Documentation for Two-Pass Assembler Program

Introduction

The provided Java code implements a Two-Pass Assembler with a graphical user interface (GUI) using Java Swing.

Assemblers convert assembly language into machine code. The code simulates a two-pass assembler process,

which reads and processes assembly language in two passes:

1. Pass One: Identifies memory locations and builds the symbol table.
2. Pass Two: Generates the object code from the intermediate code and symbol table.

The GUI allows users to input their assembly file and an opcode table (Optab) file and generates the intermediate code,

symbol table, and object code, displaying them within the application.

Classes and Methods

1. TwoPassAssemblerGUI Class

This class builds the GUI for the two-pass assembler. It includes several GUI components:

- JTextField: Fields for user inputs (Input File and Optab File).
- JButton: Buttons for file browsing and initiating the assembly process.
- JTextArea: Output areas for displaying the intermediate code, symbol table, and object code.

Key Methods:

- browseFile(): Opens a file dialog for selecting input files.
- runAssembler(): Invokes the TwoPassAssembler process, loads the opcode table, and runs the two-pass assembly process.

- displayIntermediateCode(): Displays the intermediate code in the GUI.
- displaySymbolTable(): Displays the symbol table in the GUI.
- displayObjectCode(): Displays the object code in the GUI.

2. TwoPassAssembler Class

This class performs the core functionality of the two-pass assembler process.

Key Methods:

- loadOptab(): Reads the opcode table from the Optab file and stores the mappings in a HashMap.
- passOne(): Processes the assembly code to generate the symbol table and intermediate code.
- passTwo(): Uses the symbol table and intermediate code to generate the final object code.
- getIntermediate(): Retrieves the intermediate code.
- getSymtab(): Retrieves the symbol table.
- getObjectCode(): Retrieves the object code.

3. Main Class

This class launches the application by invoking the TwoPassAssemblerGUI class.

How the Two-Pass Assembler Works

1. Pass One:
   - Reads the assembly source code.
   - Generates an intermediate file and symbol table.
   - Updates the location counter (LOCCTR) based on instructions, data directives, and labels.

2. Pass Two:
   - Uses the intermediate file and symbol table to generate the final object code.
   - Each instruction is converted to its machine code equivalent using the opcode table (Optab).

Output

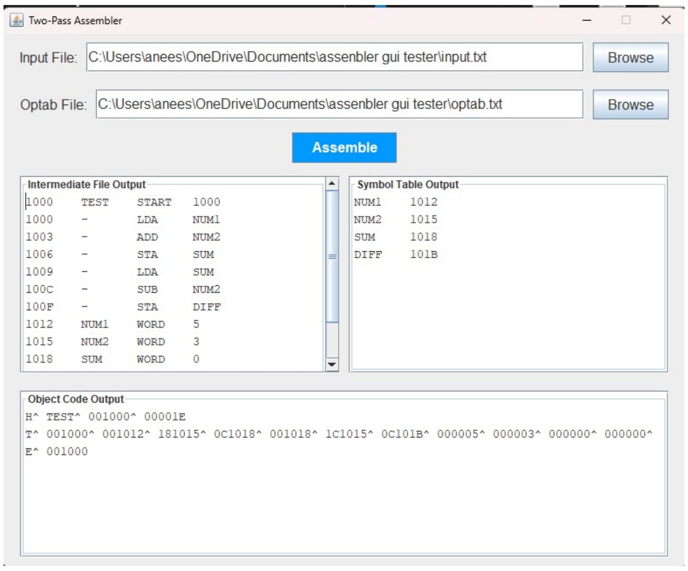
The assembler produces three key outputs:

1. Intermediate Code: Displays the assembly code with its corresponding memory locations.

2. Symbol Table: Shows each label in the program with its assigned memory location.

3. Object Code: The final machine code generated from the assembly source.

Output Image



Input File: C:\Users\anees\OneDrive\Documents\assenbler gui tester\input.txt

Optab File: C:\Users\anees\OneDrive\Documents\assenbler gui tester\optab.txt

Intermediate File Output

```
1000    TEST    START   1000
1000    -       LDA     NUM1
1003    -       ADD     NUM2
1006    -       STA     SUM
1009    -       LDA     SUM
100C    -       SUB     NUM2
100F    -       STA     DIFF
1012    NUM1    WORD    5
1015    NUM2    WORD    3
1018    SUM     WORD    0
```

Symbol Table Output

```
NUM1    1012
NUM2    1015
SUM     1018
DIFF    101B
```

Object Code Output

```
H^ TEST^ 001000^ 00001E
T^ 001000^ 001012^ 181015^ 0C1018^ 001018^ 1C1015^ 0C101B^ 000005^ 000003^ 000000^ 000000^
E^ 001000
```