

## Klase i objekti – uvod u objektno-orijentisano programiranje u jeziku Java

Klase su obrasci na osnovu kojih se kreiraju objekti. Klasa **Tacka** (vidi fajl *Tacka.java*) predstavlja apstrakciju matematičkog pojma tačke u koordinatnoj ravni i predstavlja obrazac po kojem će naš program kreirati tačke. Slično, klasa **Duz** (vidi fajl *Duz.java*) predstavlja model po kojem će naš program kreirati duži. Klasa **Trougao** je naš “model” matematičkog pojma trougao (fajl *Trougao.java*).

Promjenljive čiji je tip neka klasa (npr. t1, t2, d1, d2 u metodi main klase **Test**) nazivano objektima ili instancama klase.

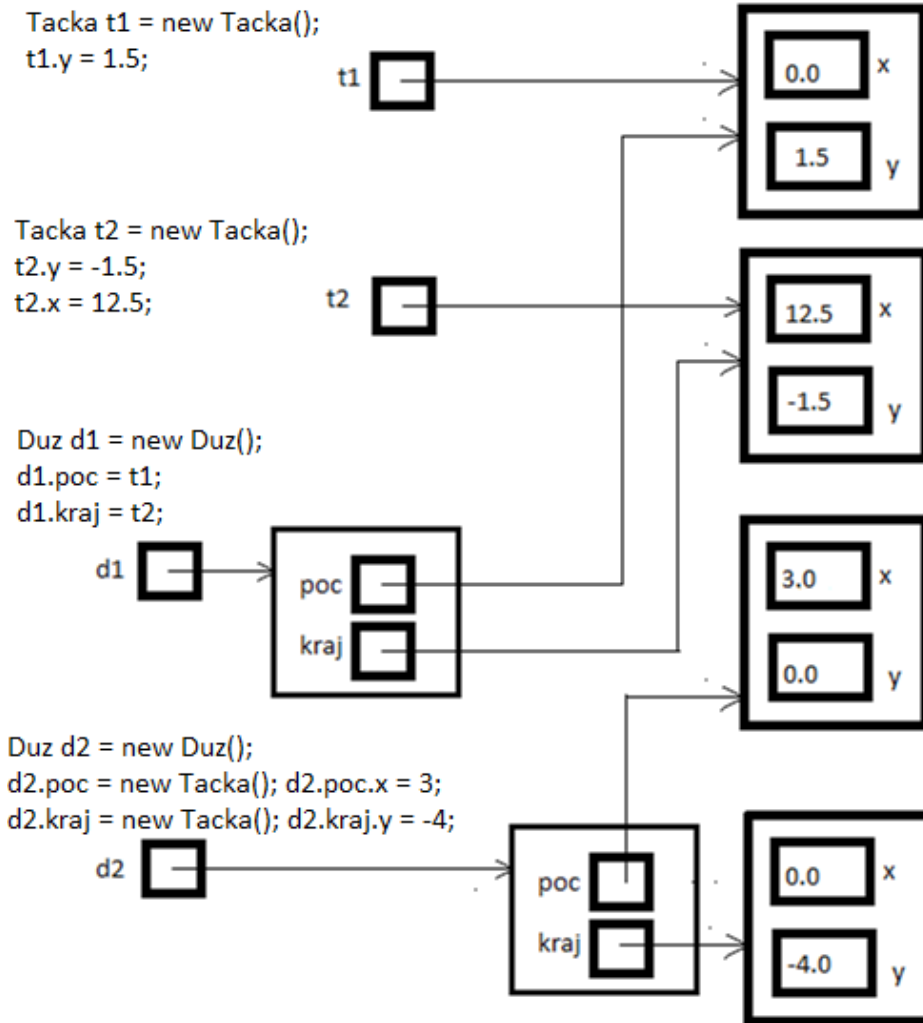
Klasa sadrži promjenljive (tzv. članice klase, engl. class members) i metode. Članice klase opisuju stanje objekta (engl. state) dok metodi opisuju ponašanje objekta (engl. behaviour) .

Npr. u klasi **Tacka**, članice klase su x i y (obje tipa double) i one redom predstavljaju apscisu (koordinatu x) i ordinatu (koordinatu y) proizvoljne tačke. Dakle, svaki objekat klase **Tacka** imaće svoju apscisu i svoju ordinatu. Slično, u klasi **Duz**, svaki objekat klase **Duz** imaće početnu tačku (članica klase poc) i krajnju tačku (članica klase kraj). U klasi **Test** možemo kreirati više objekata klase **Tacka** (vidi fajl *Test.java*, promjenljive t1 i t2) i više objekata klase **Duz** (promjenljive d1 i d2).

