Šolski center Celje
Splošna in strokovna gimnazija Lava

# C in Java

Raziskovalna naloga

Mentor:
Mojmir KLOVAR, univ. dipl. inž.

Avtorja:
Staš ŽNIDAR 4.F
Jurij REPAR 4.F

Celje, marec 2006

# Kazalo

# Povzetek

Namen najine raziskovalne naloge je prikazati razlike v zgradbi jezikov C in Java in raziskati, ali je mogoč prehod iz C na Javo. Naloga vsebuje zgodovino nastanka obeh jezikov ter njune osnovne razlike in značilnosti. Večji poudarek sva naredila na izvajalnem in razvojnem okolju Jave ter na delovanju Javanskega prevajalnika. V nadaljevanju je opisana primerjava obeh jezikov na osnovnem sintaktičnem nivoju. Zaradi posebnega delovanja Jave je razložena priprava na delovanje na računalniku v okolju Windows. Za boljšo predstavo razlike obeh programskih jezikov sva napisala dva praktična programa. S tem sva hotela prikazati razlike v sintaksi in dogajanje s kodo. Ker sva hotela prikazati poleg teoretičnega prehoda jezika C na Javo, sva razdelila anketni vprašalnik 17 študentom in 3 izkušenim programerjem.

# 1. Uvod

V današnjem programerskem svetu se je zaradi sunkovitega skoka tehnologije na strojnem nivoju računalnika posledično pojavila tudi potreba po bolj izpopolnjenih, učinkovitih in večnamenskih programskih jezikih. Eden takšnih jezikov je programski jezik Java, katerega osnove sva predstavila v tej raziskovalni nalogi. Razlogov, da sva si izbrala ta programski jezik in ga primerjala že z dobro uveljavljenim in poznanim programskim jezikom C, je več. Kot dijaka tehniške gimnazije sva pri predmetu računalništva že spoznala osnove programskega jezika C in spoznala njegove praktične vrednosti, tako prednosti kot slabosti. Ker bo kmalu v učnem programu programski jezik C nasledila Java, je bil to eden izmed povodov za nastanek te raziskovalne naloge, poleg tega, pa tudi to, da se je programski jezik Java v zadnjem času zelo razširil in uveljavil prav zaradi svojih pozitivnih strani, ki jih programski jezik C ne vsebuje. Ker programski jezik Java jemlje temelje pri jeziku C, sva predvidevala, da takšen prehod iz enega programskega jezika na drugega ne bi smel biti pretežek.

## 1.1 Opis raziskovalnih metod

Kot metodo raziskovanja sva uporabila anketni vprašalnik kombiniranega tipa.

# 2. Hipoteze

Čeprav so najine izkušnje s programskim jezikom Java še slabe, lahko predvidiva, da je za prehod iz C na Javo lažji, če ima programer nekaj predznanja v jeziku C.

Predvidevava, da je bil prvi programski jezik anketirancev C. Ker bo večina anketirancev študentov, predvidevava, da bodo večinoma uporabljali brezplačna razvojna orodja. Zaradi visoke cene profesionalnih razvojnih orodij. Pri profesionalnih programerjih pa bo odstotek uporabe plačljivih razvojnih orodij višji. Sklepava, da bo privajanje na programski jezik Java trajalo dlje časa ker so se jezik učili samostojno.

# 3. Opis jezika

## 3.1 Osnove C

C je relativno preprost programski jezik Eden izmed ciljev razvoja programskega jezika je bil, da bi bil jezik prevedljiv z prevajalnikom, ki bi kodo preveril le enkrat. To je pomembno zaradi tega, ker bi tako bilo potrebno le skromno število strojnih ukazov za vsakega izmed jezikovih glavnih elementov in brez povečanega izvajalnega časa. Prevajalnik, ki kodo preveri le enkrat, je takšen ki lahko prevede izvorni program, brez potrebe, da išče nazaj po izvorni kodi. To je razlog zakaj je potrebno napisati prototip funkcije, katere klic se pojavi pred njeno definicijo. Prav mogoče je napisati programsko kodo, ki vsebuje malo izvzetih podobnosti z zbirnim jezikom, pravzaprav jeziku C včasih pravimo visokonivojski zbirni ali namizni zbirni jezik.
Deloma zaradi jezikove maloštevilne in skromne zgradbe, se lahko C prevajalniki razvijejo sorazmerno preprosto in zaradi tega se je jezik razširil in postal dosegljiv na obširnem izboru platform. Navkljub jezikovi nižje nivojski naravi, je bil razvit da omogoči in spodbuja strojno odvisno programiranje. Standardom ustrezen in prenosen C program je lahko zato preveden za zelo raznoliko paleto računalnikov.
C so originalno razvili programerji, ki so imeli v mislih za uporabnike le-tega predvsem sebe, kljub temu pa je C dosegel zelo obširno popularnost, in tudi funkcionalnost v kontekstu, daleč stran od njegovega prvotnega namena.
C vsebuje sledeče pomembne značilnosti:

- Je preprost jedrni jezik (programski jezik z vključenimi preprostimi knjižnicami) s pomembnimi uporabnimi vrednostmi, kot so matematične funkcije in upravljanje z datotekami, ki so vsebovane v knjižnicah.
- Usmerjen je na vzorec postopkovnega programiranja s pripomočki za programiranje v strukturnem načinu.
- Ima vnosni sistem, ki preprečuje nesmiselne operacije.
- Za naloge, kot so določanje makrov in vključevanje datotek z izvorno kodo vsebuje uporabo predprocesorskega jezika, C predprocesor.
- Ima nižjenivojski dostop do računalnikovega spomina z uporabo kazalcev.
- Vsebuje minimalno število besed.
- Parametre podane po vrednosti. Parametre podane po referenci lahko simuliramo z določeno podanimi prednostmi kazalcev.
- Vsebuje zapise oziroma uporabniško definirane agregatne podatkovne tipe, imenovane strukture, ki omogočajo združitev podobnih podatkov in obravnavanje teh kot celoto.

Čeprav je poleg pozitivnih značilnosti jezika C seznam manjkajočih uporabnih značilnosti dokaj obširen, je bilo to na nek način pomembno za njegovo uveljavitev. Ker je pisanje novih prevajalnikov za nove platforme hitro, dopušča programerju kontrolo nad delovanjem programa in dovoljuje rešitve, ki so najbolj naravne za vsako platformo. To je razlog, zakaj koda , napisana v C, teče veliko bolj učinkovito kot pri drugih programskih jezikih. Ponavadi le ročno nastavljena zbirniška koda teče hitreje, ker ima popolno kontrolo nad računalnikom, vendar so napredki v C prevajalnikih in nova zapletenost modernih procesorjev očitno zmanjšali to pomanjkljivost.

## 3.2 Zgodovina C

Razvoj programskega jezika C se je začel v AT&T Bell laboratorijih med letoma 1969 in 1973. Po mnenju Dennisa Ritchieja je bilo v tem obdobju najustvarjalnejše leto 1992. Programski jezik je dobil ime C, ker je bilo mnogo njegovih »sestavin« izpeljanih od njegovega predhodnika B.

