1. Svaki proces programskog inženjerstva sadrži iteracije u pojedinim fazama. Iteracije se mogu izvesti inkrementalnim ili spiralnim postupkom. Koje su prednosti (navedi barem tri) inkrementalnog pristupa iteracijama.
    - kupac dobiva vrijednost sa svakim inkrementom
    - temeljna funkcionalnost sustava se ostvaruje u ranim fazama projekta
    - rani inkrementi služe kao prototipovi na temelju kojih se izlučuju zahtjevi za kasnije implemente
    - smanjen rizik neuspjeha projekta
    - prioritetne funkcionalne usluge sustava imaju mogućnost detaljnijeg ispitivanja

2. Navedi četiri generičke aktivnosti u svakom procesu inženjerstva
    - Studija izvedivosti
    - analiza i izlučivanje zahtjeva
    - validacija zahtjeva
    - upravljanje zahtjevima

    Ovo vrijedi za proces **programskog inženjerstva:**

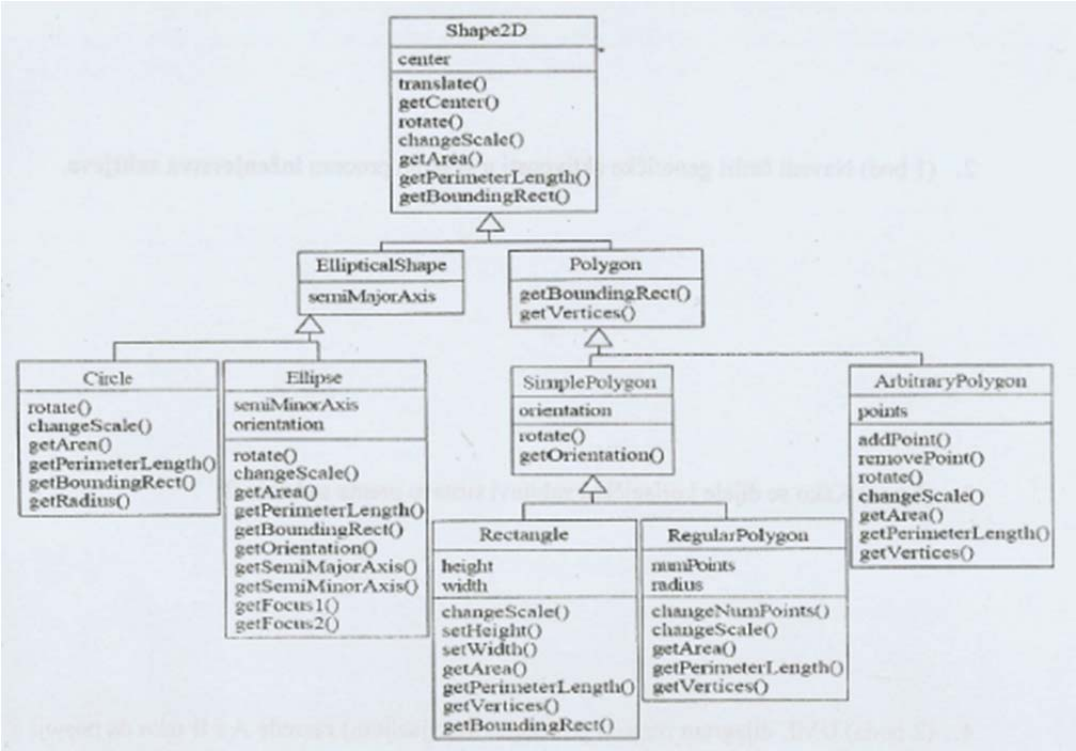
    - specifikacija
    - razvoj i oblikovanje
    - validacija i verifikacija
    - evolucija

3. Kako se dijele korisnički zahtjevi sustava prema sadržaju
    - funkcionalni zahtjevi
    - nefunkcionalni zahtjevi
    - zahtjevi domene primjene

