Šta je Objektno orijentisano programiranje?

NATALIJA MITKOVIC

- PROCEDURALNO PROGRAMIRANJE
- OBJEKTNO ORIJENTISANO PROGRAMIRANJE

PROCEDURALNO PROGRAMIRANJE

- Rani programski jezici tzv. **Proceduralni** – programer bi definisao **skup procedura**, koje bi računar potom obrađivao.
- Proceduralno programiranje koristi listu instrukcija da kaže kompjuteru šta da radi korak po korak.
- Proceduralno programiranje se oslanja na procedure. Procedura sadrži niz koraka koje treba izvršiti.
- Proceduralni programski jezici poznati su i kao **top-down jezici**, odnosno preocedure se izvršavaju redom odozgo na dole.
- Većina ranih programskih jezika je proceduralna. Primeri proceduralnih jezika uključuju **Fortran**, **COBOL** i **C**, a pored njih tu su i **BASIC** i **Pascal**.
- Proceduralno programiranje se i dalje koristi, ali kada želite da programirate nešto izvan osnovnog niza koraka, proceduralni jezici mogu postati teški za upravljanje. Tu nastupa **objektno orijentisano programiranje.**

OBJEKTNO ORIJENTISANO PROGRAMIRANJE

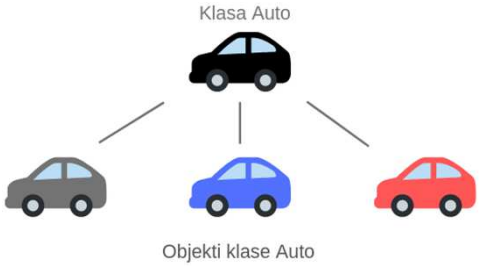
- Prvi objektno orijentisan programski jezik (Simula) – uvodi ideju o objektima. Objekti su skupovi informacija koje se tretiraju kao jedinstven entitet.
- Neki od objektno-orijentisanih programskih jezika su: **Java, Swift, Ruby, PHP, C#, C++**

- **Klase** definišu objekat: sadrže **atribute** i **metode** koje objekat treba da ima.
- Da bi se kreirao objekat, potrebno je prethodno imati klasu. Klase, dakle, predstavljaju neku vrstu šablona (uzorak, template), prema kome se kreiraju objekti => klase mogu imati više objekata.

OBJEKTNO ORIJENTISANO PROGRAMIRANJE

- Npr. možemo imatu klasu koja se zove **Auto**.
  - **Atributi** predstavljaju osobine objekta - marka automobila, boja, broj vrata itd.
  - **Metode** predstavljaju operacije koje objekat izvršava - na primeru automobila metode bi bile: kreni, zaustavi, ubrzaj i sl.

- Kada smo definisali klasu, možemo napraviti njenu instancu - odnosno **objekat**. Objekat možemo nazvati MojAuto i unutar njega navesti njegove atribute (na primer, opel, siva boja, 5 vrata). Nakon toga možemo pozvati i željenu metodu/funkciju.



Koncepti Objektno orijentisanog programiranja - Nasljeđivanje

- Nasljeđivanje je koncept u Objektno orijentisanom programiranju, koji označava da klasa može imati svoje izvedene klase - **podklase**.
- U navedenom primjeru klasa Auto (koja bi sada bila bazna klasa) može imati podklase, na primer prema vrsti vozila, koja bi mogla da ima svoje atribute i metode.
- Tako bi, klasa Auto, imala podklase - Limuzina, Karavan, Sportski automobil i sl. Objekti u podklasama bi nasljeđivali sve atribute i metode od bazne klase, kao i od izvedene klase kojoj pripadaju.

Koncepti Objektno orijentisanog programiranja - Enkapsulacija i Apstrakcija

- **Enkapsulacija** predstavlja nivoe zaštite atributa ili funkcija u okviru jedne klase. Postoje 3 nivoa zaštite:
  - **Public** - odnosno određena funkcija/atribut je dostupna bilo gdje u okviru programa, odnosno i izvan same klase u kojoj su definisani
  - **Protected** - funkcija/atribut je dostupna u okviru klase i njenih izvedenih klasa
  - **Private** - funkcija/atribut je dostupna samo u okviru klase

Koncepti Objektno orijentisanog programiranja - Enkapsulacija i Apstrakcija

- Zaštita određenih funkcija/atributa je važna, prvenstveno iz **sigurnosnih razloga**.
- Takođe, kod će biti čistiji (nije potrebno da svaka klasa u okviru programa može da koristi sve elemente drugih klasa).
- Na primjeru automobila: vozaču automobila nije potrebno znanje kako funkcioniše svaki dio: motor, menjač, brisači: automobil mora da posjeduje sve te komponente, a vozaču je potrebno da ima mogućnost da uključi brisače, pri čemu ne mora da zna na koji način oni rade.
- Ovaj proces, u kom se korisniku prikazuju samo relevantni podaci a sakrivaju nepotrebni detalji o objektu naziva se **Apstrakcija**.

Koncepti Objektno orijentisanog programiranja - **Polimorfizam**

- **Polimorfizam** u Objektno-orijentisanom programiranju predstavlja koncept prema kome se metoda ponaša različito u različitim objektima.
- U primjeru automobila, u klasi Auto, može da postoji metodu Uključi ili Isključi.
- Ista metoda se može koristiti i za Brisače i za Svjetla, pri čemu postoji razlika šta se zapravo rade.

Proceduralno vs OO programiranje

POP koristi pristup odozgo prema dole u programiranju

OOP koristi pristup odozdo prema gore u osmišljavanju programa

Proceduralno vs OO programiranje

Program je podeljen na male celine na osnovu funkcija	Program je podeljen na objekte u zavisnosti od problema
Svaka funkcija sadrži različite podatke	Svaki objekat kontroliše svoje podatke

Proceduralno vs OO programiranje

Prati sistematski pristup rešavanju problema	Fokusira se na sigurnost podataka bez obzira na algoritam
Funkcije su važnije od podataka u programu	Glavni prioritet su podaci, a ne funkcije u programu

Proceduralno vs OO programiranje

Nema jednostavnog načina za skrivanje podataka	Skrivanje podataka je moguće u OOP-u
Ne postoji koncept nasleđivanja u POP	Nasleđivanje je dozvoljeno u OOP-u