Razvoj jezika C je bil rezultat želje programerjev po igri, imenovani Space Travel. To igro so igrali na glavnem računalniku podjetja, vendar sta zaradi pomanjkanja moči na računalniku in potrebe po podpori za 100 uporabnikov Thompson in Ritchie ugotovila, da nimata dovolj učinkovite kontrole nad vesoljsko ladjo, da bi se ta izogibala trkom vesoljskih kamnov. Tako sta se odločila, da bosta igro premestila na nezaseden računalnik PDP-7 v pisarni. Računalnik ni imel operacijskega sistema, zato so se dogovorili, da ga bodo napisali. Nazadnje so se odločili, da bodo prenesli operacijski sistem v pisarniški računalnik PDP-11, vendar pa je bilo to težavno, ker je bila celotna koda napisana v zbirniku. Odločili so se, da bodo uporabili višjenivojski prenosni jezik, da bi lahko bil operacijski sistem lažje prenosljiv iz enega računalnika na drugega. Pomislili so na programski jezik B, vendar je bil premalo funkcionalen, da bi lahko izrabili prednosti nekaterih naprednih značilnosti računalnika PDP-11. Iz tega je sledila odločitev, da ustvarijo boljši programski jezik, imenovan C.

Izgovor za pridobitev prvotnega računalnika, ki je bil uporabljen za razvoj operacijskega sistema Unix, je bila izdelava sistema za samodejno potrditev patentov. Prvotna različica Unixa je bila razvita v zbirnem jeziku, kasneje pa je bil programski jezik C ustvarjen za ponovno pisanje tega operacijskega sistema.

Do leta 1973 je postal programski jezik C dovolj močan, da je bila večina Unixovih jeder, originalno napisanih v zbirnem jeziku PDP-11/20, ponovno napisana v programskem jeziku C. To je bil eden prvih delujočih sistemskih jeder, podprtih z orodjem v jeziku, ki ni zbirnik. Zgodnji primerki vsebujejo Multicov sistem (napisan v PL/I) in glavni nadzorni program (Master Control Program) za Burroghov računalnik B5000, napisan v algoritemskem jeziku ALGOL v letu 1961.

V letu 1978, sta Ritchie in Kernighan izdala prvo verzijo knjige The C Programming Language. Ta knjiga, programerjem znana kot »K&R«, je mnogo let služila kot neformalna specifikacija tega jezika. Verzija jezika C, ki jo knjiga opisuje, je navadno znana kot »K&R« jezik C.
Ta verzija programskega jezika C vsebuje naslednje značilnosti jezika:

- Podatkovni tip *struct* (strukture)
- Podatkovni tip *long int*
- Podatkovni tip *unsigned int*
- Operator =+ se je spremenil v += zaradi pomenske dvoumnosti stavka i=+10, ki bi lahko pomenil i =+10 ali i = +10

K&R C je pogosto obravnavan kot najosnovnejši del jezika, za katerega je nujno, da C prevajalnik podpira. Še mnogo let po predstavitvi standarda ANSI C je bila ta verzija »najmanjši skupni imenovalec«, ki so se ga programerji držali, kadar je bila priporočljiva maksimalna prenosnost, saj še niso vsi prevajalniki podpirali standarda ANSI C. Legalen ANSI C je lahko tudi dobro napisan K&R C.

V poznih 70. letih je začel C izpodrivati takrat najmočnejši mikroračunalniški programski jezik BASIC. V 80. letih so ga sprejeli za uporabo z IBM osebnim računalnikom in takrat se je začela njegova popularnost močno povečevati. Približno v tem času pa so na podlagi

jezika C začeli v Bell Laboratories razvijati objektno orientiran programski jezik imenovan C++, ki je danes najbolj pogost uporabljen programski jezik.
Leta 1983 je ameriški nacionalni inštitut za standarde ANSI ustanovil odbor, ki bi standardiziral programski jezik C. Standard je bil sprejet leta 1989 in se je uradno imenoval ANSI X3.159-1989 »Programski jezik C«, znan tudi kot ANSI C, ali C89.
V letu 1990 je bila ta verzija jezika C, z manjšimi popravki in novim imenom ISO/IEC 9899:1990 oz. C90, sprejeta pod okrilje mednarodne organizacije za standarde ISO.
Nekateri izmed ciljev ANSI C standardizacijskega procesa so bili združitev značilnosti K&R C, nekaterih neuradnih značilnosti, uveljavljenih skozi leta, nekatere novosti, kot je na primer prototip funkcij, in zmogljivejši predprocesor. ANSI C je sedaj podprt pri skoraj vseh največ uporabljenih prevajalnikih. Po ANSI C standardizaciji so značilnosti jezika C zaradi razvoja jezika C++ ostale za nekaj časa nespremenjene. Leta 1999 pa je bil v ISO organizaciji sprejet standard, znan kot C99, z nekaterimi novimi značilnostmi. Nekateri proizvajalci, kot sta na primer Borland in Microsoft, sta zaradi večje podpore jeziku C++ manj pozornosti namenila podpori C99, vendar je večina funkcij vseeno podprta.

## 3.3 Osnove Jave

Prevajalnik razčleni izvorno kodo na osnovne člene. Osnovne dele jezika prevede v izvedljiv program. Prevajalnik obravnava vse znake po standardu Unicode v 16-bitni obliki (Java razlikuje med velikimi in malimi črkami abecede).

Programi so sestavljeni iz členov, ki jih lahko bolj na drobno razdelimo na:

- dobesedne vrednosti
- spremenljivke
- operatorje

Ti trije členi skupaj z morebitnimi klici metod tvorijo izraze. Izrazi so elementi programa, ki se izračunajo in vrnejo novo vrednost. Najmanjša enota, pri kateri izraz nastopa samostojno kot zaključena celota programske kode, se imenuje stavek. Poleg stavkov poznamo še napovedane stavke (declerations) in stavke, ki nadzirajo tok izvajanja.

V Javi lahko simbolično ime pripišemo spremenljivkam, novim podatkovnim tipom, paketom ter metodam (ali funkcijam). Dolžina simboličnega imena ni omejena, začeti se mora s črko.

Java je predmetno usmerjen programski jezik in ne pozna samostojnih podprogramov ali funkcij. Zaključena opravila morajo biti vsebovana v metode, to pa je funkcijski podprogram nekega razreda, ki določa del njegovega obnašanja.