4. UML dijagram razreda povezuje (asocijacijom) razrede A i B tako da postoji brojnost:
   [A] 1..4_____3..5 [B]
   Neka se tijekom izvođenja u jednom času u sustavu nalazi 3 objekta nastala iz razreda A. Koliko se najmanje, a koliko najviše objekata razreda B može u tom času nalaziti u sustavu?
   MIN: 3 (Jer sve tri instance od A mogu referencirati isti B)
   MAX: 3*5=15

5. Zadana je hijerarhija razreda kao na slici. Neka je također zadana varijabla var_a određenog (zadanog) tipa. To znači da ta varijabla može sadržavati bilo koji objekt iz hijerarhije toga zadanog tipa. Pod strogim uvjetom da se ne stvaraju novi razredi ni metode navedi za svaki zadani tip var_a iz kojih razreda se mogu izvesti zadane operacije (metode). Priznaje se pojedini odgovor samo ako su navedeni SVI ispravni razredi. Također navedi za svaki poseban slučaj da li se radi o dinamičkom povezivanju (dynamic binding) ili statičkom povezivanju (static binding).

	Tip varijable **var_a**:	Operacije s varijablom **var_a**:
a)	**Ellipse**	**rotate()**
b)	**Ellipse**	**translate()**
c)	**Polygon**	**getBoundingRect()**
d)	**EllipticalShape**	**changeScale()**



Tu ima gomilu raznih rjesenja:

Moje:

a) Ellipse – statičko
b) Shape2D – statičko
c) Polygon, Shape2D – dinamičko
d) Shape2D – dinamičko

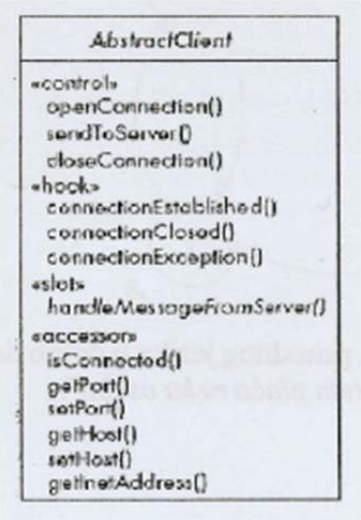
kolega Over9000Volts:

Ellipse - statičko
b) Ellipse - statičko
c)Polygon,SimplePolygon,ArbitraryPolygon,Rectangle ,RegularPolygon - dinamičko
d)EllipticalShape,Circle,Ellipse - dinamičko

EDIT: iako ne znam dal se pita razred objekta koji može bit u varijabli ili razred u kojem je definirana metoda, ako je ovo drugo onda je

a) Ellipse - statičko
b) Shape2D - statičko
c)Polygon,Rectangle - dinamičko
d)Shape2D,Circle,Ellipse – dinamičko

6. Na slici je prikazan dio generičkog radnog okvira za objektno oblikovanje klijent-poslužitelj (engl OCSF). Prikazan je razred **AbstractClient**.
Opiši prirodnim jezikom sekvencu (slijed) uporabe tog razreda. Pritom napiši samo ono što je obvezno napraviti u oblikovanju klijenta da bi zaživio, poslao poruku poslužitelju i bio spreman prihvatiti poruku poslužitelja. Navedi koje se metode pritom pozivaju.



