Practical no: 2

Problem Statement: Implement Pass-II of two pass assembler for pseudo-machine in Java using object-oriented features. The output of assignment-1 (intermediate file and symbol table) should be input for this assignment

Name: Aditi Dinesh Mulay

Class: T.E. Computer

Subject: SPOS

Div: A

Roll no: 02

PRN No. 71918146B

Assignment A-2

Aim: To design data structure for Pass-II Assembler.
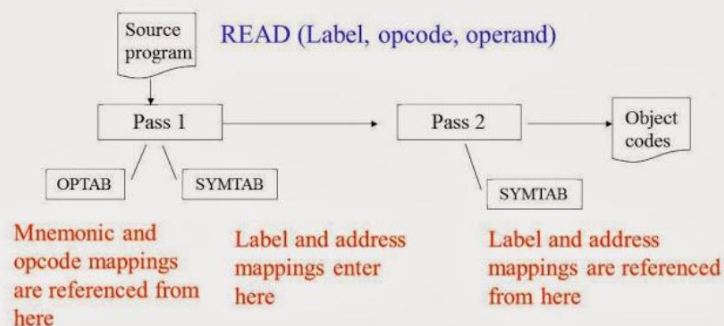
Theory: Two -pass Assembler.

The two pass assembler performs two passes over the source program. In the first pass, it reads the entire source program, looking only for label definitions. All labels are collected, assigned address, and placed in the symbol table in this pass, no instructions as assembled at ~~all~~ the end the symbol table should contain all labels defined in program. To assign address, to lables LC is maintained.

Data Structures:

- Location Counter (L.C) points to the next location where the code will be placed.
- Op-code translation table contains symbolic instructions, their lengths & their op-codes.
- Symbol table contains labels & their values.
- String Storage Buffer- contains ASCII characters for the string.
- Forward Reference Table (FRT) contains pointer to the string in SSB offset where its value will be inserted in the object code.

A Simple Two Pass Assembler Implementation

READ (Label, opcode, operand)

Source program → Pass 1 → Pass 2 → Object codes

OPTAB / SYMTAB (Pass 1)

SYMTAB (Pass 2)

Mnemonic and opcode mappings are referenced from here

Label and address mappings enter here

Label and address mappings are referenced from here

Algorithm

```
begin
        if starting address is given
            LOCCTR = starting address;
        else
            LOCCTR = 0;
        while OPCODE != END do
            begin
            read a line from the code
            if there is a label
                if this label is in SYMTAB, then error
                else insert (label, LOCCTR) into SYMTAB
            search OPTAB for the op code
            if found
                LOCCTR += N
            else if this is an assembly directive
                update LOCCTR as directed
            else error
            write line to intermediate file
            end
        program size = LOCCTR - starting address;
end
```

Conclusion: Thus we have generated machine code for source program.

Program:

```java
40          {
41
42              littab[z][0]=str.split("\\s+")[0];
43              littab[z][1]=str.split("\\s+")[1];
44              z++;
45          }
46
47          while ((str = symtab_file.readLine()) != null) {
48              symtab.put(str.split("\\s+")[0], str.split("\\s+")[1]);
49          }
50
51          String f3 = "C:\\Program Files\\Java\\jdk-10.0.1\\bin\\POOLTAB.txt";
52          FileReader fw3 = new FileReader(f3);
53          BufferedReader pooltab_file = new BufferedReader(fw3);
54          ArrayList<Integer> pooltab = new ArrayList<Integer>();
55          String t;
56          while ((t = pooltab_file.readLine()) != null)
57          {
58              pooltab.add(Integer.parseInt(t));
59          }
60
61          int pooltabptr = 1;
62          int temp1 = pooltab.get(0);
63          int temp2 = pooltab.get(1);
64
65          String sCurrentLine;
66          sCurrentLine = IC_file.readLine();
67          int locptr=0;
68          locptr=Integer.parseInt(sCurrentLine.split("\t")[3]);
69
70          while ((sCurrentLine = IC_file.readLine()) != null)
71          {
72              System.out.print(locptr+"\t");
73              String s0 = sCurrentLine.split("\t")[0];
74              String s1 = sCurrentLine.split("\t")[1];
75
76              if (s0.equals("IS"))
77              {
78                  System.out.print(s1+"\t");
79                  if (sCurrentLine.split("\t").length == 5)
```

```java
79              if (sCurrentLine.split("\t").length == 5)
80              {
81                  System.out.print(sCurrentLine.split("\t")[2] + "\t");
82                  if (sCurrentLine.split("\t")[3].equals("L"))
83                  {
84                      int add = Integer.parseInt(sCurrentLine.split("\t")[4]);
85                      System.out.print(littab[add-1][1]);
86                  }
87                  if (sCurrentLine.split("\t")[3].equals("S"))
88                  {
89                      int add1 = Integer.parseInt(sCurrentLine.split("\t")[4]);
90                      int i = 1;
91                      String l1;
92                      for (Map.Entry m : symtab.entrySet())
93                      {
94                          if (i == add1)
95                          {
96                              System.out.print((String) m.getValue());
97                          }
98                          i++;
99                      }
100                 }
101             } else
102             {
103                 System.out.print("0\t000");
104             }
105         }
106         if (s0.equals("AD"))
107         {
108             littab_file.reset();
109             if (s1.equals("05"))
110             {
111                 int j = 1;
112                 while (j < temp1)
113                 {
114                     littab_file.readLine();
115                 }
116                 while (temp1 < temp2)
117                 {
118                     System.out.print("00\t0\t00" + littab_file.readLine().split("'")[1]);
```

```java
                    System.out.print("00\t0\t00" + littab_file.readLine().split("'")[1]);
                    if(temp1<(temp2-1))
                    {
                        locptr++;
                        System.out.println();
                        System.out.print(locptr+"\t");
                    }
                    temp1++;
                }
                temp1 = temp2;
                pooltabptr++;
                if (pooltabptr < pooltab.size())
                {
                    temp2 = pooltab.get(pooltabptr);
                }
            }
            int j = 1;
            if (s1.equals("02"))
            {
                String s;
                while ((s = littab_file.readLine()) != null)
                {
                    if (j >= temp1)
                        System.out.print("00\t0\t00" + s.split("'")[1]);
                    j++;
                }
            }
            if(s0.equals("DL")&&s1.equals("01"))
            {
                System.out.print("00\t0\t00"+sCurrentLine.split("'")[1]);
            }
            locptr++;
            System.out.println();
        }
        IC_file.close();
        symtab_file.close();
        littab_file.close();
        pooltab_file.close();
    }
```

```java
                    temp1++;
                }
                temp1 = temp2;
                pooltabptr++;
                if (pooltabptr < pooltab.size())
                {
                    temp2 = pooltab.get(pooltabptr);
                }
            }
            int j = 1;
            if (s1.equals("02"))
            {
                String s;
                while ((s = littab_file.readLine()) != null)
                {
                    if (j >= temp1)
                        System.out.print("00\t0\t00" + s.split("'")[1]);
                    j++;
                }
            }
            if(s0.equals("DL")&&s1.equals("01"))
            {
                System.out.print("00\t0\t00"+sCurrentLine.split("'")[1]);
            }
            locptr++;
            System.out.println();
        }
        IC_file.close();
        symtab_file.close();
        littab_file.close();
        pooltab_file.close();
    }
    catch (IOException e)
    {
        e.printStackTrace();
    }
}
```