## 3.4 Zgodovina Jave

Java platform (ime za Javino računalniško okolje) in programski jezik Java sta bila v začetku del notranjega razvoja pri Sun Microsystems v decembru leta 1990. Medtem ko je Naughton razmišljal o selitvi v podjetje NeXT, so mu ponudili možnost sodelovanja pri projektu razvijanja nove tehnologije. Najprej se je ta projekt imenoval Stealth Project, ta pa

se je kmalu preimenoval v Green Project. Naughtonu sta se pridružila še James Gosling in Mike Sheridan. Skupaj s še drugimi inženirji so začeli z delom v pisarni v Kaliforniji. Poskušali so razviti novo tehnologijo za programiranje »pametnih« naprav naslednje generacije, ki je Sunu predstavljala novo možnost.
Ekipa je imela v začetku namen uporabljati C++, vendar pa je bil kasneje zavrnjen zaradi različnih razlogov. Ker so razvijali sistem za posebno rabo z omejenimi sredstvi, so se odločili, da C++ zahteva preveč ponavljanja in da je njegova zapletenost vodila k napakam v razvoju. Pomanjkanje programskega jezika C++ pri samostojnem čiščenju pomnilnika (garbage collection) je pomenilo programerjem ročno upravljanje sistemskega spomina. Ekipo je tudi skrbelo pomanjkanje priročnih pripomočkov za varnost, distributivno programiranje in nitenje (način razdeljevanja programa na več istočasno izvajanih poslov). Hoteli so podlago, ki bi jo bilo mogoče enostavno prevesti na več različnih sistemov.

Bill Joy si je zamislil nov jezik, ki bi združeval programski jezik Mesa in C. V dokumentu, imenovanem Further je predlagal Sunu, da naj inženirji izdelajo objektno orientirano okolje, ki bi temeljilo na jeziku C++. Prvotno je Gosling začel modificirati in razširjati C++, ki ga je imenoval C++ ++ --, a ga je kasneje opustil in začel z razvojem novega programskega jezika, Oak (hrast), imenovanega po drevesu, ki je stal pred njegovo pisarno.

Poleti leta 1992 je bila skupina programerjev pripravljena predstaviti dele nove platforme, ki je vsebovala Green operacijski sistem, Oak programski jezik, knjižnice in strojno opremo. Njihov prvi poskus je bil 3. septembra leta 1992 usmerjen v izdelavo osebnega elektronskega organizatorja (PDA), imenovanega Star7, ki je imel grafični vmesnik in »pametnega pomočnika« imenovanega »Duke«, namenjenega za pomoč uporabniku. V novembru istega leta se je projekt Green »prelevil« v FirstPerson, Inc, popolnoma podrejen Sun Microsystems, in ekipa je bila na novo naseljena v Palo Alto. FirstPerson ekipa se je zanimala za gradnjo visoko interaktivnih naprav, in ko je Time Warner pokazal zanimanje za tako imenovano »set-top« napravo, ki naj bi služila sprejemu različnih televizijskih signalov (npr. iz satelitskega krožnika, ethernet kabla, itd.), se je FirstPerson preusmeril in se odzval na to s platformo za »set-top« napravo. Vendar je bilo podjetje za kabelsko televizijo mnenja, da takšna platforma daje uporabniku preveč nadzora, in FirstPerson je izgubilo proti podjetju Silicon Graphics, Inc. Uspel ni tudi posel s podjetjem The 3DO Company. Brez interesa televizijske industrije se je podjetje FirstPerson pridružilo nazaj k podjetju Sun.

V juniju in juliju leta 1994 se je ekipa preusmerila k izdelavi platform za spletno uporabo. Predvidevali so, da se bo s prihodom brskalnika Mosaic tudi internet razvil v visoko interaktivni medij, kot se je to zgodilo s kabelsko televizijo. Kot prototip so napisali majhen internetni brskalnik, imenovan WebRunner, ki so ga kasneje preimenovali v HotJava. Istega leta se je programski jezik preimenoval v Java. Ime Java je sprejelo nekaj članov skupine v bližnji kavarni. Kot zanimivost, da je Java dobila ime po izdelkih iz kavarne, je tudi dejstvo, da so prvi štirje bajti (tako imenovana magična številka) v katerekoli class datoteki v šestnajstiškem številskem sistemu 0xCAFEBABE.

Oktobra leta 1994 sta bila brskalnik HotJava in platforma Java prikazana izvršilnemu svetu podjetja Sun. Java 1.0a je bila na voljo na internetu kot download leta 1994, vendar je prva javna izdaja Jave in spletnega brskalnika HotJava izšla 23. maja leta 1995 na SunWorld konferenci. Poleg naznanila Johna Gagea, direktorja znanosti pri Sun Microsystems, je presenetila objava podpredsednika podjetja Netscape, da bodo njihovi spletni brskalniki

podpirali Javo. Januarja leta 1996 je bila ustanovljena ekipa JavaSoft, ki je bila zadolžena za nadaljnje razvijanje tehnologije, in malo kasneje je bila izdana prva verzija Jave.
Danes, po nekaj letih popularnosti Jave, je njeno mesto v spletnih brskalnikih skoraj povsem upadlo. Njeno mesto za preproste animacije je zavzel Macromedia Flash in Shockwave, Java pa je utrpela škodo tudi zaradi pomanjkanja podpore pri Microsoftu, saj Jave ne vsebuje Internet Explorer in tudi ne operacijski sistem Windows. Za razliko od tega pa je Java popularna na strežniški strani interneta, kjer veliko spletnih strani uporablja Javine strežniške strani (JavaServer pages) in ostalo tehnologijo na podlagi Jave. Število samostojnih javanskih programov na računalnikih danes narašča prav zaradi povečane zmogljivosti računalnikov, večje učinkovitosti Java Virtual Machine in kvalitete prevajalnika. Nekateri programi kot na primer NetBeans, LimeWire, Azureus itd. so danes že dodobra uveljavljeni in tudi Java Swing aplikacije se že razvijajo kot odgovor Microsoftovi .NET tehnologiji.

# 4. Značilnosti obeh jezikov

## 4.1 Izvajalno okolje Jave

Tvorci Jave so se ozrli na raziskave navideznih strojev (virtual machines) in ta koncept uspešno uporabili.
Vendar se program v vmesni kodi lahko izvaja na poljubni napravi, za katero obstaja ustrezen tolmač. S tem je osnovni pogoj za obljubo »Napiši enkrat, izvajaj povsod« izpolnjen.
Programska koda v Javi se ne prevaja v strojno, ampak v vmesno kodo. Na običajnih procesorjih je ta koda neuporabna za izvajanje, ker potrebuje tolmača, vendar pa z ustreznim tolmačem lahko to kodo izvajamo na poljubni napravi. Javino izvajalno okolje, ki se imenuje tudi Javanski stroj (Java virtual machine), je izvajalno okolje, ki poleg funkcije tolmača vmesne kode za izvajanje na več strojnih podlagah ponuja tudi nadzorovano izvajalno okolje, ki skrbi za nekatere temeljne storitve, poleg tega pa ponuja tudi visoko stopnjo varnosti. S tem se pojavi kompromis, saj se Java, ki se sproti tolmači in preverja, zelo težko primerja s strojno kodo, ki teče neposredno v procesorju.
Programski jezik Java je bil razvit iz jezikov C in C++ in je zato znan programerjem. Kljub svoji podobnosti programskemu jeziku C nima problematičnih elementov, kar omogoča razvoj zanesljivih vsestranskih programov. To omogoča velika programska knjižnica, ki je že sestavni del jezika. V Javi se ne ponavljajo kritični elementi programskih jezikov C in C++, kot so kazalci, ročno sproščanje pomnilnika, večkratno dedovanja, preobleganje operatorjev, izpušča tudi strukture in unije, kar pa nadomesti z razredi. Izključena sta tudi stavka *define* in stavek *go to*. Neodvisnost Jave se kaže v možnosti izbire izvajalnega okolja, razvojnih orodij, dodatnih knjižnic.
Ker je Java namenjena uporabi v različnih sistemih, je varnost zelo pomembna. Najpomembnejšo vlogo pri tem prevzame javanski izvajalni sistem, ki za varnost skrbi skozi ves potek nalaganja in izvajanja kode.
Izvajanje kode v nadzorovanim okolju je danes splošno sprejeta prihodnost programiranja (uspešni javi se je pridružilo še Microsoftovo orodje .net). Na srečo se je prilagodljiva Java izvrstno znašla tudi v internetu, v njej razvit spletni brskalnik (Hotjava) pa je prepričal vodstvo podjetja Sun, da so delo razvojne ekipe ponudili kot lasten izdelek.

