AssemblerApp: A PROJECT ON 2 PASS ASSEMBLER SIMULATION IN GUI

USER MANUAL FOR TWO PASS ASSEMBLER GUI APPLICATION

Overview

AssemblerApp is a Java-based graphical user interface (GUI) application that simulates a basic two-pass assembler. The application allows users to load an assembly program, process it using a two-pass algorithm (Pass 1 and Pass 2), and generate corresponding outputs such as intermediate files, symbol tables, and object codes.

The assembler supports standard assembly directives like START, WORD, RESW, RESB, BYTE, and END. It also processes symbolic labels and machine instructions by consulting an opcode table.

Installation and Setup

- o Download the appropriate JDK for your platform.
- o Download or clone the source code of the Two-Pass Assembler GUI application to your local machine.
- o Open a command prompt or terminal.
- o Navigate to the directory where the source code is saved.
- o Compile the program:

```
javac pass2.java
```

- o Run the program:

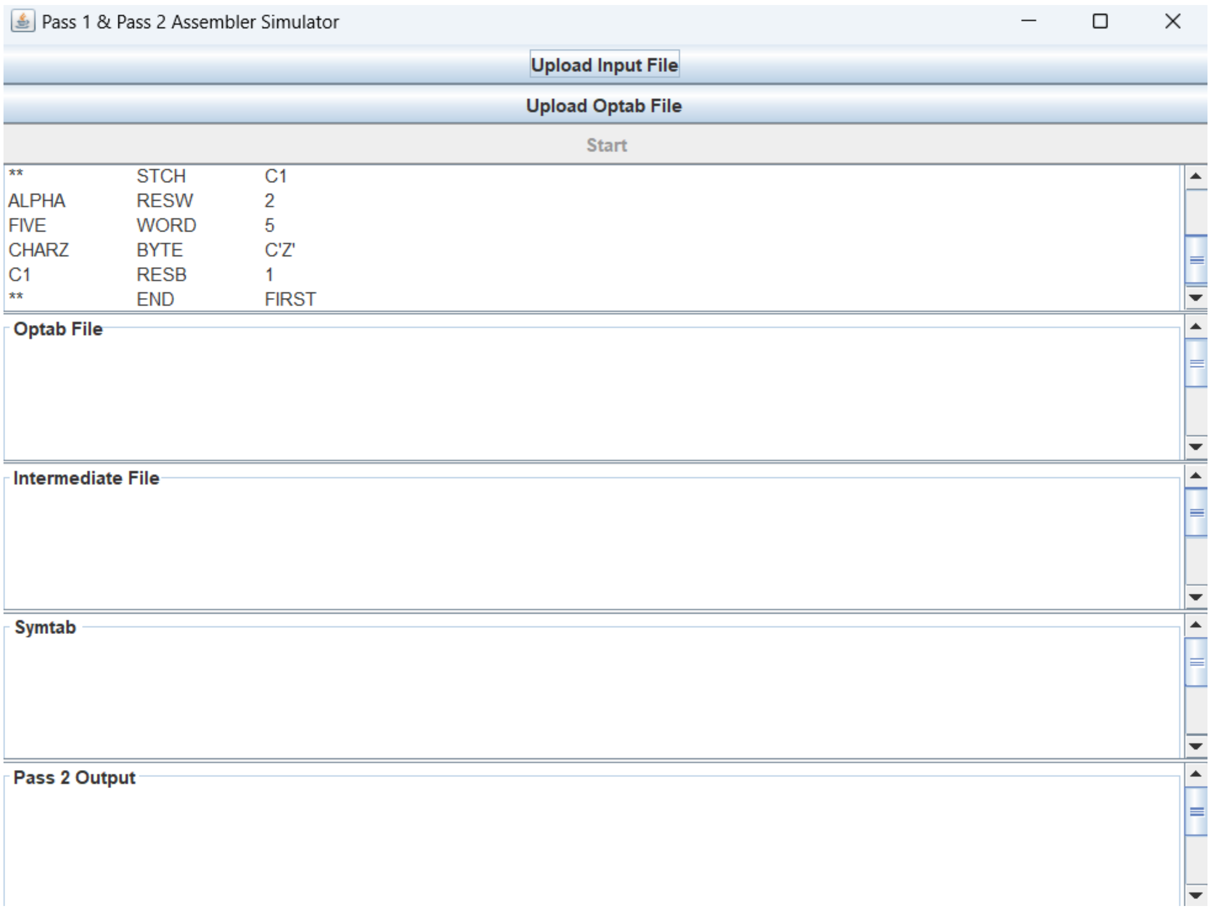
```
java AssemblerApp
```

System Requirements

- Java Development Kit (JDK) 8 or higher

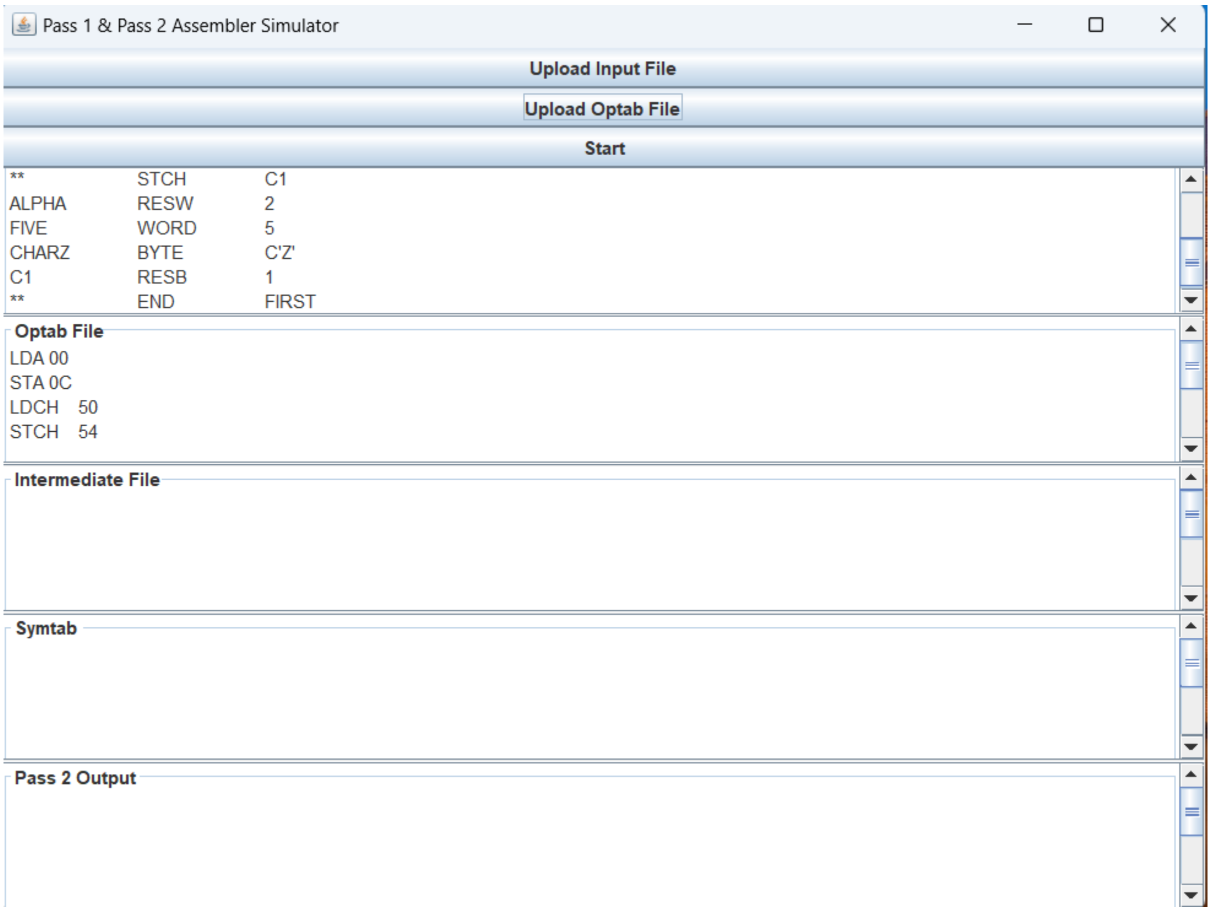
The Gui interface like the following:

USER MANUAL FOR TWO PASS ASSEMBLER GUI APPLICATION



```
**          STCH        C1
ALPHA       RESW        2
FIVE        WORD        5
CHARZ       BYTE        C'Z'
C1          RESB        1
**          END         FIRST
```

USER MANUAL FOR TWO PASS ASSEMBLER GUI APPLICATION



```
**          STCH        C1
ALPHA       RESW        2
FIVE        WORD        5
CHARZ       BYTE        C'Z'
C1          RESB        1
**          END         FIRST
```

