

**GEORGE WASHINGTON UNIVERSITY**

**Fall 2023**

**CSCI 6461 Computer System Architecture**

**Project Part 3 - Simulator for Executing All Instructions**

**User Guide**

**Team 12 -**

Sai Bharath Reddy Lattupalli - G41128949

Namana Y Tarikere - G21372717

Reshma Rajkumar - G36576199

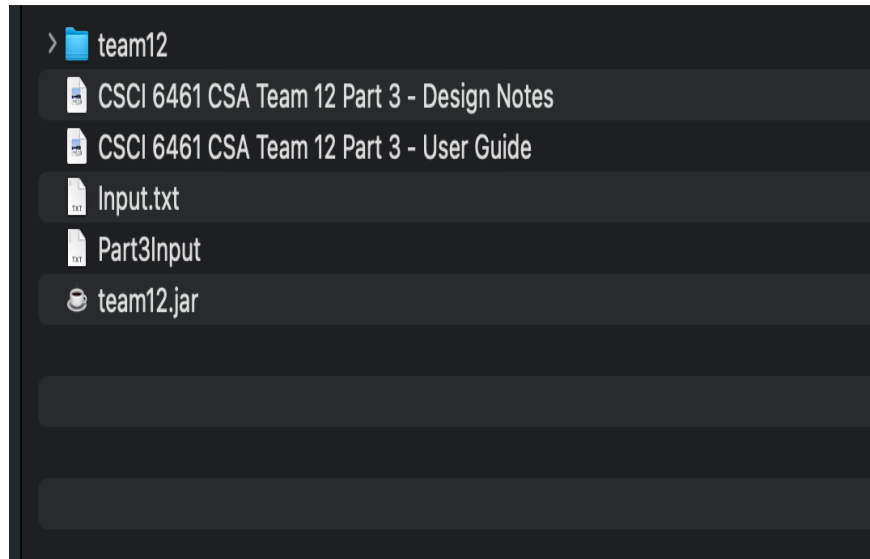
Vaishnavi Goyal - G47669343

1. **Preparation Instructions:** Install Java.

2. **Download the below file from the blackboard: Filename:** Team 12\_Project 3.zip

3. **Execution Instructions:**

- a. Download the file named Team 12\_Project 3.zip.
- b. Extract the zip file and make sure all the files indicated below are present before you run the jar file.



