PICT, PUNE

Subject : LP-1

31118

Name: Shubham Chavate

Pass-1 of two pass assembler

**Title:** Pass-1 of two pass assembler

**Problem statement:**

Design suitable data structures & implement pass-I of two pass assembler for pseudo machine in Java. Implementation should consists of a few instructions from each category & few assembler directives.

**Learning Objectives:**

1) Understand data structures of pass-I assembler.
2) Understand working of two pass assembler.
3) Understand advanced assembler directives.

**Learning Outcomes:**

1) Implement pass I of two pass assembler & understand how symbol table, literal table & pool table are generated.
2) Understand the function of advanced assembler directives.

**Software & Hardware Requirements**

OS : Windows 10 (64bit)
Eclipse 2020 for Java programming
Intel Core i5 - processor
8 GB RAM & 512 GB SSD.

**Concept Related Theory:**

**Introduction:**

There are two main classes of programming languages: High level & low level. Assembly is low level programming

PICT, PUNE

language Assembler is a program that accept input as assembly language program & produces its machine language equivalent along with information for loader.

## Pass-I Assembler:

- Functions of Assembler.
  1) Generate Machine instructions
     - i) Evaluate mnemonics to produce their machine code
     - ii) Evaluate the symbol literals, addresses to produce their equivalent machine addresses.
     - iii) Convert the data constants into their machine representation.
  2) Process pseudo operations

## Data Structures in Pass-I:

1) location Counter (LC)
2) Opcode Translation Table : Contains Symbolic instructions, their lengths & their opcodes (or subroutine to use for Translation)
3) Symbol Table (SYMTAB) : contains labels & their values
4) Literal Table (LITTAB) : contains literals & their memory address.
5) Pool Table: (POOLTAB) : Contains starting literal of every pool.

PICT, PUNE

## Statements format:

An assembly language statement has following format

[label] <opcode> <operand1> [operand2]

[] → this shows optional.

## Assembly Language Statements:

There are three types of opcodes.

i) Imperative Statements (IS)
ii) Assembler Directive (AD)
iii) Declarative Statements (DL)

1) Imperative Statements: It indicates action to be performed during the execution of the assembled program. Each imperative statement typically translates into one machine instruction.

2) Assembler Directives: It instruct the assembler to perform certain actions during the assemble of a program.

Some important Assembly directives are:
START, END, ORIGIN, EQU, LTORG.

3) Declarative Statements:

There are of two types

i) Constants declarative (DC)
ii) Storage declarative (DS)

**Algorithm:**

1. Initialize locptr, littab_ptr & pooltab_ptr.
2. Process Instructions line by line.

PICT, PUNE

a) If label is present, add label to SYMTAB
b) If LTORG statement then
   i) process literals in pool & allocate memory.
   ii) Add last literal number to pool table.
c) If ORIGIN statement then
   update loc ptr.
d) If EQU statement then
   correct the symbol table entry for the label to the address specified in the operand field.
e) If declaration statements then if a DS statement then increment loc ptr by size specified.
f) If an imperative statement then.
   i) code = machine opcode from OPTAB
   ii) increment loc-ptr
   iii) If operand is a literal ~~table~~ then add literal to LITTAB
   else if operand is a symbol then
   add symbol to SYMTAB.
   iv) Write the corresponding intermediate representation to output file.

3. Process END statement.

**Test Case:**

Behaviour proof

Screenshots are attached along with input assembly code file.

PICT, PUNE

**Conclusion:**

In this assignment I learned working of pass-1 of two pass assembler. I learned about data structure used in the process. I also implemented it in Java programming language.