**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, -999)} E’ { }

A4) E’ -> ε { }

A5) T -> F { } T’ { }

A6) T’ -> \*F {gen\_instr (MUL, -999) } T’ {}

A7) T’ -> ε { }

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

**Procedure A ()**

Begin

If token = id then

Begin

Save = token;

Lexer();

If token = “:=” then

begin

Lexer();

E();

Addr = get\_address (save);

Get\_instr (POPM, addr);

end

Else error\_message (“= expected”);

End

Else error\_massage (“ id expected”);

End;

**Procedure E ():**

Begin

T ();

E’();

End

**Procedure E’();**

Begin

If token = “+” the

Begin

Lexer();

T();

Gen\_instr(ADD, -999);

E’();

End

End;

**Procedure T();**

Begin

F();

T’();

End

**Procdure T’()**

Begin

If token = “\*” then

Begin

Lexer();

F();

Gen\_instr(MUL, -999);

T’();

end

End

**Procedure F();**

Begin

If token = id then

Begin

Addr = get\_address (token);

Gen\_instr(PUSHM, addr);

Lexer();

End

Else error\_message(“id expected”);

End;

**Procedure gen\_instr(op, oprnd)**

**/\* instr\_address shows the current insturction address is global \*/**

Begin

Instr\_table [instr\_address].address = inst\_address;

Instr\_table [instr\_address].op = op;

Instr\_table [instr\_address].oprnd = oprnd;

Instr\_address++;

End;

**Example:**

x ::= a + b\*c (addresses a =1001, b=1002, c=1003 and x =1004)

**INSTR\_TABLE**

|  |  |  |
| --- | --- | --- |
| Address | Op | oprnd |
| 1 | PUSHM | 1001 |
| 2 | PUSHM | 1002 |
| 3 | PUSHM | 1003 |
| 4 | MUL | -999 |
| 5 | ADD | -999 |
| 6 | POPM | 1004 |
|  |  |  |

**Print from INSTR\_TABLE ignoring “-999”**

2. While Statement

W -> while ( C ) S

C -> E R E

R -> =, !=, >, <, =>, <=

**Procedure while\_statement();**

Begin

If token = “while” then

Begin

Addr = instr\_address;

Gen\_instr(“LABEL”, -999);

Lexer();

If token = “(“ then

Begin

Lexer();

Condition();

If token = “)” then

Begin

Lexer();

Statement ();

Gen\_instr(JUMP, addr);

Back\_patch (instr\_address);

End;

Else error\_message (“ ) expected”);

Else error\_message (“( expected”);

End

Else error\_message (“while expect”);

End;

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

Begin

Addr = pop\_jumpstack();

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

End

**Procedure condition ()**

Begin

E();

If roken in R then

Begin

Op = token;

Lexer();

E();

Case Op of

< : gen\_instr (LES, -999);

Push\_jumpstack(instr\_address);

Gen\_instr (JUMPZ, -999);

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

=:

!=:

…

Endcase

End

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

End

**Example: while ( i < max) i := i + 1; with addresses I =1000, max = 1001**

1. LABEL -999
2. PUSHM 1000
3. PUSHM 1001
4. LES -999
5. JUMPZ 11 /\* back patch \*/
6. PUSHM 1000
7. PUSHM 1001
8. ADD -999
9. POPM 1000
10. JUMP 1
11. …..

**3. if statement**

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

**Procedure I ();**

Begin

If token =”if” then

Begin

Addr = instr\_address();

Lexer();

If token =”(“ then

Begin

Lexer();

C();

If token = “)” then

Begin

Lexer();

S();

Back\_patch(instr\_address);

If token = “endif”

lexer();

Else error\_messgage (“endif expected “);

End

Else error\_message (“) expected “);

End;

Else error\_message (“) expected”);

End

Else error\_message (“if expected”);

End;

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

**With addresses a = 2000, b = 2001, c = 2002**

1. PUSHM 2000
2. PUSHM 2001
3. LES
4. JUMPZ 7
5. PUSHM 2002
6. POPM 2000

**NOTE:**

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