- Kreiraj podrazred od AbstractClient
- U podrazredu implementiraj handleMessageFromServer() <<slot>> metodu
- Napiši kod koji:
  - Kreira instancu klijenta(podrazreda, start prve dretve)
  - Pozove openConnection (start druge dretve)
  - Šalje poruku poslužitelju uporabom sendToServer() metode
- Implementiraj connectionEstablished() callback metodu
- Implementiraj connectionClosed() callback metodu
- Implementiraj connectionException()callback metodu

7. Neka su F1 i F2 dobro definirane formule matematičke logike. Definiraj što znači da:
   a. F1 i F2 su semantički ekvivalentne
   b. F2 je logička posljedica F1

a) F1 i F2 su semantički ekvivalentne ili jednake ako imaju jednaku (istu) istinosnu vrijednost za svaku interpretaciju
b) F2 je logička posljedica F1 ako svaka interpretacija koja je lijevo od |= daje istinitost mora i desnoj strani dati istinitost (F1|=F2)

8. Definiraj predikate i preslikaj rečenicu prirodnog jezika u dobro definiranu formulu logike predikata prvog reda: „U Hrvatskoj postoje najmanje dva nacionalna parka"

x – nacionalni park
y – nacionalni park
postojiu(x, mjesto) – postoji u mjestu
=(x,y) – x i y su jednaki

$$\exists x \exists y [postojiu(x, Hrvatska) \land postojiu(x, Hrvatska) \land \neg = (x, y)]$$

Kako je Sruk riješio na predavanju:
NP(x) - x je nacionalni park
HR(x) - x je u Hrvatskoj
RAZ(x,y) - x i y nisu isti

$$\exists x \exists y [NP(x) \land HR(x) \land NP(y) \land HR(y)(x, y)]$$

9. Definiraj predikate i preslikaj rečenicu prirodnog jezika u dobro definiranu formulu logike predikata prvog reda: „Svaku štetu plaća neka osoba"
   x – šteta
   y - osoba

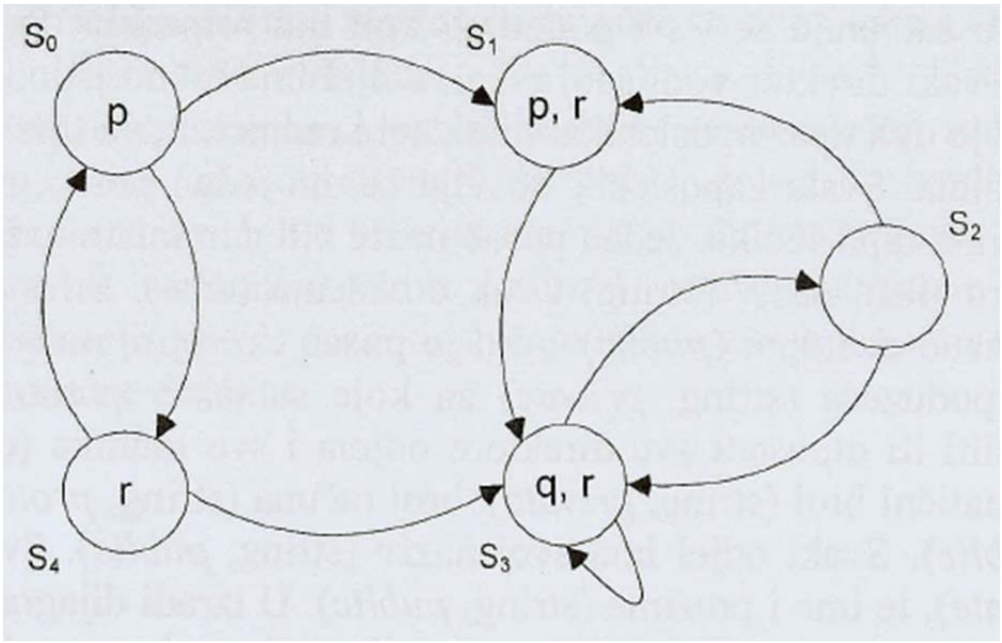
$$\forall x \exists y(plati(x, y))$$

10. Preslikaj rečenicu prirodnog jezika u dobro definiranu formulu CTL vremenske logike: „Proces uvijek može iz nekritičnog stanja u slijedećem trenutku zahtijevati ulazak u kritično stanje"

