Projekat iz predmeta “Programski prevodioci 1”

Izveštaj projekta - nivo A

Student: Stefan Teslić 2017/0124
Profesor: dr Dragan Bojić
Asistenti: ms Maja Vukasović, ms Kristijan Žiža

Opis projekta

Cilj projekta je izrada malog, ali sasvim funkcionalnog kompajlera (programskog prevodioca) za jezik *Microjava*. Ovom prilikom izrađen je nivo A projekta koji obuhvata pojednostavljen skup elemenata proceduralnog programskog jezika.

Za nivo A je neophodno bilo da se omogući deklarisanje globalnih promenljivih i definisanje funkcije *main* koja može da prihvati i lokalne parametre. Neophodno je bilo i omogućiti dodelu, inkrementiranje i dekrementiranje. Neophodno je bilo i omogućiti rad sa funkcijama *print* i *read*. Treba *da se* omoguci I rad sa ternarnim operatorom koji prihvata pojednostavljen uslov bez operatora za logičko “i” i “ili”. Rad sa nizovima je takođe neophodno implementirati, ali nizovi su isključivo od prostih tipova, dakle *int*, *char* i *bool*. Neophodno je bilo i omogućiti rad sa aritmetičkim izrazima.

Kompletna specifikacija jezika za nivo A je:

```
Statement := DesignatorStatement ";".
DesignatorStatement := Designator "=" Expr.
DesignatorStatement := Designator "++".
DesignatorStatement := Designator "--".
Statement := DesignatorStatement ";".
Statement := "read" "(" Designator ")" ";".
Statement := "print" "(" Expr [“,” numConst] ")" ";".
Expr := ["-"] Term {Addop Term}

| CondFact "?" Expr ":" Expr.

Term := Factor {Mulop Factor}.

Factor :=Designator | numConst | charConst | "(" Expr ")" |
boolConst | "new" Type "[" Expr "]". Designator := ident
[ "[" Expr "]" ].
Addop := "+" | "-" .
Mulop := "*" | "/" | "%".
```

Korišćeni alati prilikom realizacije projekta

Kompajler kao takav je realizovan korišćenjem pomoćnih alata uz pomoć kojih su realizovane leksička analiza, sintaksna analiza, tabela simbola i generisanje koda.

Za realizaciju leksičke analize, korišćen je alat ***JFlex***. Alat, na osnovu *.flex* specifikacije generiše ***Yylex*** klasu koja vrši spomenutu analizu.

Za realizaciju sintaksne analize, korišćen je ***CUP*** alat uz proširenje za rad sa apstraktnim sintaksnim stablom (***CUP-AST***). Ovaj alat, na osnovu *.cup* specifikacije generiše klasu ***MJParser*** koja vrši sintaksnu analizu. Generiše se takođe i klasa ***sym*** koja kodira sve tokene od leksičkog analizatora. Generiše se potom folder ast u kojem se nalaze sve klase neophodne za generisanje apstraktnog sintaksnog stabla.

Semantička analiza je odrađena uz pomoć biblioteke ***symboltable*** koja predstavlja implementaciju tabele simbola i uz pomoć pomoćnih alata iz ast foldera. Implementiran je ***SemanticAnalyzer*** koji nasleđuje klasu VisitorAdaptor i koji predstavlja implementaciju projektnog uzorka posetilac gde se prolazi kroz apstraktno sintaksno stablo i vrši se provera kontekstnih pravila kako i popunjavanje tabele simbola. Obilazak stabla se vrši u postorder redosledu.

Za realizaciju generisanja koda koristila se pomoćna klasa ***Code*** koja predstavlja neophodni interfejs za generisanje koda.

Biblioteka ***mj-runtime*** predstavlja realizaciju Microjava virtualne mašine. Unutar ove biblioteke se nalazi i gorespomenuta klasa Code.

Za implementaciju log-a, korišćena je biblioteka ***log4j***.

Implementirane klase za realizaciju kompajlera

Pored generisanih klasa od strane biblioteka, uvedene su i implementirane nove klase koje se koriste za realizaciju semantičke analize i generisanje koda.

Klasa ***SemanticAnalyzer*** predstavlja nasleđenu klasu klase ***VisitorAdaptor*** i implementira skup

```
public void visit(SyntaxNode syntaxNode);
```

metoda koje se pozivaju kad god se nalazimo u nekom čvoru tj. SyntaxNode-u. Ove metode su implementirane po potrebi i svaki čvor nema redefinisanu metodu. Implementirano je zarad provere semantičkih pravila jezika.

Klasa ***CodeGenerator*** takođe vrši obilazak po stablu, ali sad zarad generisanja koda. ***CodeGenerator*** je takođe podklasa klase ***VisitorAdaptor*** i ona isto implementira određen skup visit metoda.

Klasa ***SyntaxAnalysisWatcher*** je klasa koja se koristi za brojanje pojavljivanja klasa, metoda, globalnih promenljivih, konstanti i slično.