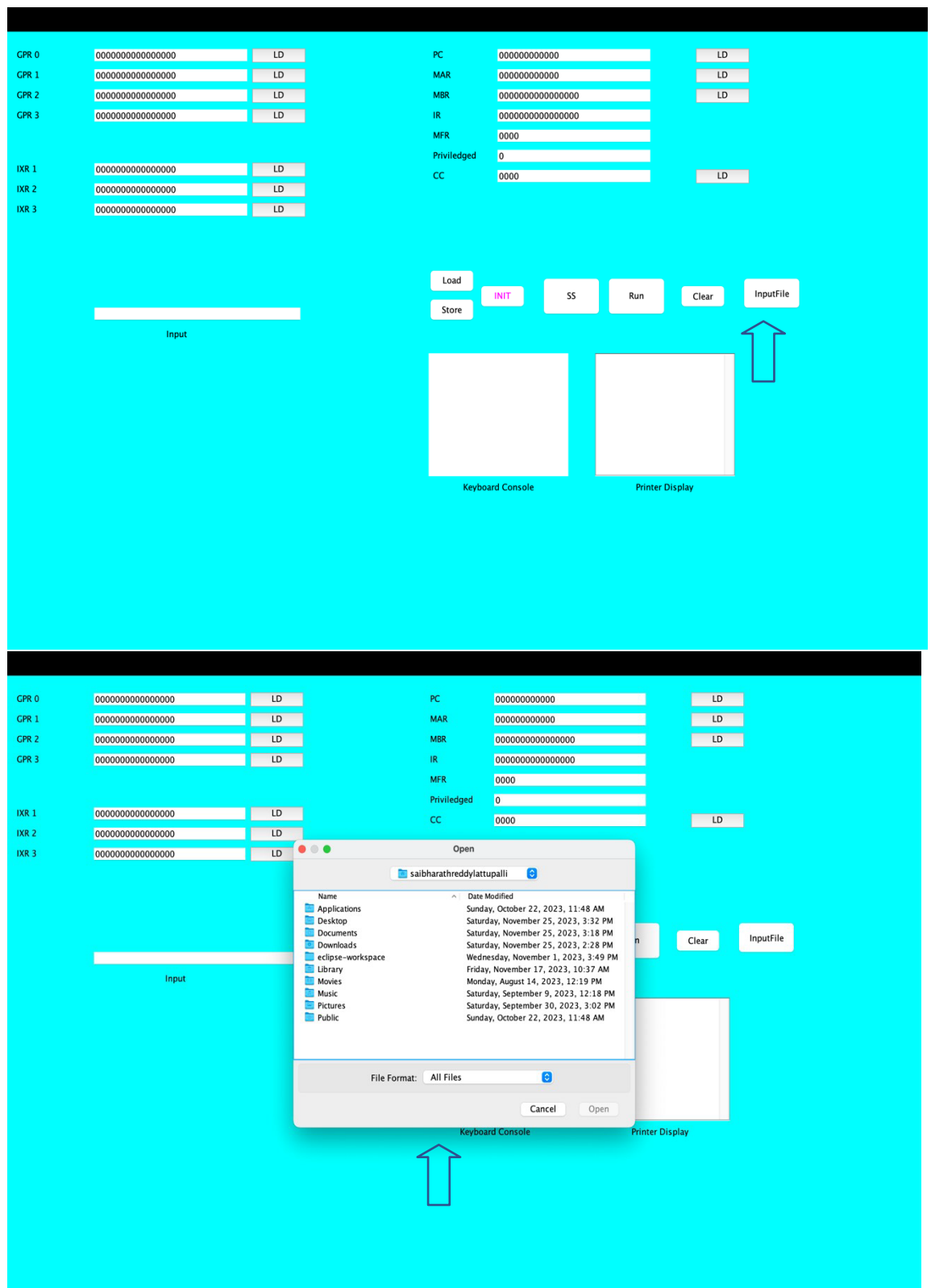
- c. Double click on team12.jar to run the jar file or open terminal and execute the “java -jar team12.jar” from the folder where you have these files.

4. **Updates in this Project Phase 3:**

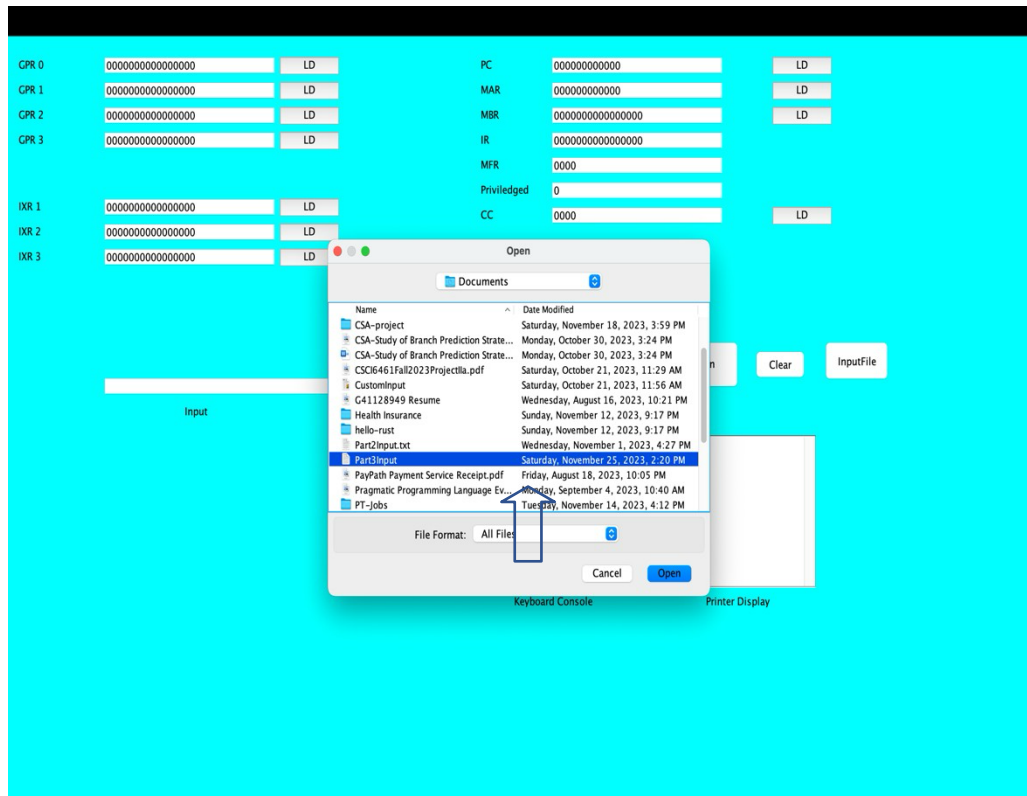
As continuation from part2, in this phase we have implemented all Instructions and added “Keyboard” and “Printer” consoles to the UI. Now the assembler will take a file with instructions from the user and convert its contents to hexadecimal format and then decodes those instructions.

