Programski prevodioci: Projekat Šifra:PP-8JC5Q

Sadržaj

1. Napomena

Za svaki zadatak potrebno je uraditi:

1. Sintaksnu analizu
2. Semantičku analizu
3. Generisanje koda

2. Zadatak 1

Omogućiti upotrebu logičkih operatora (and, or) i definisanje uslovnih izraza sa upotrebom ovih operatora.

Primer:

```
if (a > 3 and b < 4){
}
while ( c < 5 or a > 4 and v != 3){
}

NOTE: Nije neophodno da pravite while iskaz, osim ukoliko to nije traženo u nekom od
narednih zadataka.
```

3. Zadatak 2

Proširiti jezik LOOP iskazom koji ima sledeći oblik:

```
"loop" "(" <id> "," <lit1> "," <lit2> "," <lit3> ")"
    <statement>
```

Gde:

- <id> predstavlja lokalnu promenljivu ili parametar, iterator petlje
- <lit1> predstavlja literal, početna vrednost iteratora
- <lit2> predstavlja literal, granica iteratora
- <lit3> predstavlja literal, korak za koji se iterator menja
- <statement> predstavlja iskaz

Realizovati semantičke provere:

1. <id> mora biti prethodno deklarisan
2. <id> i <litN> moraju biti istog tipa

Izvršavanje:

- Inicijalizacija iteratora se vrši samo jednom, pre prvog izvršavanja petlje (<lit1>).
- Ako je lit2 >= lit1:
  - Na početku svake iteracije treba proveriti da li je iterator jednak ili manji od <lit2>, pa ako jeste, izvršiti telo petlje.
  - Nakon izvršenja tela petlje, iterator se uvećava za <lit3> i odlazi se na njegovu proveru.
- Ako je lit2 < lit1:
  - Na početku svake iteracije treba proveriti da li je iterator veći od <lit2>, pa ako jeste, izvršiti telo petlje.
  - Nakon izvršenja tela petlje, iterator se umanjuje za <lit3> i odlazi se na njegovu proveru.

Omogućiti i ugnježdene loop iskaze.

Primer:

```
int a;
loop ( a , 4, 6, 7)
    a = a + 5;

loop ( a, 10, 3, 2) {
    a = a + 3;
    b = a;
}
```

4. Zadatak 3

Proširiti jezik BRANCH iskazom koji ima sledeći oblik:

```
"branch" "[" <var> "->" <const1> "->" <const2> "->" <const3> "]"
    "one" "->" <statement1>
    "two" "->" <statement2>
    "three" "->" <statement3>
    "other" "->" <statement4>
```

Gde:

- <var> predstavlja ime promenljive
- <const1>, <const2> i <const3> predstavljaju konstante
- <statement1>, <statement2>, <statement3> i <statement4> predstavljaju iskaze

Uvek mora postojati tačno tri konstante i tačno četiri iskaza

Realizovati sledeće semantičke provere:

1. Promenljiva var mora biti prethodno deklarisana.
2. Konstante const1, const2 i const3 moraju biti istog tipa kao i var.

Izvršavanje:

- Na početku branch iskaza se izvrši provera vrednosti promenljive var.
- U zavisnosti od te vrednosti, ukoliko je ona jednaka vrednosti konstante const1 izvršava se one naredba, ukoliko je ona jednaka vrednosti konstante const2 izvršava se two naredba, ukoliko je ona jednaka vrednosti konstante const3 izvršava se three naredba.
- Nakon izvršavanja one, two ili three naredbe, tok izvršavanja se preusmerava na kraj branch iskaza.
- Other naredba se izvršava ukoliko se vrednost promenljive var razlikuje od konstanti const1, const2 i const3

Primer:

```
branch [ a -> 1 -> 3 -> 5 ]
    one -> a = a + 1;
    two -> a = a + 3;
    three -> a = a + 5;
    other -> a = a - 3;
```