Sve članice klase se prilikom kreiranja objekta postavljaju na “nultu” vrijednost za odgovarajući tip. Za primitivne tipove byte, short, int i long ta vrijednost je nula (0), za tipove float i double je 0.0, a za tip char je blanko karakter ' '. Ako je članica klase objekat, tada je njena vrijednost null. Npr. u klasi **Trougao** (fajl *Trougao.java*) imamo samo tri članice klase: tjemeA, tjemeB i tjemeC. Prilikom kreiranja objekata tipa Trougao, sve tri članice klase postavljaju se na „nultu” vrijednost za tip **Tacka** tj. na vrijednost null.

Svi objekti u programu, tj. promjenljive tipa klase, predstavljaju se referencama, kako je prikazano na slici.

Obratite pažnju da reference omogućavaju da neki dijelovi memorije budu zajednički za više objekata (npr. tačka t1 je početak duži d1).



Svaka klasa sadrži jedan ili više konstruktora koji su odgovorni za kreiranje objekata te klase. Konstruktor ima isto ime kao i klasa i može imati argumente. Ako ne napišete konstruktor, Java sama napravi podrazumijevani (tzv. default) konstruktor bez argumenata koji vrijednosti svih članica klase postavlja na „nultu vrijednost“ za taj tip (0 za brojeve bilo kog tipa, null za reference). Npr. u klasi Trougao (fajl Trougao.java) imamo samo tri članice klase: tjemeA, tjemeB i tjemeC. Kako nismo napisali konstruktor, Java je obezbijedila konstruktor bez argumenata Trougao() koji sve tri članice klase postavlja na „nultu“ vrijednost za tip Tacka tj. na vrijednost null. Ako napišete konstruktor, kao u klasama TackaA i DuzA, podrazumijevani konstruktor više ne postoji (vidi klase DuzA, TackaA), osim ako ga sami ne napišete. Klasa može imati više konstruktora, ali se moraju razlikovati po broju i/ili tipu argumenata. U klasi TackaA napisali smo dva konstruktora: TackaA(double x, double y) i Tacka(double x). U klasi Duz napisali smo dva konstruktora: Duz (Tacka poc, Tacka kraj) i Duz(double ax, double ay, double bx, double by).

U klasi Trougao nismo napisali konstruktor, pa Java prevodilac sam kreira konstruktor bez argumenata Trougao().

Konstruktor kopije (copy constructor) kreira kopiju objekta klase na osnovu drugog objekta iste klase. Jedini argument konstruktora kopije je tipa kao i klasa (vidi primjer klase TackaA).

Svi objekti u programu, tj. promjenljive tipa klase, predstavljaju se referencama, kako je prikazano na slici. Obratite pažnju da reference omogućavaju da neki dijelovi memorije budu zajednički za više objekata (npr. tačka t1 je početak duži d1).

Članice i metodi klase mogu imati modifikator: public, protected ili private. Ako ne napišete modifikator, podrazumijeva se tzv. default modifikator tj. možete pristupati članici ili metodi unutar istog paketa (engl. package). O modifikatoru protected govorićemo kada budemo učili nasljeđivanje. Ako koristite private, tada se članicama ili metodima može pristupati samo iz same klase u kojoj se nalazi metod/članica. U tom slučaju, za članice klase je potrebno kreirati metode za čitanje vrijednosti (obično su oblika *getIme* pa se nazivaju geteri) i izmjenu vrijednosti članice (obično su oblika *setIme* pa se nazivaju seteri). Primjenom Eclipse-a, mogu se generisati preko stavke kontekstnog menija Source/Generate Getters and Setters...

Referenca this se pojavljuje kada generišete getere, setere i konstruktore i primjenom stavke Source iz kontekstnog menija u programu Eclipse (desni klik na fajl, stavka Source). Referenca this predstavlja referencu na trenutni objekat tj. na onaj objekat koji u tom trenutku poziva metod klase ili pristupa članici klase. U seterima i konstruktorima možete izbjeći upotrebu this ako argumente setera i konstruktora nazovete nekim drugim imenom a ne imenom članice klase.

Statički metodi klase se pozivaju ili direktno (ako se metod nalazi u istoj klasi iz koje se poziva) ili navođenjem imena klase ispred imena metoda (npr. Math.sqrt()). Nestatički metodi se pozivaju preko objekta klase (npr. d1.rastojanje() ili new Duz().rastojanje()).