5. **Operating the simulator:**

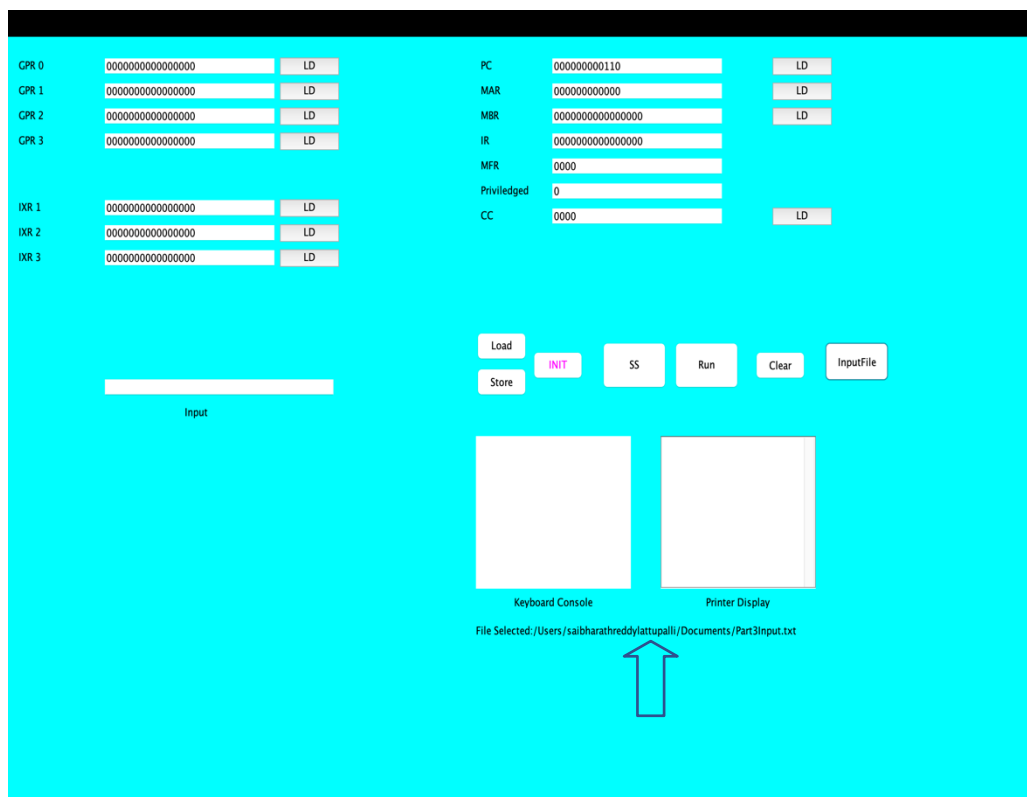
- a. Click on “InputFile” button. A popup will appear asking the user to select their intended file.



