Page :
Date : / /

Assignment A3

Title : Pass I of a 2-Pass Macro Processor

Date of Completion : 23/01/2020

Problem Statement :
Design Suitable data structures & implement pass-I of a two pass macro processor using OOP features in Java

Objectives :
i. Implement Pass I of a two pass macro processor
ii. Implement using OOP features

Outcomes :
Students should be able to
i. Implement Pass I of a two pass macro processor
ii. Apply concepts of OOP

Software & Hardware ~~Algorithm~~ Requirements :
Fedora 64 bit OS
Eclipse IDE Java
i5 processor
4 GB RAM
500 GB HDD

14 | Implement algorithmic techniques by virtue ofadvance programming skill set and Free and Open Source Software (FOSS) tools.

Page:
Date:

Theory :

Macro :

Macro allows a sequence of source language code to be defined once & then referred to by name each time it has to be referred.
Each time this name occurs in a program, the sequence of code is substituted at that point.

A Macro consists of

1) Name of the macro
2) Set of Parameters
3) Body of the macro.

Example

```
MACRO                    ~ Start of definition
   mymacro               ~ Macro Name
   ADD areg, X           } Macro body
   ADD breg, X           }
MEND                     ~ End of Macro Definition
```

A macro is called by writing the macro name with actual parameters in an assembly program

Syntax :

<macro name> [<list of parameters>]

Example :

```
INCR    X
 ↑      ↑
macro name  parameter
```

Page :
Date : / /

Each call to a macro is replaced by its body

During replacement, actual parameter is used in place of formal parameter

A macro can have two types of parameters
(1) Positional Parameters
(2) Keyword Parameters

(1) Positional Parameter :
A Positional parameter is written as &parameter_name.
During macro calls, actual values of parameters are substituted on the basis of positions in the macro call statements

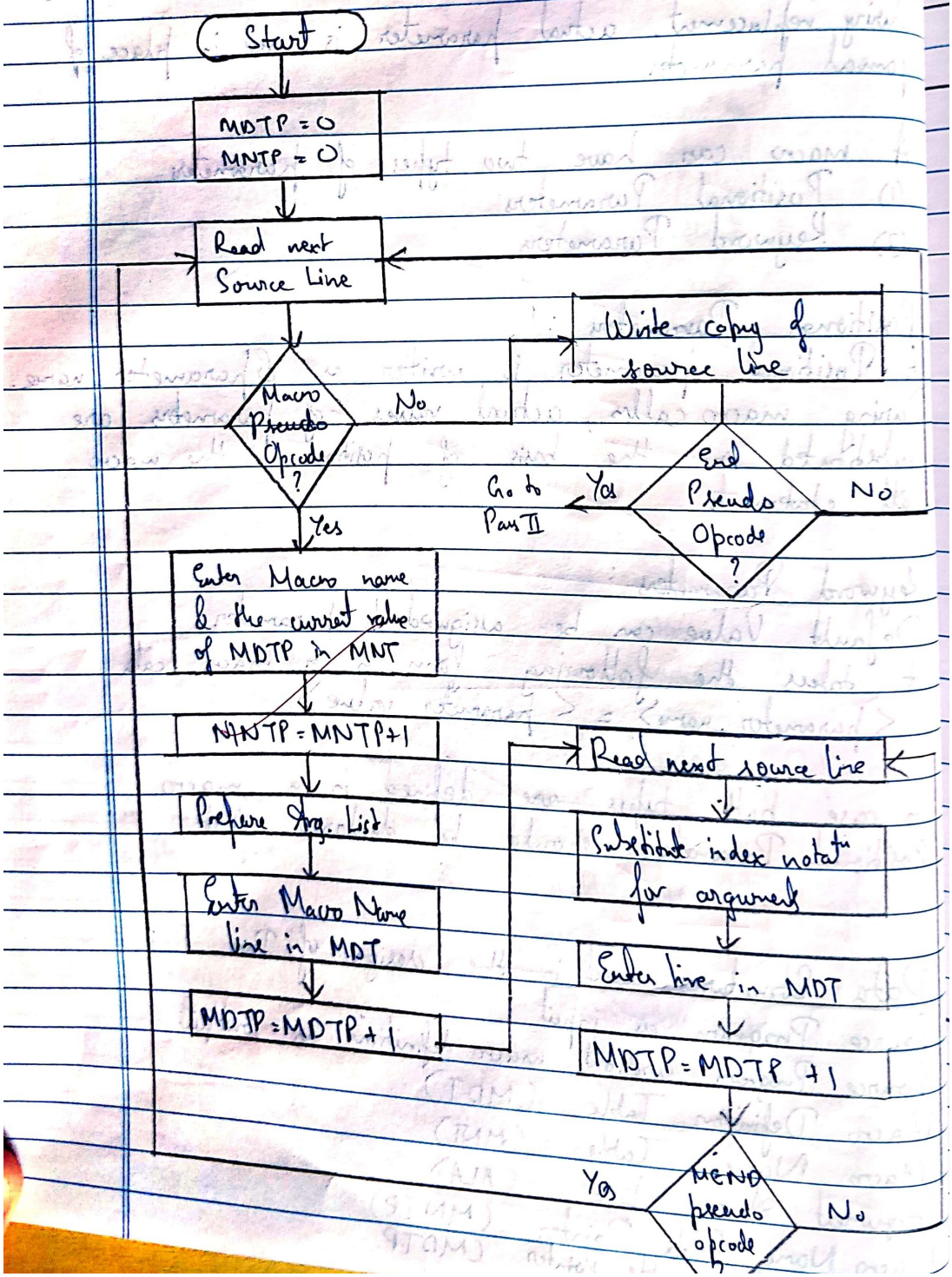
(2) Keyword Parameters :
Default Value can be assigned to parameter
It takes the following form in a macro call
<parameter_name> = <parameter_value>.

In case both types are defined in a macro, Positional Parameters should be declared first.

Data Structures used in the design of Pass I :-
1. Source Program as input
2. Source Program without macro definition as output
3. Macro Definition Table (MDT)
4. Macro Name Table (MNT)
5. Argument Array List (ALA)
6. Macro Name Table Pointer (MNTP)
7. Macro Definition Table Pointer (MDTP

Page:
Date:

Flow Chart for Pass I



Page :
Date : / /

Test Cases

Description	Input	Output	Result
MNT	Macro Code	<table><tr><th>name</th><th>#PP</th><th>#KP</th><th>MDTP</th><th>KPDTP</th></tr><tr><td>M1</td><td>2</td><td>2</td><td>1</td><td>1</td></tr><tr><td>M2</td><td>2</td><td>2</td><td>6</td><td>3</td></tr></table>	Success
KPDTAB	Macro Code	<table><tr><th>Name</th><th>Value</th></tr><tr><td>A</td><td>AREG</td></tr><tr><td>B</td><td>-</td></tr><tr><td>U</td><td>CREG</td></tr><tr><td>V</td><td>DREG</td></tr></table>	Success
PNTAB of M1	Macro Code.	<table><tr><th>Sr No.</th><th>PN</th></tr><tr><td>1</td><td>x</td></tr><tr><td>2</td><td>y</td></tr><tr><td>3</td><td>A</td></tr><tr><td>4</td><td>B</td></tr></table>	Success
MDT (Line 4)	ADD &A, ='5'	ADD (P,3), ='5'	Success

Sushant 14/02/20

Conclusion :
Using the concepts of OOP, We have succesfully implemented pass I of a two pass macro processor.

techniques by virtue of advance programming skill set and Free and Open Source Software (FOSS) tools.