# Istorijski razvoj programskih jezika

**Programski jezik C#** je viši programski jezik opšte namene koji u potpunosti podržava paradigmu objektno orijentisanog programiranja. To je programsko okruženje koje ima ogromne biblioteke standardnih programskih modula i podržava obrade bez kojih danas ne može da se zamisli nijedan ozbiljan program. Tu spadaju komunikacija sa korisnikom preko prozora, višenitna obrada, rad preko Interneta, rad sa bazama podataka itd. Jezik C# je srodnik jezika C++ i u velikoj meri liče jedan na drugi.

Programski jezik C je nastao 1970-tih godina. Pošto su u centru pažnje **postupci (procedure) koji se primenjuju nad podacima**, ta tehnika programiranja se naziva i **PROCEDURALNO PROGRAMIRANJE**. Autor ovog programskog jezika je Denis Riči (*Dennis Ritchie*).

Početkom 1980-tih godina su računarski programi narasli do tih veličina da su se tehnike proceduralnog programiranja pokazale neefikasnim. Rešenje se našlo u paradigmi **OBJEKTNO ORIJENTISANOG PROGRAMIRANJA**, kod koje se **u centru pažnje nalaze podaci (objekti) nad kojima se primenjuju neki postupci**. Tada je najveći uspeh imala nadogradnja jezika C, poznata pod imenom C++. C++ je **hibridni jezik** pošto, pored novijeg OOP, podržava i starije proceduralno programiranje. Autor jezika C++ je Bjarn Strostrup (*Bjarne Stroustrup*).

Početkom 1990-tih godina pojavila se potreba za programima koji mogu da se izvršavaju na bilo kom računaru bilo kog proizvođača. Do tada su izvorni kodovi morali da se prevode za različite arhitekture pa je bilo neophodno da postoji veliki broj programa prevodioca.

**Prvi programski jezik koji je prenosiv sa platforme na platformu u prevedenom obliku bio je Java**. Za autora se smatra Džejms Gazling (*James Gosling*). Prva verzija jezika objavljena je 1995. godine, a od tada se svakih par godina objavljuje nova, razvijenija verzija. U početku se Java oslanjala na standard firme iz koje je potekao *Sun Micrpsystem, Inc.*, a danas iz firme *Oracle*, koja je kupila *Sun*.

**Programski jezik C#** razvijen je firmi *Microsoft* u okviru *.NET Framework*. Za autora jezika smatra se Anders Halzberg (*Anders Hejlsberg*). Kako u jeziku C++ oznaka ++ sugeriše na poboljšanje jezika C, tako je u jeziku C# oznaka # simbol za povisilicu ili 4 simbola +, odnosno unapređenje jezika C++. Prva verzija jezika C#, koja je objavljena 2000. godine, u velikoj meri je podsećala na jezik Java, ali je imao i neke nove koncepte. Počevši od verzije 2 jezika C# počelo je znatno razilaženje sa jezikom Java. Jezik C# počev od 2003. godine ima zvanični *ECMA* i *ISO/IEC* standard. Referentni prevodilac mu je *Microsoft Visual C#*.

# Uvod u objektno orijentisano programiranje

U svakom računarskom programu mogu da se uoče dve grupe elemenata: **naredbe i podaci**. Naredbe određuju šta se radi, a podaci čime se radi. Organizacija programa može da bude orijentisana ka jednoj od ove dve grupe elemenata. Klasičan stil programiranja okrenut je prema postupcima i naziva se **proceduralno programiranje**. Mana ovakvog pristupa je veliki stepen povezanosti delova složenog sistema (u slučaju kada imamo veliki broj funkcija, procedura, ...). Da bi se unele nove mogućnosti potrebno je preraditi vrlo veliki deo već gotovih programa.

Savremeni stil programiranja okrenut je prema podacima i naziva se **objektno orijentisano programiranje (OOP)**. Po tom modelu program se sastoji od **OBJEKATA** koji imaju neka moguća **STANJA** i **PONAŠANJA**. **Stanja (atributi)** predstavljaju vrednosti objekata, koje mogu da se menjaju. **Ponašanja (metode)** predstavljaju pravila menjanja atributa, reakcije na uticaje okoline i načine uticanja na okolinu. Celokupna obrada ostvaruje se u obliku međusobnih delovanja objekata u programu.

Svi objekti u programu grupišu se po svojim osobinama. Grupe objekata sa sličnim osobinama čine **KLASE**. Objekti iste klase imaju iste moguće atribute i nad njima se mogu vršiti iste metode.

*Primer pojma klase i objekata:* *Pas* (**klasa**) je životinja sa poznatim opštim osobinama i ponašanjem. *Lesi* (**objekat**) je jedan tačno određeni pas sa konkretnim osobinama (boja, težina, starost) i ponašanjem koje se uklapa u ponašanje svih pasa.

**KLASE** su tipovi podataka prema definiciji jer određuju moguće vrednosti svojih primeraka i moguće radnje nad tim primercima. Prema tome, **objekti su podaci klasnih tipova**. Za razliku od prostih tipova podataka, kao što su celi ili realni brojevi, **KLASE SU SLOŽENI TIPOVI PODATAKA**.

**STANJA (ATRIBUTI)** objekata predstavljaju se podacima unutar objekata koji se nazivaju **POLJA**. Polja mogu da budu podaci prostih ili složenih tipova. U proste tipove spadaju celi brojevi, realni brojevi itd, a u složene nizovi i klase (na primer *string* je u objektnom programiranju klasa, a ne prost tip).

**PONAŠANJE** objekta ostvaruje se postupcima unutar klasa koji se nazivaju **METODE**. Metode odgovaraju funkcijama u klasičnim programskim jezicima.

Polja i metode klasa zajedničkim imenom nazivaju se **ČLANOVI KLASA**.

**PREDNOSTI OOP** ogledaju se u tome što je pri menjanju mogućnosti programskog sistema potrebno prerađivati samo mali deo već gotovog programa. Ako se prošire mogućnosti neke klase nije potrebno promeniti i deo programa koji koristi tu klasu. Takođe, ako se u programski sistem uvode nove klase, deo programa koji koristi stare klase ne treba promeniti.

OOP zasniva se na pet osnovnih principa: **apstrakciji**, **učaurivanju**, **nasleđivanju**, **polimorfizmu** i **ponovnom korišćenju koda**.

# Modelovanje

Uočavaju se objekti koji učestvuju u rešavanju problema u realnom svetu i aktivnosti koje izvršava svaki od tih objekata.

 **Kucni savet** (sadrži podatke o stanovima, od aktivnosti ima učitavanje podataka, izbor predsednika i određivanje učesnika krečenja )

 **Stan** (sadrži podatke o jednom stanu i pruža uvid u te podatke)

**Stan**

-brojStana: String

-povrsina: Double

-imeVlasnika: String

-starostVlasnika: Integer

-radniOdnos: RadniOdnos

+getBrojStana(): String

+setBrojStana(brojStana: String)

+getImeVlasnika(): String

+setImeVlasnika(imeVlasnika: String)

+getPovrsina(): Double

+setPovrsina(povrsina: Double)

+getRadniOdnos(): RadniOdnos

+setRadniOdnos(radniOdnos: RadniOdnos)

+getStarostVlasnika(): Integer

+setStarostVlasnika(starostVlasnika: Integer)

**KucniSavet**

-stanovi: Stan[\*]

+UcitajPodatkeIzFajla(nazivFajla: String)

-SortirajPoGodistuVlasnika(): Stan

+OdrediPredsednika(): String

+OdrediMolere(): String