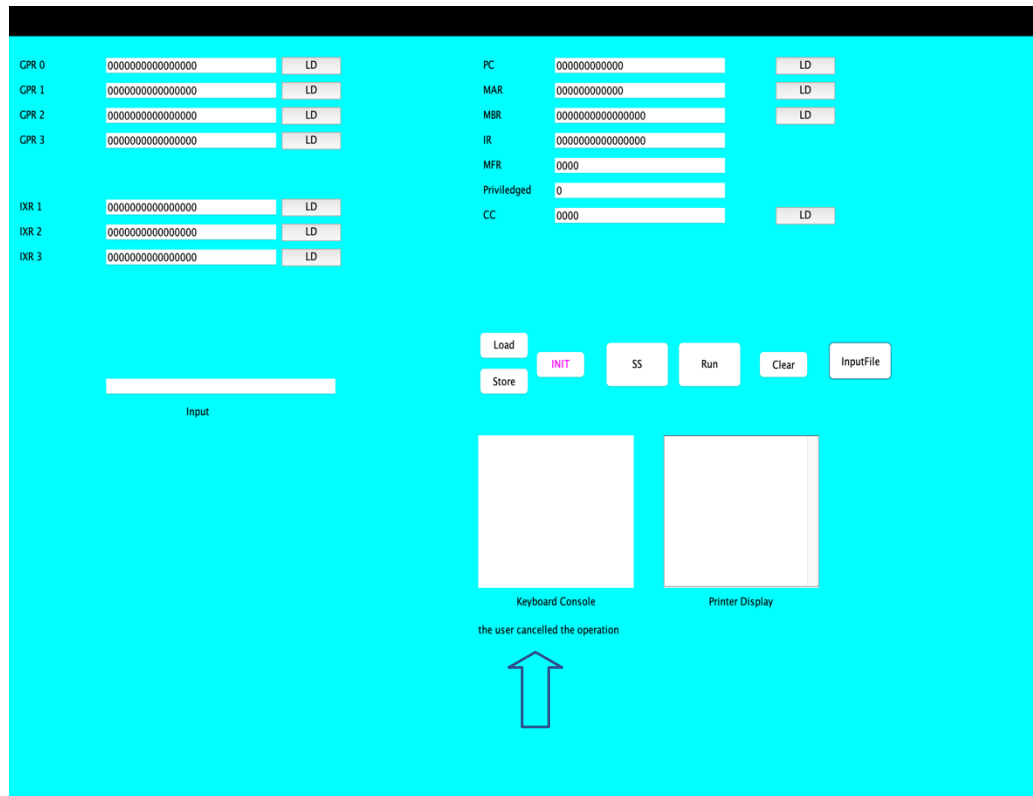
b. Select the input file "Part3Input" and then click on open.



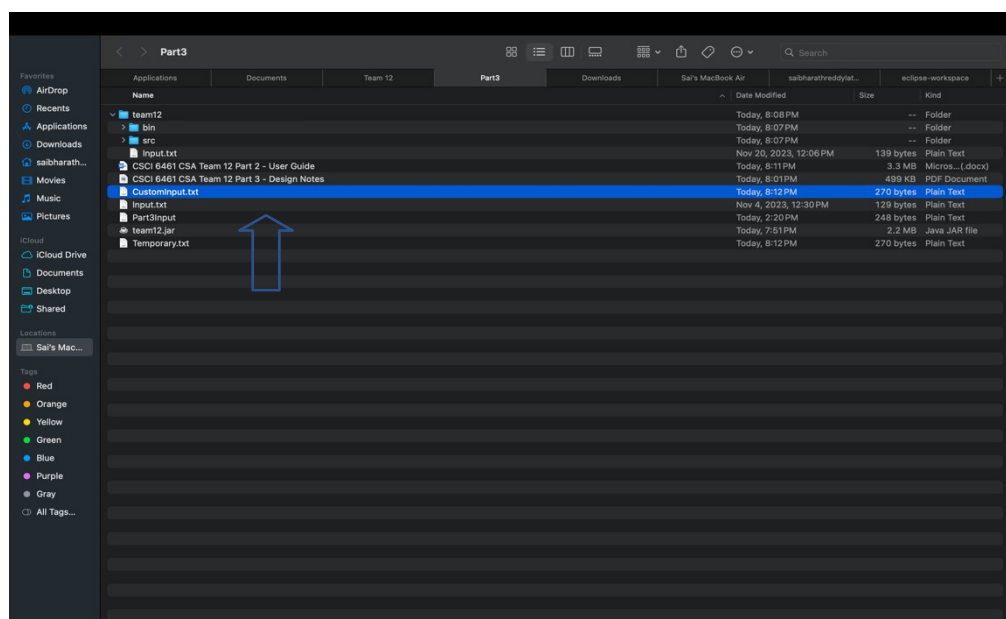
- c. If file is selected successfully then the absolute path of the selected file will be displayed on the UI.

