Name: Digvijay Jondhale
Roll No: PC92
PRN: 10322017770
Panel: C

SSC LCA 04

Aim: Design suitable data structure & implement pass II of Two pass Macroprocessor.

Theory:

- **Algorithm for Phass II :**

1. **Initialization:** Initialize necessary data structure and counters for Pass II processing.
2. **Read Intermediate code:** Read the intermediate code, perform the following steps:
   a.) **Parse Line:** Parse the intermediate code line to identify its components, such as opcode, operands, and symbols.
   b) **Resolve Symbols:** If there are unresolved symbols in the line, look up their values in the symbol table. If a symbol is found, replace it with its corresponding value. If a symbol is not found, report an error.
   c) **Generate Machine Code:** Translate the intermediate code into machine code or another target representation, taking into account the resolved symbols and macro instructions.

d.) Output Machine Code: Write the generated machine code or target representation to the output file or memory.

4.) Repeat: Repeat the above steps for each line of intermediate code until you have processed the entire program.

5.) Finish processing: close any open files, release allocated memory, and perform any cleanup tasks.

6.) End of Pass II: Pass II is complete, and the output file should now contain the final assembled code, ready for execution.

- Data Structures Required for Two-Pass Macroprocessor:

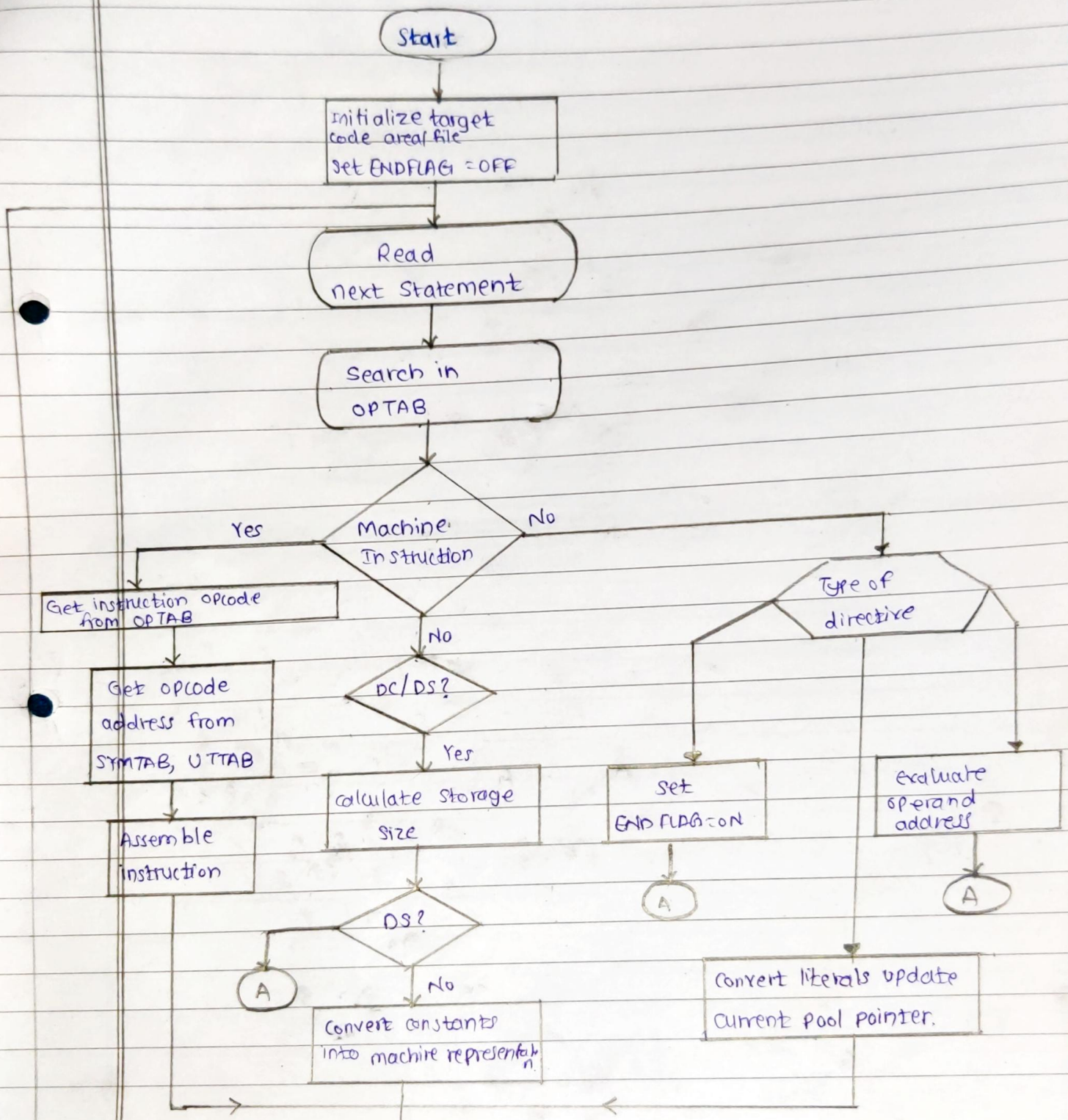
1. Intermediate Code Buffer: This data structure holds the intermediate code generated during Pass I. It can be implemented as an array or a linked list, with each element representing a line of intermediate code.
2. Symbol Table: A symbol table is used to store symbols encountered during Pass I and their corresponding values or addresses. This table is essential for resolving symbols during Pass II. It can be implemented as a hash table, binary search tree, or a simple array.
3. Macro Definition Table: This table stores information about macros defined in this source code during Pass I. It includes the macro name, its parameters, and the corresponding macro code. It allows the macroprocessor to expand macros correctly during Pass II.