Optab File

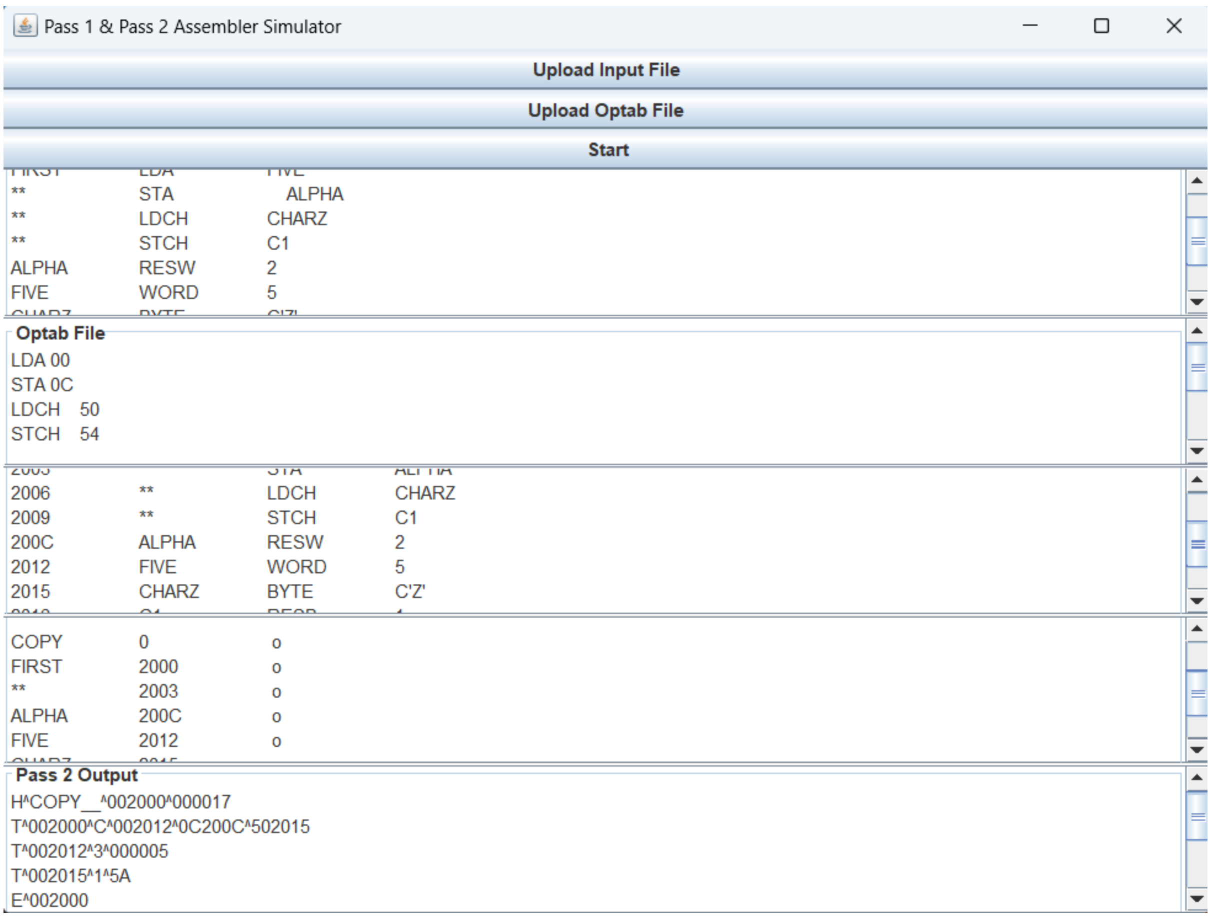
```
LDA 00
STA 0C
LDCH    50
STCH    54
```

Intermediate File

Symtab

Pass 2 Output

USER MANUAL FOR TWO PASS ASSEMBLER GUI APPLICATION



```
**          STA         ALPHA
**          LDCH        CHARZ
**          STCH        C1
ALPHA       RESW        2
FIVE        WORD        5
```

Optab File

```
LDA 00
STA 0C
LDCH   50
STCH   54
```

```
2006        **          LDCH        CHARZ
2009        **          STCH        C1
200C        ALPHA       RESW        2
2012        FIVE        WORD        5
2015        CHARZ       BYTE        C'Z'
```

```
COPY        0           o
FIRST       2000        o
**          2003        o
ALPHA       200C        o
FIVE        2012        o
```

Pass 2 Output

```
H^COPY__^002000^000017
T^002000^C^002012^0C200C^502015
T^002012^3^000005
T^002015^1^5A
E^002000
```

Properties of Key Buttons:

Input button: Used to load the assembly input file into the input text area.

Upload optab File: Used to load the assembly optab file into the optab text area.

Run : Used to run the assembler by executing Pass 1 and Pass 2.

Intermediate file : Used to display intermediate text.

symtab: Used to display symtab.

Pass 2 output: Used to display pass 2 output (final object code in HTE (Header, Text, End) format).

Steps to Use the GUI

1. Entering the Source Code: Type or paste your assembly language code in the Input Source Code text area using Input File button.
2. Entering the opcode Code: Type or paste your assembly language code in the Input Source Code text area using optab File button.
3. Click on rum buttion to display pass 2 output.

Example for input and optab:

Input:

```
COPY    START   2000
FIRST   LDA     FIVE
        STA     ALPHA
        LDCH    CHARZ
        STCH    C1
ALPHA   RESW    2
FIVE    WORD    5
CHARZ   BYTE    C'Z'
C1      RESB    1
        END     FIRST
```

Optab:

```
LDA 00
STA 0C
LDCH    50
```

STCH   54

Assembly Directives Supported

- **START**: Indicates the starting address of the program.
- **END**: Marks the end of the program.
- **WORD**: Allocates a 3-byte word.
- **RESW**: Reserves a specified number of words in memory.
- **RESB**: Reserves a specified number of bytes in memory.
- **BYTE**: Allocates memory based on character or hexadecimal data.

Notes

- Make sure that your assembly input file follows the correct syntax and structure to avoid errors.
- The program assumes a simple SIC (Simplified Instructional Computer) architecture, with 3-byte instructions and a basic set of directives.

Conclusion

AssemblerApp is a simple, intuitive tool designed for students and professionals who want to understand the workings of a two-pass assembler. By following this user manual, you should be able to effectively load assembly programs, run the assembler, and interpret the generated outputs.