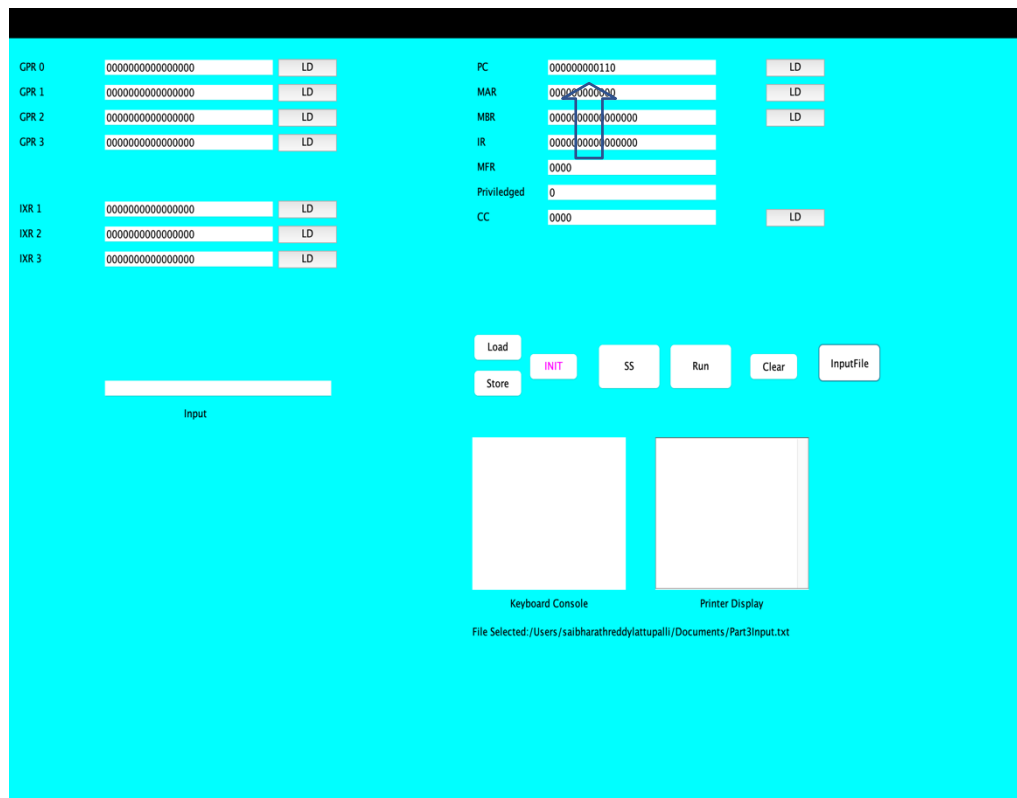


- d. Else, if user selected “Cancel” in step 2, the message “The user cancelled the operation” will be displayed on the UI.