Javino izvajalno okolje, imenovano tudi javanski navidezni stroj (JVM, Java Virtual Machine), je postalo mnogo več kot tolmač vmesne kode, ki omogoča programom, da se izvajajo na različnih strojnih podlagah. JVM ponuja nadzorovano izvajalno okolje, ki skrbi za nekatere temeljne storitve (npr. samodejno sproščanje pomnilnika), hkrati pa ponuja visoko stopnjo varnosti. Prekoračitev medpomnilnika (buffer overflow), pogosta varnostna luknja v sodobni programski opremi, v Javi preprosto ni mogoča.

Številne prednosti nadzorovanega okolja imajo seveda svojo ceno. Vmesna koda, ki jo je treba tolmačiti, hkrati pa se kar naprej preverja zaradi varnosti, se ne more primerjati s strojno kodo, ki neovirano teče neposredno v procesorju.
Prve različice Jave so bile precej neučinkovite in neprimerne za bolj zapletene programe. Delno rešitev z učinkovitostjo je prineslo pravočasno prevajanje (JIT, Just In Time complication). JIT vmesno kodo pred prvim izvajanjem še kar enkrat prevede, tokrat v strojno kodo. Hitrost se izboljša, vendar JIT potrebuje kar nekaj časa, da opravi svoje delo, kar povzroči neprijetne zastoje ob prvem zagonu.

## 4.2 Razvojno okolje Jave

Ker je Java neodvisna od platforme, je nekoliko posebna tudi pri razvojnem okolju. Če hoče programer napisati program na standardni različici platforme J2SE, ne potrebuje ničesar drugega kot razvojni komplet (JDK, Java Development Kit) in preprostega urejevalnika besedila. Razvojni komplet lahko za nekatere najbolj priljubljene operacijske sisteme (Windows, Linux, Solaris) najdemo na spletnih straneh java.sun.com. Nekateri operacijski sistemi pa razvojni komplet za Javo že vsebujejo (MacOS X), nekateri ponudniki strojne opreme pa zagotavljajo podporo za razvoj in izvajanje programske opreme v Javi v praktično vseh računalniških sistemih (IBM). Zaradi splošne priljubljenosti jezika, ki ga podpira zelo veliko ponudnikov strojne in programske opreme, so razvojna orodja za Javo zelo priljubljena za razvoj v računalniški industriji.

Nekatera razvojna okolja, ki se danes pogosto uporabljajo so:

**Visual J#**
Microsoftovo razvojno okolje za Javo. Ponuja nam delo na Microsoftovih podlagah. Na voljo je osnovna brezplačna različica Express, malce bolj napredna plačljiva Standard in še dve poklicni različici Professional in Team System.

**Borland JBuilder 2006**
Borlandova ponudba razvojnih orodij je sestavljena iz brezplačne različice, imenovane Foundation, naprednejše Developer in najbolj zmogljive Enterprise.

**Eclipse**
Najbolj znano brezplačno razvojno okolje, razvito v Javi, je Eclipse. To razvojno okolje je odprtokodno in namenjeno razvijanju nevtralnih, odprtih programerskih okolij in ogrodij. Za Eclipse je na svetovnem spletu na voljo veliko dodatkov, tako plačljivih kot zastonjskih, zato se da osnovno verzijo dodobra razširiti. Uporabniški vmesnik ne uporablja standardne knjižnice Swing, ampak IBM-ovo alternativo, knjižnico SWT. Ker je Eclipse le ogrodje, je zelo pomemben standard, s katerim se razvijajo dodatki zanj, ki pa spet ni združljiv s standardom za programerska orodja, ki ga je vzpostavil Sun s svojim orodjem NetBeans.

Eclipse deluje v okoljih Windows, Linux, Solaris 8, AIX, HP-UX, Mac OSX. IBM uporablja Eclipse tudi za osnovo drugih orodij, predvsem za IBM WebSphere Studio.

**NetBeans**
NetBeans je tako kot Eclipse zastonjsko odprtokodno orodje za razvijanje v Javi, podprto s strani Sun Microsystems. NetBeans je namenjeno za razvoj javanskih aplikacij, spletnih storitev in mobilnih aplikacij. Na voljo so tudi dodatki, ki omogočajo uporabo drugih programskih jezikov in tehnologij, npr. C, C++, Fortran, XML, Java Server Pages itd.

## 4.3 Razvojno okolje C

Razvojna okolja za programski jezik C so zaradi njegovega dolgoletnega obstoja na voljo že v najrazličnejših verzijah. Verjetno najbolj znana podjetja, ki ponujajo razvojna okolja za C, so na primer Microsoft in Borland, pri katerih imamo na voljo razvijalna okolja tako za začetnike kot tudi tista za profesionalno uporabo. Seveda je temu primerna tudi cena, saj so tako izpopolnjena orodja zelo draga. Ker midva jezik C šele odkrivava, sva uporabila tudi najinemu znanju primerno razvojno orodje. **DevC++** je brezplačen program, ki ga lahko povlečete z interneta in je tudi dokaj razširjen, zlasti med začetniki v tem programskem jeziku. DevC++ vsebuje vse potrebno za začetek programiranja v jeziku C in je tudi dokaj enostaven za uporabo.

## 4.4 Delovanje prevajalnika

Pri Javi veliko vlogo odigra prevajalnik, saj nas opozarja na napake. Prevajalnik kodo prevede do vmesne in vidimo lahko, da se poleg datoteke **.java* pojavi tudi prevedena datoteka **.class*. Slednja datoteka se pojavi, če v izvorni kodi ni napak.
Prevajalnik izvorno kodo razčleni na osnovne člene, zanemari vse komentarje, nadomesti ubežne kode in izloči prazen prostor, ki ni del konstantnih nizov. Tako mu preostanejo osnovni deli jezika (členi), ki jih prevede v izvedljiv program. Posebnost prevajanja Jave je, da prevajalnik znake obravnava po standardu Unicode v 16-bitni obliki. Java tudi razlikuje med velikimi in malimi črkami.

