**OBJEKTNO ORIJENTISANO**

**PROGRAMIRANJE 1**

ODGOVORI NA PITANJA ZA

USMENI DEO ISPITA

JANUAR 2021

Git: <https://github.com/NikolaVetnic/OOP1>

Objekti i klase.

01

**OBJEKTNO ORIJENTISAN PROGRAM:** ideja – pakovanje podataka i funkcija koje ih obrađuju u jednu celinu predstavljenu jednom promenljivom, tj. objektom sačinjenom od atributa/polja (promenljive, opisuju stanje) i metoda (funkcije, definišu ponašanje); OO program je skup objekata u međusobnim interakcijama (npr. objA poziva metoda nad objB).

**KLASA:** definicija skupa istorodnih objekata (ista specifikacija stanja i ponašanja, tj. atributi i metode), klase su referencijalni tipovi objekata (tip određuje skup vrednosti i skup operacija nad promenljivom); pisanje OO programe podrazumeva definiciju klasa i instanciranje (kreiranje) objekata definisanih klasa. *Primer klase i objekata sa crteža* – klasa kao „prototip“, objekti kao konkretne instance sa različitim stanjima.

**IMPERATIVNI PJ:** program je formalni opis procesa (imperativni PJ) ili specifikacije (deklarativni PJ) izračunavanja u PJ, ostvaren kroz skup naredbi (naredbe dodele, kontrole toka, itd.) koje menjaju vrednosti skupa promenljivih (odnosno stanje programa). Kategorije: 1) proceduralni – dekompozicija u funkcije i procedure, lokalne i globalne promenljive; 2) modularni – dekompozicija u module (logički srodne definicije promenljivih, funkcija i tipova, sa privatnim i javnim delom); 3) OO – dekompozicija u klase (logički srodne def. promenljivih i funkcija, klase kao tipovi, privatni i javni deo, nasleđivanje klasa).

**KONSTRUKTORI I DESTRUKTORI:** konstruktor – specijalni metod koji se poziva pri instanciranju klase radi inicijalizacije stanja objekta, destruktor – specijalni metoda koji se poziva kada se objekat uništava. U Java PJ: 1) instanciranje operatorom new; 2) rezultat je referenca na instancirani objekat; 3) moguće je više referenci na isti objekat; 4) nema destruktora, automatsko uništavanje objekata (*garbage collector*) kada više nema referenci na objekat.

Apstrakcija, enkapsulacija i sakrivanje informacija.

02

**APSTRAKCIJA:** suština PJ je da omoguće mehanizme: 1) apstrakcije - sakrivanje (zanemarivanje, odlaganje, zaboravljanje) nebitnih (tehničkih, implementacionih) detalja, što smanjuje kompleksnost; 2) dekompozicije – razlaganje entiteta (problema) u pod-entitete koji su manje kompleksnosti. Paradigme: 1) proceduralni PJ – dekompozicija u procedure, gde korisnik zna samo zaglavlje a ne i telo procedure da bi je koristio; 2) OO PJ – dekompozicija u objekte, gde korisnik zna samo zaglavlja javnih metoda objekta.

**SAKRIVANJE INFORMACIJA:** public i private delovi objekta, kroz modifikatore vidljivosti/pristupa prilikom definicije atributa i metoda klase – privatni atributi vidljivi samo u klasi, privatne metode se pozivaju samo iz iste klase, javnim atributima i metodama se može pristupati iz drugih klasa.

**ENKAPSULACIJA:** dizajn objekta koji skriva detalje nebitne za korišćenje od korisnika (ako su svi atributi privatni tada objekat ima potpunu kontrolu nad svojim stanjem).

Agregacija i kompozicija klasa.

03

**TEKST:** tekst