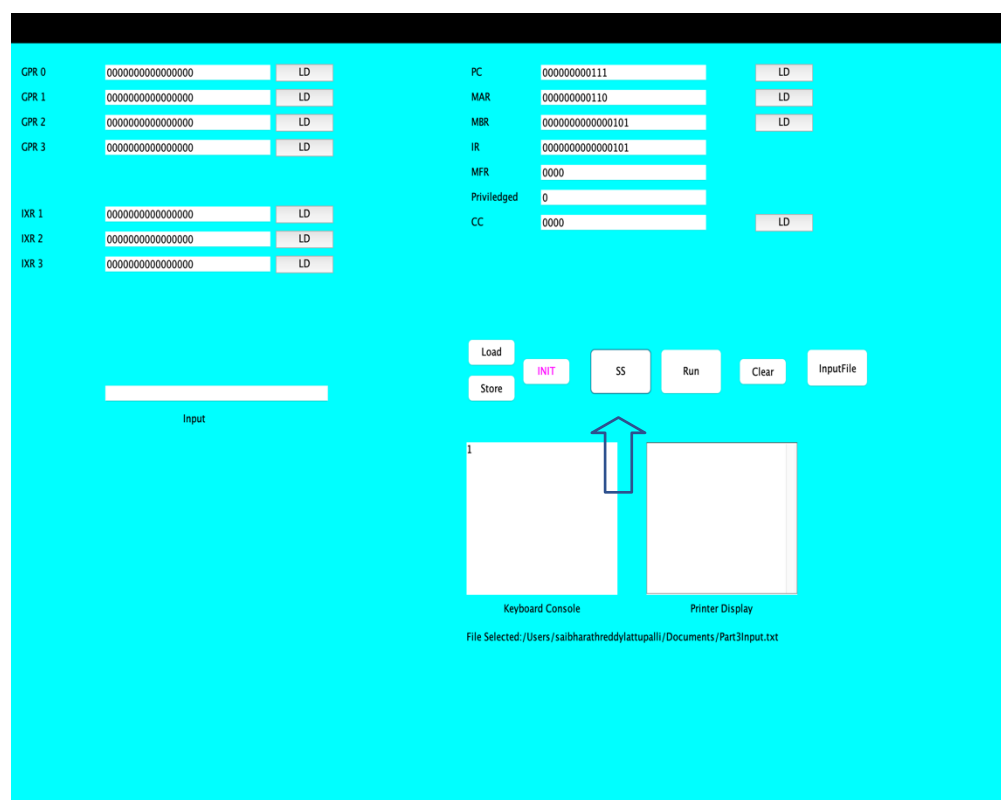


- e. If file is selected successfully, then the content of the file will be converted to hexadecimal format and will be written to a new file “CustomInput.txt” and the first address in this file will be displayed at “PC”.





- f. Click on “SS” to execute the instructions single step at a time.



- g. To execute “IN” and “OUT” instructions use “Keyboard Console” and “Printer Display”.

While executing “IN”, give a number into the “Keyboard Console”. This number will get stored into the register mentioned in the Instruction.

E.g.: IN 1,0 – This will store the number into GPR1.

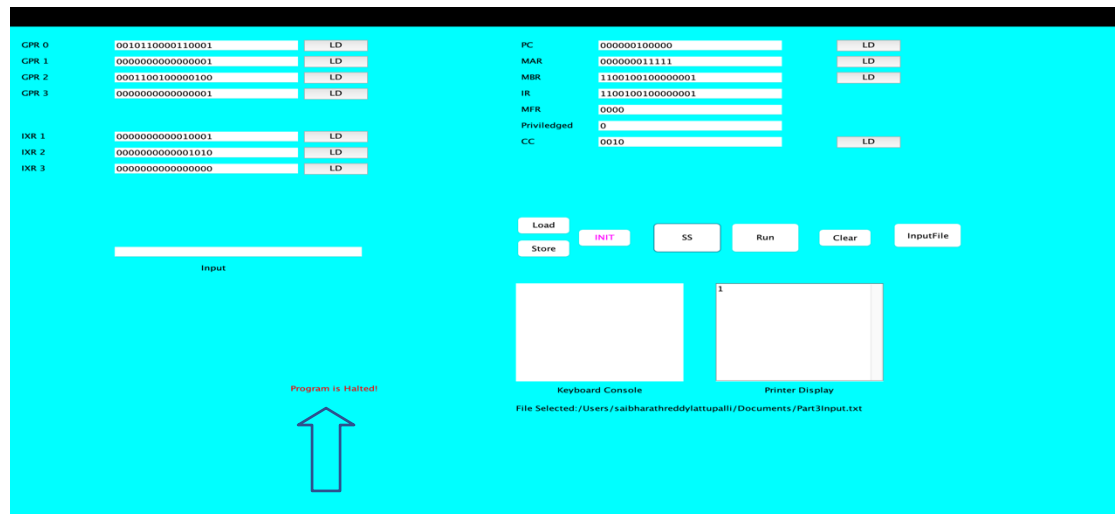
The screenshot shows a computer architecture simulator interface. On the left, there are registers: GPR 0, GPR 1, GPR 2, GPR 3, IXR 1, IXR 2, and IXR 3. Each register has a value field and an LD button. A blue arrow points from the 'Input' field (a text box) to the value field of GPR 1. On the right, there are more registers: PC, MAR, MBR, IR, MFR, Privileged, and CC. Each has a value field and an LD button. Below the registers, there are buttons: Load, Store, INIT (highlighted in pink), SS, Run, Clear, and InputFile. At the bottom, there are two large empty boxes labeled 'Keyboard Console' and 'Printer Display'. Below these boxes, it says 'File Selected: /Users/saibharathreddylattupalli/Documents/Part3Input.txt'.