Pri jeziku C se za prevajanje izvorne kode uporablja enoprehodni (one-pass) prevajalnik. Prednosti takšnega prevajalnika so njegova hitrost delovanja in enostavnost izdelave. Izvorna koda se pri prevajalniku za C najprej prevede v objektno kodo. Linker oz. združevalnik objektno kodo združi v izvršilno kodo (to je datoteka z končnico *.exe*).

## 4.5 Primerjava programskega jezika Java z jezikom C

V tem poglavju so opisane podobnosti in razlike obeh programskih jezikov.

### 4.5.1 Podatkovni tipi

V programskem jeziku Java je preverjanje podatkov strožje kot v jeziku C.
Podatkovni tipi v C so precej podobni podatkovnim tipom v Javi. Tako C kot Java imata podatkovne tipe int in double, vendar je v programskem jeziku C obseg številskih tipov, kot

je int, strojno odvisen. Na 16-bitnih sistemih kot DOS ali Windows 3.x na osebnem računalniku PC je tip intiger dvobajten in ima številski obseg bolj omejen kot štiribajtni Javin celoštevilski int tip. Na teh računalnikih moramo tip int nadomestiti s tipom, ki ima večji obseg, na primer double. Znakovni podatkovni tip *char* vsebuje v Javi znake, zapisane po standardu Unicode, C pa uporablja standard ASCII.

### *Polja*

Polja (arrays) so zbirka več podatkov istega tipa. Z vsemi elementi, ki so zapisani v polju nekega podatkovnega tipa, lahko počnemo enako kot z vsako spremenljivko tega tipa.

Čeprav so polja pravzaprav predmeti, jih Javanski prevajalnik obravnava na poseben način, zato je njihova uporaba nekoliko drugačna, kot v jeziku C. Njihova predmetna narava omogoča precej lažje in bolj varno programiranje. Obseg polja je vedno znan, prekoračitev pa v Javi ni dopustna in jo izvajalni sistem prepreči. C prevajalnik meje ne preverja, za to je odgovoren programer sam.

Ker je Java močno tipizirana, je v polje mogoče shraniti le podatke vnaprej predpisanega tipa. Polja lahko shranjujejo podatke enega tipa, tipi polj pa so lahko primitivni in sklicni. Pri delu s polji najprej napovemo spremenljivko ustreznega tipa, potem pa zasežemo ustrezen pomnilniški prostor. Oboje se lahko zgodi v enem ali dveh programskih stavkih. Pri napovedi spremenljivke, ki bo hranila polje podatkov, uporabimo oglate oklepaje, ki jih lahko pišemo bodisi po simboličnem imenu ali po sami ključni besedi.
Polja v Javi so vedno enodimenzionalna. Večdimenzionalna polja »ponaredimo« tako, da vsakemu elementu polja priredimo novo polje. Toda ni potrebno, da so vsa polju prirejena polja enakih dimenzij. Java pozna operator *new*, ki izdela nov izvod želenega predmeta ali polja in vrne njegov naslov. Indeksi vseh polj v javi se začnejo z 0. Ker so polja svojevrstni predmeti, poznajo lastnost *length*, ki hrani njihovo lastnost.

Primer polja v C:

*int polje[20]*

Primer polja v Javi:

*String[] pozdrav*
Tukaj smo le napovedali spremenljivko, ki bo hranila naslov bodočega polja
*float[][] matrika*

- Zasežemo določeno pomnilniško lokacijo, ki bo polje dejansko shranjevala. Polje pripravimo za delo z operatorjem *new* in s prireditvenim stavkom.

*polje = new int [20]*

Operator n*ew* naredi nov izvod predmeta ali polja in vrne njegov naslov.

Posebnost Jave – metode

Polja so v Javi zelo uporabna zaradi nekaterih metod. Polja primitivnih tipov lahko uredimo s pomočjo statične metode *sort( )*. Metoda je izdelana kot prilagojeni algoritem QuickSort, ki je najbolj učinkovit na predhodno neurejenih podatkih.

***Nizi***

Podatkovni tip za uporabo nizov tako kot v C tudi v Javi imenuje *string*, vendar z nekaj razlikami. Tako kot pri znakovnem podatkovnem tipu *char* se podatki v niz shranijo po standardu Unicode, v C pa po standardu ASCII. Nize si predstavljamo kot polje znakov tipa *char[]*. Nizi so v Javi nespremenljivi, v C pa jih lahko poljubno spreminjamo. Nizi so v obeh programskih jezikih združljivi samo z istim tipom, torej nizom, ne moremo pa jih združevati z drugimi tipi. Zaradi pogoste uporabe nizov jih javanski prevajalnik tako kot polja obravnava na poseben način. Prevajalnik vsak niz samodejno pretvori v predmet tipa *String*, to pa pomeni, da lahko programer namesto katerega koli tipa String uporabi tudi konstanten niz znakov, celo nad konstantnim nizom lahko uporabimo metode tega razreda:

*int len = » Dolžina niza ?«.length( );*

Zaradi samodejne pretvorbe lahko nove izvode razreda String programer ustvari zgolj s podajanjem konstantnega niza. Za primerjavo nizov med seboj uporabljamo tako pri Javi kot pri C relacijske operatorje. V jeziku C so to <, >, >=, <=, ==, *!*=. Prvi štirje našteti operatorji so dejansko bolj ustrezni kot *equals* in *compareTo,* ki se pojavljajo v jeziku Java.

### 4.5.2 Spremenljivke in konstante

V Javi so lokalne spremenljivke definirane enako kot v C. Na primer: *int n = 5*. Kljub temu pa obstaja razlika. C prevajalnik ne preverja, ali so vse lokalne spremenljivke inicializirane, preden so prebrane, medtem ko Java prevajalnik (Javac) to preverja. Ta lastnost prevajalnika se lahko izkaže pri odpravi programskih napak, saj nedefinirana spremenljivka naredi v pomnilniku naključen vzorec, kar lahko posledično pomeni napako.

Java ima lahko, prav tako kot C statične (globalne) spremenljivke in podatkovna polja, vendar pri Javi le-ta ne smejo biti deklarirana zunaj funkcije oz. razreda. V Javi so se temu izognili zaradi dejstva, da je z globalnimi spremenljivkami težko upravljati. Primer konstante v jeziku C je: *const float STEVILO_PI = 3,14.*

Java obravnava konstante na svoj način. Konstante so pravzaprav »dokončne« spremenljivke. Če pred najavo spremenljivke navedemo besedo *final*, potem ta vrednost v nadaljevanju ostane nespremenjena. Primer takšnega stavka je: *final float STEVILO_PI = 3,14F*.

### 4.5.3 Pogojna stavka

Pogojna stavka omogočata razvejevanje izvajanja (branch) glede na pogoj. Naslednji programski stavek je izbran glede na izid logičnega izraza. Oba programska jezika nam dajeta na voljo stavek *if,* ki je sposoben dvosmerne odločitve, in stavek *switch*, ki omogoča večsmerne odločitve.

