MikroJava prevodilac

Izveštaj sa projekta

Ilija Marković, 0406/2019

# Postavka

Zadatak projekta na predmetu Programski prevodioci 1 (13S114PP1) jeste pravljenje prevodioca za jezik MikroJava (kao Java, ali sa smanjenim mogućnostima za edukativne svrhe) u bajtkod koji se zatim može izvršavati od strane MikroJava virtuelne mašine. Projekat se sastoji od četiri faze:

1. leksička analiza korišćenjem alata JFlex,
2. sintaksna analiza korišćenjem alata AST-CUP,
3. semantička analiza obilaskom apstraktnog sintaksnog stabla,
4. generisanje bajtkoda ponovnim obilaskom i korišćenjem klase Code.

Projekat je rađen za nivo A

# Pokretanje

Projekat se može pokrenuti korišćenjem Ant alata. Nakon pozicioniranja u direktorijum projekta, gde se nalazi build.xml fajl, moguće je pokrenuti ant komandu u slučaju da je direktorijum Ant instalacije dodat u PATH kako bi se projekat kompajlirao. Takođe je moguće pokrenuti komandu ant run kako bi projekat uzeo zadati test program, preveo ga u zadati objektni program pri čemu ispisuje rezultate sintaksne analize, semantičke analize i tabelu simbola. Opcionim argumentima komandne linije može se ispisati bajtkod prevedenog programa kao i pokrenuti taj program sa argumentima datim u datoteci input.txt. Argumenti komandne linije mogu se kontrolisati kroz build.xml.

# 

# Test primeri

## test301.mj

Test primer za A nivo projekta. Test primer prvo testira osnovne operacije i štampa promenljivu bodovi, zatim pravi niz i testira operacije sa nizom i konstantama, kao i destruktuirajuću dodelu, i na kraju radi detaljno testiranje rada sa karakterima i aritmetičkim izrazima.

program test301

const int nula = 0;

const int jedan = 1;

const int pet = 5;

int niz[], niz2[], a, b;

char nizch[];

{

void main()

int bodovi;

bool bt;

{

bodovi = 0;

bodovi++;

bodovi = bodovi + jedan;

bodovi = bodovi \* pet;

bodovi--;

print(bodovi);

niz = new int[3];

niz[nula] = jedan;

niz[1] = 2;

niz[niz[jedan]] = niz[niz[0]] \* 3;

bodovi = niz[2]/niz[0];

print(bodovi);

print(niz[2]);

[a,b] = niz;

print(a); // niz[0] -> 1

print(b); // niz[1] -> 2

[,a,] = niz;

print(a); // niz[1] -> 2

niz2 = new int[2];

[niz2[0],,niz2[niz[nula]]] = niz;

print(niz2[0]); // niz[0] -> 1

print(niz2[1]); // niz[2] -> 6

// [niz2[0],,,niz2[niz[nula]]] = niz; // runtime error!

nizch = new char[3];

nizch[0] = 'a';

nizch[jedan] = 'b';

nizch[pet - 3] = 'c';

print(nizch[1]);

print(nizch[jedan \* 2]);

read(bodovi);

bodovi = -bodovi + (pet \* a / 2 - jedan) \* bodovi - (3 % 2 + 3 \* 2 - 3);

print(bodovi);

}

}

## syntax-errors.mj

Test koji proverava sve sintaksne greške definisane u postavci projekta.

program syntax\_errors

int a+; // Error: oporavak do ;.

int a+, b; // Error: oporavak do ,.

{

void main() {

b = 43+123+; // Error: oporavak do ;.

b = 123;

}

}

# Klase

Klase u projektu su sledeće:

* **rs.etf.pp1.Main:** glavna klasa prevodioica koja na osnovu argumenata komandne linije određuje ulazne i izlazne fajlove, a zatim poziva odgovarajuće klase kako bi izvršila prevođenje
* **rs.etf.pp1.CommandLineParser:** parsira argumente komandne linije
* **rs.etf.pp1.ClassUtils:** uslužna klasa sa pomoćnim metodama za rad sa klasama
* **rs.etf.pp1.lex.Yylex:** autogenerisana klasa leksera od strane JFlex
* **rs.etf.pp1.syntax.Parser:** autogenerisana klasa parsera od strane AST-CUP
* **rs.etf.pp1.syntax.sym:** autogenerisana klasa lekserskog simbola od strane AST-CUP
* **rs.etf.pp1.ast.\*:** paket sa autogenerisanim čvorovima apstraknog sintaksnog stabla od strane AST-CUP
* **rs.etf.pp1.semantics.SemanticAnalyzer:** posetilac apstraktnog sintaksnog stabla koji vrši semantičku analizu programa
* **rs.etf.pp1.generation.CodeGenerator:** posetilac apstraktnog sintaksnog stabla koji vrši generisanje bajtkoda za MikroJava VM