

Programski prevodioci: Vežbe 8

Sadržaj

1. Uvod.....	1
2. Napomena za rešavanje zadataka	1
3. Rešenja zadataka	1
3.1. Zadatak 1: uslovni izraz	1
3.2. Zadatak 2: postinkrement izraz	2
3.3. Zadatak 3: for iskaz	2

1. Uvod

U dokumentu su data rešenja zadataka koji su rađeni na osmim vežbama.

2. Napomena za rešavanje zadataka

Svi zadaci se rešavaju sledećim redosledom:

- Dodati nove tokene na vrh **.y** datoteke.
- Definirati regularne izraze u **.l** datoteci za nove tokene.
- Proširiti gramatiku jezika tako da sintaksno podržava novu konstrukciju.
- Dodati semantičke provere.
- Osmisliti, za 1 konkretan primer, kako ekvivalentan asemblerski kod treba da izgleda.
- Uopštiti asemblerski kod iz prethodnog koraka i implementirati generisanje koda.

3. Rešenja zadataka

3.1. Zadatak 1: **uslovni izraz**

```
%type <i> cond_exp
// tip vrednosti pravila cond_exp je int, jer prenosi indeks u tabeli simbola
// na kom se nalazi lokalna promenljiva ili konstanta

exp :
    ...
    | _LPAREN rel_exp _RPAREN _QMARK cond_exp _COLON cond_exp
    {
        int out = take_reg();
        lab_num++;
    }
```

```

if(get_type($5) != get_type($7))
    err("exp1 i exp2 nisu istog tipa");

code("\n\t\t%s\t@false%d", opp_jumps[$2], lab_num);
code("\n@true%d:", lab_num);
gen_mov($5, out);
code("\n\t\tJMP \t@exit%d", lab_num);

code("\n@false%d:", lab_num);
gen_mov($7, out);

code("\n@exit%d:", lab_num);

$$ = out; //ovaj registar ce biti oslobodjen u MOV naredbi iz iskaza dodele
}
;

cond_exp
: _ID
{
    if( ($$ = lookup_symbol($1, (VAR|PAR))) == NO_INDEX )
        err("'%' undeclared", $1);
}
| literal
;

```

3.2. Zadatak 2: postinkrement izraz

U globalnom nizu treba zapamtiti koje simbole treba inkrementirati, pa onda nakon `assignment_statement`-a proći kroz ceo niz u petlji i izgenerisati inkrement za svaki simbol.

3.3. Zadatak 3: for iskaz

```

%token _FOR
%token _INC

statement
: compound_statement
| assignment_statement
| if_statement
| return_statement
| for_statement
;

```

```

for_statement
: _FOR _LPAREN _ID _ASSIGN literal

```

```

{
    $<i>$ = ++lab_num;
    int i = lookup_symbol($3, VAR|PAR);
    if(i == NO_INDEX)
        err("nedeklarisno %s", $3);
    gen_mov($5,i); //inicijalizacija iteratora, pre pocetka petlje
    code("\n@for%d:", lab_num);
}
_SEMICOLON rel_exp
{
    code("\n\t\t%s\t@exit%d", opp_jumps[$8], $<i>6);
}
_SEMICOLON _ID _INC _RPAREN statement
{
    int i = lookup_symbol($11, VAR|PAR);
    if(i == NO_INDEX)
        err("nedeklarisno %s", $11);
    //generisanje koda za inkrement (na kraju petlje)
    if(get_type(i) == INT)
        code("\n\t\tADDS\t");
    else
        code("\n\t\tADDU\t");
    gen_sym_name(i);
    code(", $1,");
    gen_sym_name(i);

    code("\n\t\tJMP \t@for%d", $<i>6);
    code("\n@exit%d:", $<i>6);
}
;

```