Stavek *if* omogoča pogojno izvajanje določenega stavka. Po programerjevi potrebi lahko s besedo *else* predpišemo še nadomesten stavek, ki se izvede, če je podani izraz neresničen.

S stavkom *switch* pregledno zapišemo večsmerno odločitev, ki jo lahko dosežemo tudi z več gnezdenimi stavki if. Stavek začne ključna beseda *switch*, pri kateri zapišemo izraz. Ta se mora izvrednotiti v znak ali celo število, vendar ne v tip *long*. V stavčnem bloku zapišemo poljubno število vrednosti izraza, na katerega se želimo odzvati z ustreznim stavkom ali stavki. Na koncu lahko, če želimo, s ključno besedo *default* določimo privzete stavke, ki se izvršijo, če ni bila izračunana nobena od naštetih vrednosti.

### 4.5.4 Krmilni stavki

***Zančni stavki***

Zančni stavki so v obeh programskih jezikih trije. To so *while*, *do while* in *for.*

Pri *while* zanki se logični pogoj vrednoti na začetku zanke, zato je mogoče, da se ustrezni stavki ne bodo nikoli izvršili, če je pogoja na začetku neresničen. Mogoče je tudi (pri nespretnem pisanju pogoja), da naredimo neskončno zanko, ki se nikoli ne izteče. Zaradi tolmačenja javanske kode so neskončne zanke praviloma nenevarne in jih lahko prekinemo z zaustavitvijo izvajalnega sistema.

Pri zanki *do while* se logični pogoj vrednoti na koncu, zato se zančno telo zagotovo izvede vsaj enkrat.

Zanka *for* je izpopolnjena oblika zanke *while*. Pri zanki for poleg pogoja za izvajanje zanke podamo tudi stavek, ki se izvede na začetku zanke, in stavek, ki se izvede po vsakem koraku zanke.
Pogoj mora biti logični izraz, preostali stavki pa so lahko kakršni koli programski stavki. Zanka *for* je še posebej primerna za delo s števci. Števec je spremenljivka, ki v vsakem koraku zanke spremeni svojo vrednost.

### 4.5.5 Funkcije

Funkcije morajo biti v Javi primeri postopka ali statična funkcija razreda, pri jeziku C pa so funkcije lahko definirane enako, lahko pa tudi niso del kateregakoli razreda. Takšnim funkcijam pravimo v C globalne funkcije in so deklarirane na naslednji način:

```
int main()
{ . . .
}
```

Tako v Javi kot v C so rezultati funkcije podani po vrednosti, vendar se v C pojavi tudi podajanje rezultatov po referenci, česar pri Javi ni mogoče uporabiti. Primer takšnega klica po referenci v C je sledeč:

```
void funkcija (int &a, float &b)
{ . . .
}
```

Glavna vrednost prenosa parametrov po referenci je v tem, da je vsaka sprememba spremenljivke v funkciji vidna tudi v glavnem programu oz. globalni funkciji.

### 4.5.6 Vhod in izhod

V jeziku C je standardni vhod in izhod predstavljen s stavkoma *cin* in *cout*. Uporabljamo << operator za izpis izhodnega stavka.

```
cout << "Pozdravljeni!" ;
```

Izpišemo lahko tudi več predmetov:

```
cout << "Odgovor je: " << x << "\n";
```

Za branje številk ali besed z vhodnim stavkom uporabljamo >>

```
int x;
cout << "Vpišite x: ";
cin >> x;
```

### 4.5.7 Kazalci

V programskem jeziku C spremenljivke vsebujejo vrednosti. Java se v tem od jezika C razlikuje, saj vsaka spremenljivka vsebuje naslov, kje se nahaja vrednost te spremenljivke. V jeziku C lahko enak rezultat dosežemo z uporabo kazalcev. Če spremenljivka *k* vsebuje določeno vrednost katerega koli tipa, potem je **k* kazalec, ki kaže na to vrednost. Druga možnost je, da spremenljivki priredimo naslov v pomnilniku, kar storimo takole: *&a = 1000*.
Kot je bilo že omenjeno, Java vsebuje samodejno upravljanje s pomnilnikom, tako imenovan »garbage collector«, ki samodejno upravlja z objekti, ki niso več potrebni. V programskem jeziku C pa mora to ročno narediti programer. Če pozabimo izbrisati objekt, lahko postopoma potrošimo celoten pomnilnik, če pa še naprej uporabljamo objekt, ki je bil že izbrisan, pa se lahko zgodi, da prepišemo podatke, ki niso namenjeni nam. S tem se pojavijo napake, ki jih je težko zaslediti in popraviti, zato je priporočljivo, da zmanjšamo uporabo kazalcev, če programiramo v jeziku C.

## 4.6 Kako pripraviti Javo za delovanje na računalniku

Prva stvar, ki jo moramo storiti, je, da nastavimo Javino razvijalno (developer's kit) orodje JDK+.- na računalnik. To orodje najdemo na Sun Java spletni strani.
Ko smo prenesli JDK, moramo slediti navodilom. Obstajata dva načina, kako jo zagnati. Osnovni način je, da kliknemo »Start« ikono na spodnjem delu Windows okolja, kliknemo »Zaženi« in napišemo »CMD«.
Če si nismo pripravili poti, potem bomo morali poiskati binarni imenik (bin directory).
Če smo pripravili pot, potem lahko zaženemo Javo v kateri koli mapi hočemo. Potem napišemo »edit Hello.java«. Edit odpre preprost pisalni program (editor) in ime našega programa je »Hello.java«.
Vsak Javin razred ima Javino razširjenost.
Če na primer napišemo preprosto kodo, ki nam prikaže na ekranu »Dobrodošli v programiranju«, shranimo, zapremo in potem napišemo »javac Hello.java«.
Javac je Javin prevajalnik in potem računalnik prevede (prevede iz besedila, ki smo ga napisali v jezik, ki je računalniku poznan) program.
Če bi pogledali v našo mapo, bi videli nov razred Hello.class. Ko smo ga prevedli v Hello.java, je prevajalnik ustvaril Hello.class, ki je računalniku razumljiv jezik. Če bi ga poskušali brati, ga ne bi znali, ker ga lahko samo računalnik.
Če napišemo java.Hello, ki pove računalniku, da zažene razred, imenovan Hello.
Paziti moramo, ker je Java razlikuje med velikimi in malimi črkami. Če imamo razred imenovan Hello in potem napišemo java.hello, program ne bo deloval.

# 5. Primer programa v C in Javi

Za primerjavo programov v obeh programskih jezikih sva napisala dva programa. Program, v katerega mora uporabnik vnesti 10 števil in dobi izpis koliko je števil, ki so deljiva s številom 2. Program sva napisala v razvojnem okolju Netbeans za Javo in v DevC++ za C.