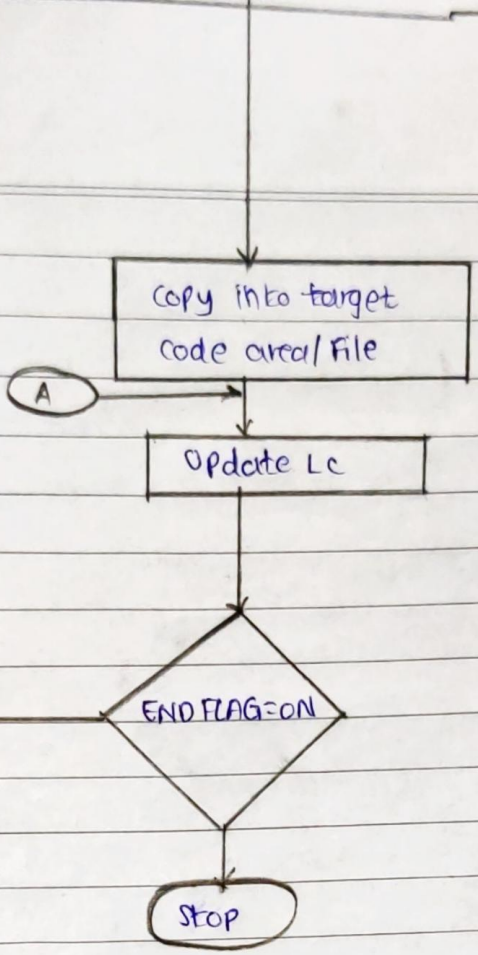
4. Output Buffer: An output buffer is used to temporaily store the generated machine code or target representation before writing it to the output file. It helps in efficiently managing the output data.

5. Error Reporting Mechanism: A mechanism for reporting errors encountered during pass II is essential. This may include an error log or a console output to inform the user about any issues in the source code.

6. Counters and Flags: Various counters and flags are needed to keep track of the current processing state, such as the line number being processed and whether unresolved symbols were encountered.

- **Flowchart of Pass II**




Copy into target
code area/File
A
Update Lc
No
END FLAG=ON
Stop