusu je najranije upotrebljive cjeline koju se može testirati

*Beta testiranje - o*dabrani skup potencijalnih budućih korisnika testira novu reviziju programske potpore.

24. Vrste logike.

Propozicijska, predikatna - propozicijska logika je algebarski sustav: propL=(A,Ω,Z,I). Sistematski ju izučava matematička logika. U predikatnoj p i q su sudovi i koristi se logika sudova.

Logika drugog reda

Logike višeg reda

Modalna, temporalna - modalna logika uvodi modalitete: moguće, vjerojatno i nužno kao dodatne operatore uz standardne I, ILI, negacija... Temporalna logika je poseban slučaj modalne logike koja modalne operatore interpretira u vremenu. Uvodi modalitete kao temporalne operatore: slijedeće, uvijek, sve dok, eventualno.

25. Vrste algebri.

Linearna, procesna, komunikacijske algebre, algebre utemeljene na logici.

26. Podjela logike.

Neformalna logika - proučava argumente iskazane običnim (govornim) jezikom

Filozofska logika - formalni opis prirodnog jezika

Matematička logika - logika sudova

Kompjuterska logika - primijenjena logika u telekomunikacijama, informatici, računarstvu

27. Objasniti razliku između predikatne, boolove i SAT logike.

Predikatna logika - predikati sadrže varijable i konstante, povezani su konjunkcijama, disjunkcijama, implikacijama, ekvivalencijama… Može biti neki izraz, tvrdnja o nekom objektu, ekspresivnija od boolove.

Boolova logika - provjeravaju se praktički samo true ili false izrazi bez konkretnih tvrdnji.

SAT logika - engl. satisfiability, sustav jednadžbi kroz CNF (konjunktivna normalna forma) ili DNF (disjunktivna) logičke formule mora bit istinite, problem modeliramo kroz logičke formule, daje bolje rezultate nego ostale metode.

28. Što je SAT test i navedi primjere.

SAT dolazi od engleske riječi satisfiability. Znači da sustav jednadžbi iskazan kroz CNF ili DNF logičke formule mora biti istinit. Problem iskazujemo (opisujemo ili modeliramo) pomoću sustava logičkih formula koje interpretira odgovarajući program - interpreter. Primjena: problemi u grafovima (bojenje grafa, planiranje kretanja robota, optimalna alokacija procesa u distribuiranim sustavima).

29. Razlika između validacije i verifikacije u objektno orijentiranom programiranju.

Validacija - radi li program ono za što je namijenjen?

Verifikacija - kako program radi ono za što je namijenjen?

30. Objasniti mock, integracijsko testiranje, H-L logika.

Mock - korištenje umjetnih/lažnih "mock" objekata koji glume pomoćne objekte za testiranje

onog što nama treba. Radi se testirajuća klasa koja je što "lokalnija" za jedinično testiranje. Testiramo samo objekt kojim se trenutno bavimo, a ne druge koji mu trebaju za njegov posao odnosno s kojima surađuje ili im proslijeđuje zadatke.
Integracijsko testiranje - zajedno sa jediničnim testiranjem čini white-box testiranje koda pojedinih programskih jedinica. Razlika je u odnosu na jedinično to što unit koristi lažne komponente sustava (mockove) dok integracijsko stvarne komponente sustava.
Floyd-Hoare logika - P,C,Q, P i Q su tvrdnje (engl. assert) a C je komanda; ako je P točno prije izvršenja C, onda je Q točno.
Tvrdnje se pišu u vitičastim zagradama {y=7} odnosno assert(y==7) kao instrukcija.
Komande su tipa poziv funkcije zbroji(x,y) ili x=y+1.

31. Kako prebaciti string u integer?

Koristeći Integer.parseInt() - parsira string kao signed decimal integer i vraća integer vrijednost.

32. Što je test slučaj (TESTCase)?

Test slučaj opisuje pojedinačne testove i sadržava: objekt/funkcionalnost koju testiramo (dio programskog koda), ulazne parametre, izlazne parametre, očekivane izlazne parametre. Može sadržavati jednu ili više assert jUnit naredbi.