**Java**

```
public class program  {
  public static void main(String[] args) {

    for (int i=1;i<10;i++) {
      if (i % 2 = 0)
       System.out.println (i + " ");}
        System.out.println ("Bla");
  }
}
```

**C**

```
int main()
{int i, polje[10], st=0;
for(i=0;i<10;i++)
  {cin>>polje[i];
   if(polje[i]%2==0) st++;
  }
cout<<st;
return 0;
}
```

Ker se programi v Javi ne izvajajo neposredno v procesorju, ker niso prevedeni v strojno kodo, za njihovo izvajanje poskrbi izvajalno okolje. Ta kodo naloži ter razvrsti v pomnilnik in jo preveri. Šele nato jo prilagodi procesorju in jo izvede. Pri jeziku C pa se, kot že opisano v delovanju prevajalnika, izvorna koda prevede v objektno in se nato združi v izvršilno, ki jo lahko na našem računalniku poženemo.

# 6. Anketa

Sva dijaka tehniške gimnazije in delava raziskovalno nalogo C in Java. Pod tem naslovom je mišljen pretežno prehod iz C na Javo in razlike v sintaksi. Prosiva Vas, da na vprašalnik odgovorite objektivno.

1. V katerem jeziku ste se najprej naučili programirati?
   a) C
   b) Java

2. Ali menite, da je zaradi podobnosti obeh programskih jezikov prehod iz enega na drugega lažji? Obkroži ustrezni odgovor in utemelji.

   a) Da
   b) Ne

3. Koliko časa že programirate v katerem koli programskem jeziku. Navedite katerega najbolj obvladate?

4. Kakšen se vam je zdel prehod iz C v Javo (ali obratno)?

   a) lahek ker sem poznal že druge programske jezike
   b) srednje lahek, ker sem imel malo predznanja
   c) težak, ker prej nisem nič programiral

5. Katero razvojno okolje uporabljate?

6. Uporabljate brezplačna okolja ali profesionalna plačljiva okolja. Navedite razloge.

7. Koliko časa ste se privajali na poznavanje elementov novega jezika?

   Sintakso:

   Poznavanje standardnih knjižnic:

   Učinkovito programiranje:

## 6.1 Analiza ankete

Zaradi programerske neizkušenosti obeh jezikov sva se odločila, da bova anketo razdelila študentom na Fakulteti za računalništvo. Poskušala sva najti tudi starejše izkušene programerje. Ker sva želela, da bi bili odgovori čim bolj objektivni in zanesljivi, sva anketo razdelila zadnjim letnikom na tej fakulteti in trem programerjem v dveh privatnih podjetjih. Anketo sva razdelila 30 študentom, vendar se je zaradi pomanjkanja znanja oz. nezainteresiranosti za programski jeziku Java nanjo odzvalo samo 17 študentov.

*1. Programerjev prvi programski jezik*

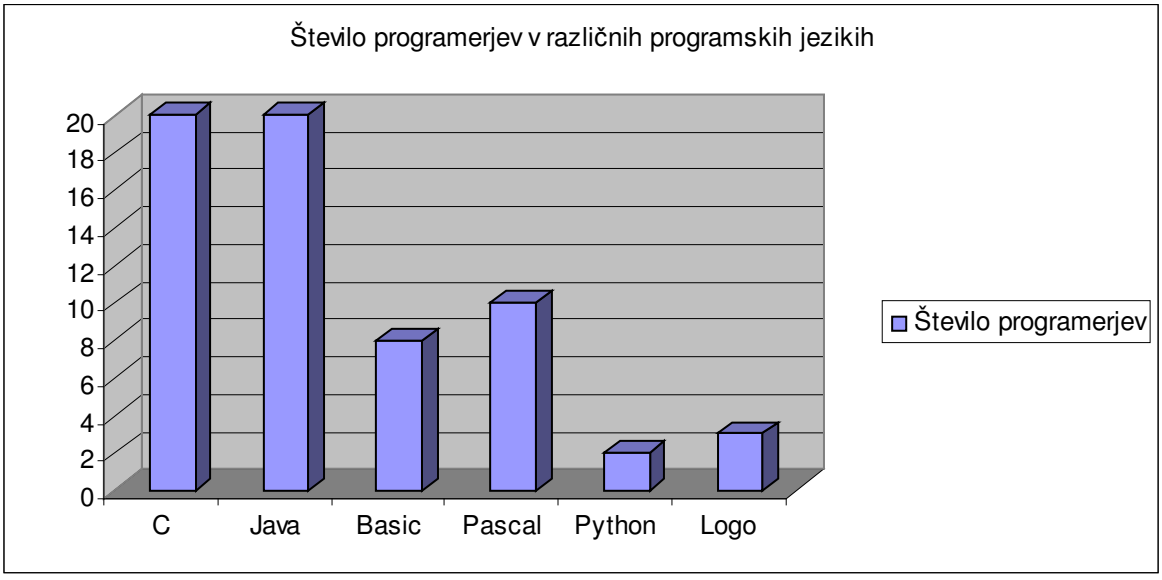
Vseh 20 anketirancev je najprej programiralo v C. Razlog za to je očiten, ker je C nastal pred Javo.

*2. Zaradi podobnosti jezikov sledi lažji prehod.*

15 anketirancev je na to odgovorilo z da (75 %), 5 z ne (25 %).

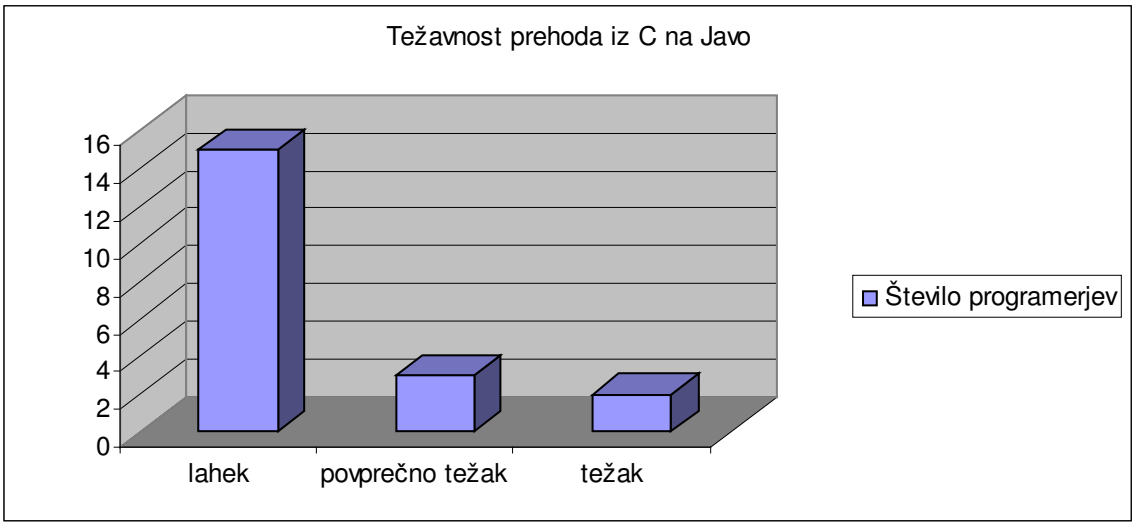
*3. Programerska izkušenost*

Zaradi splošne priljubljenosti programskega jezika C je največ anketirancev začelo programirati v tem jeziku. Najini anketiranci imajo znanje o C in Javi, poleg tega pa še 10 v Pascalu, 2 v Pythonu, 8 v Basicu in 3 v Logo.




