**Assignment 3 solutions**

**Using RDP**

1. **Assignment Statement**

A1) A -> id := E **{gen\_instr (POPM, get\_address(id)) }**

A2) E -> T { } E’ { }

A3) E’ -> + T **{ gen\_intsr (ADD, nil)}** E’ { }

A4) E’ -> ε { }

A5) T -> F { } T’ { }

A6) T’ -> \*F **{gen\_instr (MUL, nil) }** T’ {}

A7) T’ -> ε { }

A8) F -> id **{gen\_instr(PUSHM, get\_address(id)) }**

**Procedure A ()**

{

If token = id then

{

**Save = token;**

Lexer();

If token = “:=” then

{

Lexer();

E();

**Addr = get\_address (save);**

**Get\_instr (POPM, addr);**

}

else error\_message (“= expected”);

}

else error\_massage (“ id expected”);

}

**Procedure E ():**

{

T ();

E’();

}

**Procedure E’();**

{

If token = “+” the

{

Lexer();

T();

**Gen\_instr(ADD, nil);**

E’();

}

}

**Procedure T();**

{

F();

T’();

}

**Procdure T’()**

{

If token = “\*” then

{

Lexer();

F();

**Gen\_instr(MUL, nil);**

T’();

}

}

**Procedure F();**

{

If token = id then

{

**Addr = get\_address (token);**

**Gen\_instr(PUSHM, addr);**

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 =3001, b=3002, c=3003 and x =3004)

**INSTR\_TABLE**

|  |  |  |
| --- | --- | --- |
| Instr Address | Op | oprnd |
| 1 | PUSHM | 3001 |
| 2 | PUSHM | 3002 |
| 3 | PUSHM | 3003 |
| 4 | MUL | nil |
| 5 | ADD | nil |
| 6 | POPM | 3004 |
|  |  |  |

**Print from INSTR\_TABLE ignoring “nil”**

2. While Statement

W -> while ( C ) S

**Procedure while\_statement();**

{

If token = “while” then

{

**Addr = instr\_address;**

**Gen\_instr(“LABEL”, nil);**

Lexer();

If token = “(“ then

{

Lexer();

Condition();

If token = “)” then

Lexer();

If token = “do” then

{

Lexer();

Statement ();

**Gen\_instr(JUMP, addr);**

**Back\_patch (instr\_address);**

}

else error\_message (“do expected”);

else error\_message (“ ) expected”);

else error\_message (“( expected”);

}

else error\_message (“while expect”);

}

**Procedure back\_patch (jump\_addr)**

{

Addr = pop\_jumpstack();

Instr\_table[addr].oprn = jump\_addr;

}

**Procedure condition ()**

/\*

C -> E R E

R -> =, /=, >, <, =>, <=

\*/

{

E();

If roken in R then

{

**Op = token;**

Lexer();

E();

**Case Op of**

**< : gen\_instr (LES, nil);**

**Push\_jumpstack(instr\_address);**

**Gen\_instr (JUMPZ, nil);**

**>: /\* you need to do them \*/**

**=:**

**!=:**

**……….**

}

else error\_message (“ R token expected”);

}

**Example: while ( i < max) i := i + 1 with addresses I =3000, max = 3001**

1. LABEL nil
2. PUSHM 3000
3. PUSHM 3001
4. LES nil
5. JUMPZ 11 /\* back patch \*/
6. PUSHM 3000
7. PUSHM 3001
8. ADD nil
9. POPM 3000
10. JUMP 1
11. …..

**3. if statement**

**I -> if ( C ) S endif**

**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 = “endif”

lexer();

else error\_messgage (“}if expected “);

}

else error\_message (“) expected “);

}

else error\_message (“) expected”);

}

else error\_message (“if expected”);

};

**Example: if (a < b) a := c endif**

**with addresses a = 3000, b = 3001, c = 3002**

1. PUSHM 3000
2. PUSHM 3001
3. LES
4. JUMPZ 7
5. PUSHM 3002
6. POPM 3000

**NOTE:**

* **You need work on <Compound>, <Read>and <Write> statement**
* **DO NOT create your own instructions**