Pomoć: Neka su oznake za propozicijske simbole : n-proces nije u kritičnom stanju; c – proces je u kritičnom stanju; t – proces zahtijeva ulaz u kritično stanje

$$\mathrm{AG}[\mathrm{n} \Rightarrow AXt]$$

11. Za Kripke strukturu M zadanu slikom potrebno je odrediti:
    a. S (skup stanja), R (relaciju prijelaza), L (funkciju označavanja)
    b. skup stanja za koja vrijedi EX p
    c. skup stanja za koja vrijedi EG r



a) S:{S0, S1, S2, S3, S4}
   R:{(S0,S1), (S0,S4), (S1,S3), (S2,S1), (S2,S3), (S3,S3), (S3,S2), (S4,S0), (S4,S3)}
   L:L(S0)=p, L(S1)=p,r, L(S3)=q,r, L(S4)=r
b) S0, S2, S4
c) S1, S3, S4

12. Navedi tri tipična uzroka kvara sučelja komponente
    - Pogrešna uporaba
    - Nerazumijevanje sučelja
    - Vremenske pogreške
13. Neka je sustav koji prima na ulazu cijele brojeve u rasponu 10-20 podvrgnut funkcijskom ispitivanju (kao crna kutija). Navedi pogodne ispitne slučajeve tako da se pokriju sve ekvivalentne podjele.
    - 0-9
    - 10-20
    - 21-inf

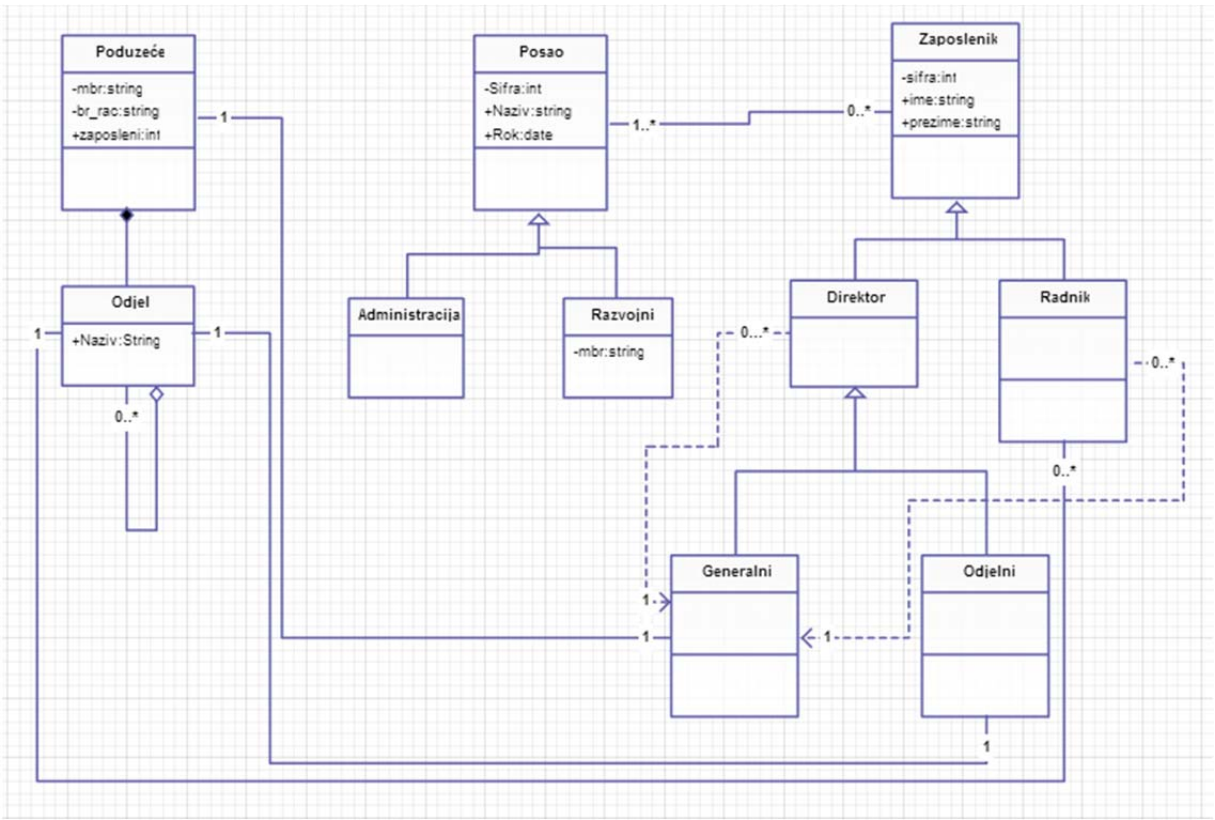
napomena by kolega Pjuwy: ispitni slučaj je par (ulaz, ocekivan izlaz), a ne particije