*4. Težavnost prehoda med C in Javo*

Zaradi splošnega poznavanja jezika C za večino programerjev prehod ni bil problematičen. Za ostale anketirance, ki niso imeli znanja iz jezika C, pa je bil ta prehod težji.

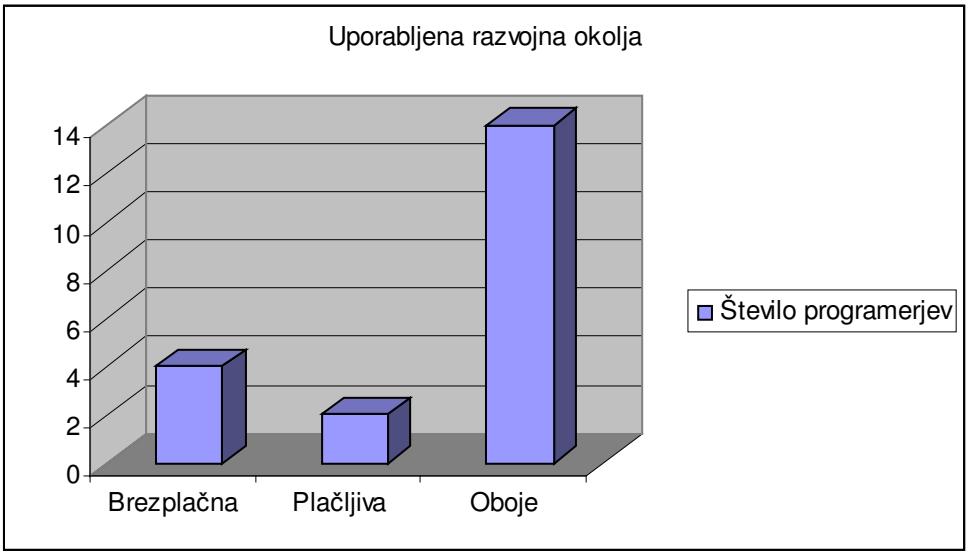



*5. Uporabljena razvojna okolja*

Anketiranci so odgovorili, da uporabljajo Eclipse, NetBeans, MS Visual Studio, Borland C++ Builder in IntelliJ IDEA.

*6. Brezplačna, plačljiva razvojna okolja ali oboje*

Programerji, ki se ukvarjajo s programiranjem neprofesionalno (za študijske namene) večinoma uporabljajo brezplačna razvojna okolja. Tisti, ki so zaposleni kot programerji v raznih podjetjih, pa uporabljajo licenčna razvojna orodja, kajti podjetja le-ta morajo imeti.




*7. Privajanje na programski jezik Java*

Privajanje na programski jezik ima več faz. Najprej mora programer spoznati osnove programskega jezika, kot je sintaksa. Sledi poznavanje osnovnih programskih knjižnic, ki traja dlje časa kot prva faza. Zadnja faza poznavanja pa je praktično programiranje, ki traja najdlje, saj učenje poteka na dejanskih projektih, ki jih uresničujemo kot programerji. Najini anketiranci so potrebovali za učenje sintakse programskega jezika Java v povprečju

en teden. Za dobro programiranje v povprečju pol leta in za kvalitetno programiranje tudi več kot eno leto. To je razumljiv podatek, kajti vsi anketiranci so se Javo učili sami, s pomočjo programerskih priročnikov in drugih programerjev. Rezultat hitrosti učenja bi bil drugačen, če bi Javo spoznavali kot učni predmet v šoli ali na fakulteti.

# 7. Zaključek

Ključno vprašanje najine raziskovalne naloge je bilo raziskati ali je možen prehod iz jezika C v Javo. Za rešitev tega problema sva morala najprej poiskati osnove obeh programskih jezikov in v osnovi spoznati programski jezik Java. Zaradi podobnosti obeh jezikov pri spoznavanju Jave nisva imela večjih težav, če pa so se pojavile, sva jih rešila s pomočjo literature in mentorja. Najbolj so naju zanimala mnenja programerjev, ki sva jih pridobila s pomočjo anketnega vprašalnika. Najina glavna hipoteza glede težavnosti prehoda se je z anketnim vprašalnikom potrdila. To dejstvo je razumljivo, kajti C je nastal pred Javo in je zanjo osnova. Večina programerjev uporablja plačljiva in zastonjska programska okolja, kar je nenavadno, kajti večina anketirancev je bilo študentov. To dejstvo se ni skladalo z najinimi hipotezami, vendar so anketiranci pojasnili, da so poleg študija zaposleni in da zaradi programerskega dela dobijo licenčna plačljiva razvojna orodja. Programerska izkušenost najinih anketirancev se kaže tudi v poznavanju drugih programskih jezikov. S to raziskovalno nalogo sva dokazala, da prehod iz jezika C na Javo ni težak, vendar mora imeti programer osnovno predznanje jezika C.

Zahvalila bi se rada vsem anketirancem za njihovo pomoč pri anketi.

# 8. Zahvala

V veliko pomoč nama je bil najin mentor, ki naju je pri nastajanju raziskovalne naloge usmerjal in nama svetoval.

# 9. Viri in literatura


LOKAR, M. (2000). Prvi programi v programski jezik C. Ljubljana: DMFA – založništvo.

MESOJEDEC U. in B. FABJAN. (2004). Java2 temelji programiranja. Ljubljana: Pasadena.

Java programming language. [Online]. [Citirano 7.1.2006; 18.15]. Dostopno na spletnem naslovu: http://en.wikipedia.org/wiki/Java_programming_language.

C programming language. [Online]. [Citirano 8.1.2006; 12.30]. Dostopno na spletnem naslovu: http://en.wikipedia.org/wiki/C_Programming_Language_language.

GORENBURG M. (1998). Similarities and Differences Between C++ and Java. [Online]. [Citirano 8.1.2006; 14.10]. Dostopno na spletnem naslovu: http://acc6.its.brooklyn.cuny.edu/~cis22/java/jvcpp.html.

Moving from Java to C++. [Online]. [Citirano 9.1.2006; 9.30]. Dostopno na spletnem naslovu: http://www.horstmann.com/ccj2/ccjapp3.html.

How to set up Java on my computer. [Online]. [Citirano 12.2.2006; 14.30]. Dostopno na spletnem naslovu: http://www.apl.jhu.edu/~hall/java/beginner/settingup.html.

MESOJEDEC U. (2006). Sveža razvojna orodja.[Online]. [Citirano 20.2.2006; 16.45]. Dostopno na spletnem naslovu: http://monitor.si/clanki.php?id=1179.