Klasa ***Table*** I ***TableRider*** su klase koje nasleđuju klase ***Tab*** i ***DumpSymbolTableVisitor***, respektivno. Table je realizacija tabele simbola, a TableRider je ralizacija klase koja vrši obilazak tabele simbola prilikom ispisa. Table je implementiran kako bi se realizovala podrška za logički tip *bool*.

Pokretanje kompajlera

Prvo je neophodno izgenerisati parser. To postižemo pokretanjem *compile* target-a iz Ant-ovog build fajla.

Potom je potrebno pokrenuti proces kompilacije. To postižemo pokretanjem klase Compiler uz dodatna dva parametra komandne linije

- Putanja do ulaznog fajla
- Putanja do izlaznog fajla

Važna napomena - putanje treba da koriste *backslash* (/) za razdvajanje direktorijuma, odnosno poddirektorijuma.

Pokretanje iskompajliranog Microjava programa se vrši uz pomoć dve konzolne komande:

- *`java -cp lib\mj-runtime.jar rs.etf.pp1.mj.runtime.disasm <program-name>`*
- *`java -cp lib\mj-runtime.jar rs.etf.pp1.mj.runtime.Run <program-name> [-debug]`*

Prva komanda služi za disasembliranje objektnog fajla, druga služi za samo pokretanje programa. Prva varijanta druge komande jeste pokretanje čistog koda, bez izlistavanja linije-po-liniju po toku izvršavanja i ispis *expression stack-a,* dok druga varijanta to omogućava.

Program je moguće pokrenuti i preko Ant build target-a. Jedini problem koji se tiče takvog pokretanja jeste čitanje sa standardnog ulaza. Naime ne podržava *read* i *bread* i neophodno je pokrenuti tad program iz čiste konzole.

Automatizacija pokretanja

Moguće je automatizovati pokretanje procesa kompajliranja i testiranja (u određenoj meri). Postoje dve *powershell* skripte koje imaju zadatak da pokrenu sve neophodne klase za uspesno kompajliranje programa, a potom i njegovo pokretanje.

Sve što je potrebno jeste da se jedan od tih skripti pokrene u root folderu projekta desnim klikom na skriptu i *Run with PowerShell*.

- ***automate_compilation.ps1*** - pokreće proces kompilacije svih dostavljenih testova
- ***automate_start_tests.ps1*** - prethodno je potrebno iskompajlirati sve dostavljene testove. Pokretanjem ove skripte pokreću se svi objektni fajlovi, iliti test programi u standardnom režimu rada
- ***automate_start_tests_debug.ps1*** - prethodno je potrebno iskompajlirati sve dostavljene testove. Pokretanjem ove skripte pokreću se svi objektni fajlovi, iliti test programi u debug režimu rada

Testiranje kompajlera

Testiranje programa je izvršeno kroz nekoliko test primera izlistanih u nastavku bez nekog određenog redosleda.

Ime test primera	Šta je testirano	Izlaz test primera
test1.mj	Dodela globalnim promenljivama i njihov ispis	Uspešan. - test1.obj - test1.out
test2.mj	Dodela lokalnim promenljivama i njihov ispis	Uspešan. - test2.obj - test2.out
test3.mj	Sakrivanje globalnih promenljivih, njihova dodela i ispis (ponašanje ispisa i dodele kao test2.mj)	Uspešan. - test3.obj - test3.out
test4.mj	Test poziva funkcije print za oblik *print(designator, numConst)*. Neophodno ispisati *designator* onoliko puta koliko je definisano *numConst*	Uspešan. - test4.obj - test4.out
test5.mj	Deklarisanje promenljivih u obliku type decl, decl; na globalnom nivou	Uspešan. - test5.obj - test5.out
test6.mj	Deklarisanje promenljivih u obliku type decl, decl; na lokalnom nivou	Uspešan. - test6.obj - test6.out
test7.mj	Testiranje definisanja konstanti	Uspešan. - test7.obj - test7.out
test9.mj	Testiranje svih tipova nizova - inicijalizacija, indeksiranje, dodela, ispis.	Uspešan. - test9.obj - test9.out
test10.mj	Ispis literala	Uspešan. - test10.obj - test10.out
test11.mj	Čitanje bool i int sa standardnog ulaza i ispis podrazumevanih vrednosti	Uspešan. - test11.obj - test11.out
test12.mj	Čitanje char sa standardnog ulaza i ispis podrazumevanih vrednosti	Uspešan. - test12.obj - test12.out