ispituju se brojevi iz svake skupine te je potrebno ispitati granične slučajeve

14. Dijagram razreda

14. (4 boda) Modelirajte UML dijagramom razreda organizaciju poduzeća.

Neko poduzeće sastoji se od jednog ili više odjela. Poduzeće vodi generalni direktor. Svaki odjel može sadržavati jedan ili više pododjela koji imaju ista svojstva kao odjeli. Zatvaranjem nadređenog odjela ne zatvaraju se i svi pododjeli koji mu pripadaju. Svaki odjel ima svog direktora i radnike. Svaki direktor vodi svoj odjel. Odjel ima točno jednog direktora i barem jednog radnika. Postoje dva tipa zaposlenika: direktori i radnici, i dva tipa direktora: generalni direktor i direktor odjela. Svaki zaposlenik obavlja barem jedan posao, a svaki posao može obavljati nijedan ili više zaposlenika. Jedan posao može biti administrativni ili razvojni. Svaki posao ima svoju šifru (int), naziv (string) i rok dovršenja (date). Šifra je privatni podatak (*private*), ostali su javno dostupni (*public*). Ako je posao razvojni, onda on sadrži i matični broj nekog drugog poduzeća (string, *private*) za koje se takav posao obavlja. Generalni direktor može zaposliti ili otpuštati sve direktore odjela i sve radnike (oni ovise o njemu). Poduzeće ima svoj matični broj (string, *private*), broj računa (string, *protected*) i ukupan broj zaposlenika (int, *public*). Svaki odjel ima svoj naziv (string, *public*). Svaki zaposlenik ima svoju šifru (int, *private*), te ime i prezime (string, *public*). U izradi dijagrama navedite nazive rola gdje je to potrebno, te označite vidljivosti svih atributa i operacija razreda pomoću simbola.



*ovisnost nema brojnost, pa to zanemarite! :)

15. Dijagram stanja

15. (4 boda) Modelirajte UML dijagramom stanja (engl. *statechart*) izmjenu podataka u jednoj ćeliji unutar tabličnog kalkulatora (npr. Excela).

Unos podataka u ćeliju započinje klikom miša ili pozicioniranjem putem tipkovnice, prilikom čega se na ekranu podeblja ćelija u tablici i omogući se unos teksta. Pretpostavite da unos teksta traje dok se pritišće bilo koji znak osim znakova "Enter", "Tab" ili "Esc". Prilikom unosa, pritisnuti znak se ispisuje na ekran, ali ne briše se dosadašnji sadržaj ćelije. U slučaju uspješnog završetka unosa (znakovi "Enter" ili "Tab"), dolazi do izmjene podataka u ćeliji. Najprije se dosadašnji sadržaj ćelije briše iz memorije. Potom se trenutni podatak pohrani u memoriju i na kraju se podatak ispiše u ćeliju. U slučaju obustave unosa (znak "Esc"), podatak u ćeliji ostaje netaknut. U svakom slučaju nakon završetka unosa ćelija se defokusira, odnosno postaje normalne debljine.