33. Što je test sekvenca (TESTSuite)?

Test sekvenca je skup koji sadržava jedan ili više pojedinačnih test slučajeva. Testira funkcionalnost prema modelu ili specifikaciji - mora se znati što PP implementira. Jedan ili više testova ili test sekvenci je sastavni dio dokumentacije koja se predaje testerima ili služi za održavanje PP.

34. Što predstavlja assert u jUnitu?

Assert je testirajuća metoda u jednoj liniji - tvrdnja. Omogućuje testiranje točnosti pretpostavki u programskom kodu.

35. Što je integracijsko testiranje?

Kod integracijskog testiranja svaka od klasa je u interakciji s nekim klasama: potrebno je par po par, skup po skup takvih interakcija testirati. Kod neobjektnih programa testiramo interakcije komponenti, blokova, modula…

36. Što je regresijsko testiranje?

Svaka klasa, komponenta, modul, … tijekom testiranja / životnog vijeka doživljava niz modifikacija (zbog pogrešaka ili zbog dodatnih zahtjeva na funkcionalnost). Modifikacije mogu promijeniti (već testirano) ponašanje. Regresijsko testiranje mora ustanoviti uzroke takvog ponašanja programske potpore. Modifikacije ne smiju poremetiti funkcionalnost ostatka sustava.
Regresijsko testiranje nakon novih promjena mora pronaći: klase koje nitko ne koristi, metode koje nitko ne poziva, promijenjenu funkcionalnost sustava.

37. Čemu služi Java PathFinder?

JavaPathFinder je razvijen za otkrivanje grešaka i neispravnosti u Java programima. Namijenjen je za verifikaciju raspodijeljenih sustava. U pozadini JPF-a nalazi se provjera modela (SPIN model checker). Dodatno može otkriti bugove proizašle iz konkurentnosti

(mrtve petlje, istovremenu neautoriziranu upotrebu resursa, nepodržane signale, nikad pozvane metode…), Java runtime greške (unhandled exceptions, krivo korištenje heap resursa, cycle budgets…).

Verifikacija Java programa odvija se u sljedećim koracima: 1) dodavanje posebnih klasa i assert() naredbi, 2) prevođenje java datoteka iz java u class oblik, 3) pozivanje class fileova s JPF, 4) potrebno je dodati opcije za JPF.

38. Greška, pogreška i neispravnost, što je Expect i što je TTCN-3 i testiranje primjenom modela?

Greška / pogreška - (error) se događa kada iznimka (exception) nađe put do testirajuće metode dok se test izvodi npr. NullPointerException.

Expect - programsko pomagalo za automatizaciju interaktivnih programa, skriptni jezik koji razgovara s programom. Svaki expect program sastoji se od 1) incijalizacije 2) send (nalog) 3) expect (očekivanja vrijednosti) 4) ponavljanja 2->3 5) kraja. Ako izostane očekivani rezultat, test nije uspio.

TTCN-3 - Testing and Test Control Notation ver. 3 - standardizirani jezik za opis i definiciju testova, test slučajeva i test podataka. Ima bogatiji jezik od Expecta i opisuje sve na formalan način. Podatke definira preko ASN.1 notacije (weak - strong types).

Expect i TTCN-3 definiraju procedure - sekvence tako da “razgovaraju", odnosno međusobno razmjenjuju poruke.

Praktična pitanja

1. **Nacrtane su dvije klase, potrebno ih je spojiti asocijacijom te napisati u Javi.**
2. **Zadane su dvije klase, povezati ih agregacijom i napisati Java kod za to.**
3. **Zadan je class dijagram - treba napisati kod u Javi za njega.**
4. **Nacrtana dva razreda u uml dijagramu, treba ih napisati u Javi, obične set i get metode.**
5. **Napisati XML kod za neki telefonski imenik. Napisati izraz kojim bi dobio brojeve telefona iz njega.**
6. **Zadan tekst odvojen zarezima - zapisati ga u XML-u, nacrtati stablo.**