Ime test primera	Šta je testirano	Izlaz test primera
test13.mj	Čitanje sa standardnog ulaza i upis u niz i ispis podrazumevanih vrednosti	Uspešan. - test13.obj - test13.out
test14.mj	Čitanje sa std ulaza i upis u celobrojne promenljive. Ispis njihov i računanje nekog jednostavnijeg i kompleksnog aritmetičkog izraza kako i njihov ispis.	Uspešan. - test14.obj - test14.out
test15.mj	Čitanje celobrojnih promenljivih sa ulaza. Korišćenje pročitanih promenljivih u jednostavnijem i kompleksnijem (ugnežđenom) ternarnom operatoru	Uspešan. - test15.obj - test15.out
test16.mj	Čitanje celobrojnih promenljivih sa ulaza. Korišćenje pročitanih promenljivih u jednostavnijem i kompleksnijem (ugnežđenom) ternarnom operatoru. Indeksiranje nizova u ternarnom operatoru i vraćanje vrednosti indeksiranog polja.	Uspešan. - test16.obj - test16.out
test17.mj	Dodela null vrednosti nizovima. Logičko poređenje niyovnog tipa sa null u ternarnom operatoru. Ispis vrednosti nizova i karaktera	Uspešan. - test17.obj - test17.out
test18.mj	Testiranje kompleksnog ternarnog operatora sa funkcijama ord(), chr() i len()	Uspešan. - test18.obj - test18.out
test19.mj	Redefinicija ugrađenog imena int, char i bool. Int nije rezervisano ime te je moguće redefinisati ga	Uspešan. - test19.obj - test19.out
test20.mj	Semantička provera povratne vrednosti funkcije main i stvarne povratne vrednosti	Uspešan. - test20.obj - test20.out

Ime test primera	Šta je testirano	Izlaz test primera
test21.mj	Semantička provera povratne vrednosti funkcije main i stvarne povratne vrednosti. Stvarna vrednost je dobijena iz ord() ugradjene funkcije.	Uspešan. - test21.obj - test21.out
test22.mj	Semantička provera povratne vrednosti funkcije main i stvarne povratne vrednosti. Stvarna vrednost je dobijena iz len() ugradjene funkcije.	Uspešan. - test22.obj - test22.out
test23.mj	Semantička provera povratne vrednosti funkcije main i stvarne povratne vrednosti. Stvarna vrednost je dobijena indeksiranjem niza	Uspešan. - test23.obj - test23.out
test26.mj	Inkrementiranje i dekrementiranje elemenata niza	Uspešan. - test26.obj - test26.out
test27.mj	testiranje glomaznog ternarnog operatora. Menjanje “term” sa MINUS term. Provera ispravnosti	Uspešan. - test27.obj - test27.out
test1_error.mj	Leksicka greska - pogresno napisan int, char I bool u globalnom delu.	Neuspešan. - test1_error.err
test2_error.mj	Leksicka greska - pogresno napisan int, char I bool u lokalnom delu.	Neuspešan. - test2_error.err
test3_error.mj	Leksicka greska - pogresno napisan int, char I bool u lokalnom delu I globalnom delu.	Neuspešan. - test3_error.err
test4_error.mj	Leksicka greska -Izostavljen jedan apostrof za literal char-a.	Neuspešan. - test4_error.err
test5_error.mj	Sintaksna i leksicka greska -Izostavljen jedan apostrof za literal char-a i zaboravljen drugi Term prilikom sabiranja.	Neuspešan. - test5_error.err

Ime test primera	Šta je testirano	Izlaz test primera
test6_error.mj	Sintaksna greska - pokusaj sabiranja bool-a, i izostavljen drugi operator	Neuspešan. - test6_error.err
test7_error.mj	Semanticka greška - pokušaj dodele neodgovarajućih tipova vrednosti promenljivama drugog tipa	Neuspešan. - test7_error.err
test8_error.mj	Testiranje razlicitih tipova promenljivih za ord(), len() i chr()9	Neuspešan. - test8_error.err
test9_error.mj	Pokusavanje alokacije memorije za niz jednom tipu, dodela nizu drugog tipa	Neuspešan. - test9_error.err
test10_error.mj	Razliciti tipovi u true I false delu ternarnog operatora.	Neuspešan. - test10_error.err
test11_error.mj	Testiranje poređenja različitih tipova u ternarnom operatoru	Neuspešan. - test11_error.err
test12_error.mj	- Pokusaj sabiranja int i char - Pokusaj poredjenja int-char, int-bool, int-array - Razliciti tipovi u true I false delu ternarnog operatora	Neuspešan. - test12_error.err
test13_error.mj	Deklaracija promenljive unutar tela funkcije. Sabiranje int i char Prazne Expr smene u ternarnom operatoru	Neuspešan. - test13_error.err
test14_error.mj	Pokusaj negacija razlicitih tipova promenljivih	Neuspešan. - test14_error.err
test15_error.mj	Poredjenje razlicitih tipova u ternarnom operatoru sa prethodnim pokusajem negacije, sabiranja i slucno	Neuspešan. - test15_error.err
test16_error.mj	Suvisna zagrada za zatvaranje prilikom rada sa ternarnim operatorom	Neuspešan. - test16_error.err
test20_error.mj	Pokušaj da se sakrije naziv globalne konstante. Ovo ne sme da prođe sa uvedenom pretpostavkom	Neuspešan. - test20_error.err