

Assignment 3 solutions Using RDP

By: Dr. James Choi

*Semantic actions are under-lined

1) Assignment Statement

- A1) $A \rightarrow id = E \{ \text{gen instr (POPM, get address(id)) } \}$
- A2) $E \rightarrow T E'$
- A3) $E' \rightarrow + T \{ \text{gen instr (ADD, nil) } \} E'$
- A4) $E' \rightarrow \epsilon$
- A5) $T \rightarrow F T'$
- A6) $T' \rightarrow * F \{ \text{gen instr (MUL, nil) } \} T'$
- A7) $T' \rightarrow \epsilon$
- A8) $F \rightarrow id \{ \text{gen instr (PUSHM, get address(id)) } \}$

Procedure A ()

```
{
If token = id then
{
save = token;
lexer();
If token = "=" then
{
lexer();
E();
get instr (POPM, get address (save) );
}
else error_message ( "= expected" );
}
else error_message ( " id expected" );
}
```

Procedure E ():

```
{
T ();
E'();
}
```

```
}
```

Procedure E'();

```
{  
  If token = "+" then  
    {  
      lexer();  
      T();  
      gen_instr (ADD, nil);  
      E'();  
    }  
};
```

Procedure T();

```
{  
  F();  
  T'();  
}
```

Procedure T'()

```
{  
  If token = "*" then  
    {  
      lexer();  
      F();  
      gen_instr(MUL, nil);  
      T'();  
    }  
}
```

Procedure F();

```
{  
  If token = id then  
    {  
      gen_instr(PUSHM, get_address(token));  
      lexer();  
    }  
}
```

```
else error_message("id expected");
};
```

Procedure gen_instr(op, oprnd)

```
/* instr_address shows the current insturction address is global */
{
  Instr_table [instr_address].address = inst_address;
  Instr_table [instr_address].op = op;
  Instr_table [instr_address].oprnd = oprnd;
  Instr_address++;
};
```

Example:

$x = a + b * c$

(addresses a = 5001 , b=5002, c=5003 and x = 5004)

INSTR_TABLE

address	Op	Oprnd
1	PUSHM	5001
2	PUSHM	5002
3	PUSHM	5003
4	MUL	nil
5	ADD	nil
6	POPM	5004

Print from INSTR TABLE ignoring "nil"

2. While Statement

W1) W -> while (C) S whileend
W2) C -> E R E
W3) R -> == | ^= | > | < | => | =<

```
Procedure while_statement();  
{  
  If token = "while" then  
    {  
      addr = instr address;  
      gen_instr("LABEL", nil);  
      lexer();  
      If token = "(" then  
        {  
          lexer();  
          C ( );  
          If token = ")" then  
            {  
              lexer();  
              S();  
              gen_instr(JUMP, addr);  
              back_patch(instr address);  
              if token = "whileend"  
                lexer();  
              else error_message ("whileend expected");  
            };  
          else error_message (" ) expected");  
          else error_message ("( expected");  
        }  
      else error_message ("while expected");  
    }  
}
```

```
};
```

Procedure C ()

```
{
E();
If token in R then
{
op = token;
lexer();
E();
case op of
    < : gen_instr (LES, nil);
        push_jumpstack (instr_address); /* another stack need */
        gen_instr (JUMPZ, nil);

    > : /* you need to do other operators*/

    == :
    ^= :

    etc.
}case
}
else error_message (" R token expected");
}
```

Procedure back_patch (jump_addr)

```
{
addr = pop_jumpstack();
Instr_table[addr].oprn = jump_addr;
}
```

Example: while (i < max) i = i + 1; whileend
with addresses I =5000, max = 5001

1. LABEL nil
2. PUSHM 5000
3. PUSHM 5001
4. LES nil
5. JUMPZ 11 /* back patch */
6. PUSHM 5000
7. PUSHM 5001
8. ADD nil
9. POPM 5000
10. JUMP 1
11.

3. if statement

I -> if (C) S ifend

Procedure I ();

```
{
If token ="if" then
{
addr = instr address();
lexer();
If token ="(" then
{
lexer();
C( );
If token = ")" then
{
lexer();
S( );
back_patch(instr_address);
If token = "ifend"
lexer();
else error_message ("ifend expected ");
}
else error_message ("(" expected ");
};
else error_message ("( expected");
}
else error_message ("if expected");
};
```

Example: if (a < b) a = c; ifend

With addresses a = 5000, b = 5001, c = 5002

1. PUSHM 5000
2. PUSHM 5001
3. LES nil
4. JUMPZ 7
5. PUSHM 5002
6. POPM 5000
7. LABEL

NOTE:

- You need work on <Compound>, <Scan>and <Print> statement
- DO NOT create your own instructions