The “OUT” instruction will read the value from the register specified in the instruction and displays it in “Printer Display”.

E.g.: OUT 1,1 – This will read the value from gpr1 and displays.

The screenshot shows the same computer architecture simulator interface as before. The registers and buttons are the same. However, the 'Printer Display' box now shows the number '1'. A blue arrow points from the 'Printer Display' box to the 'Input' field. Below the 'Printer Display' box, it says 'Program is Halted!'. The 'File Selected' text at the bottom remains the same.

- h. Once all the instructions from the memory are executed, “Program is Halted” gets displayed on the UI.

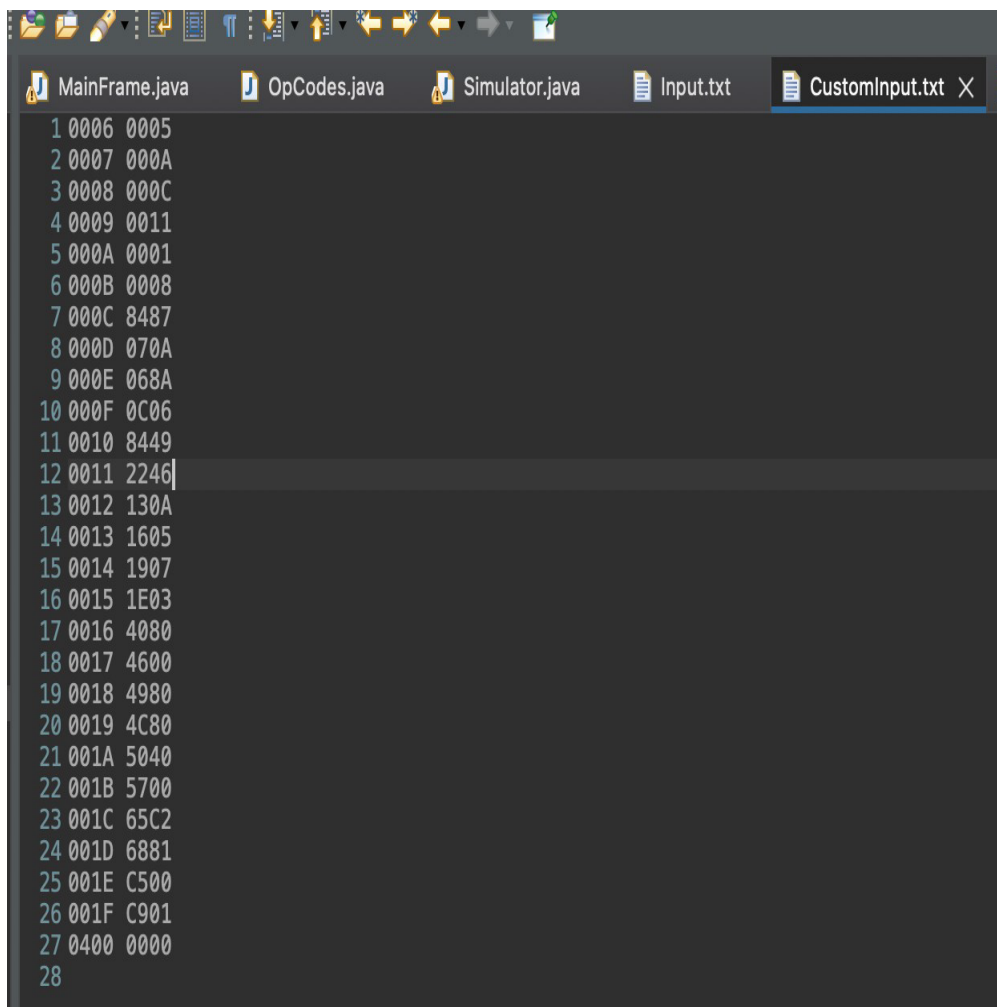


- i. Content of “Part3Input” (User File)

```
Part3Input
LOC 6
Data 5
Data 10
Data 12
Data 17
Data 1
Data 8
LDX 2,7
LDR 3,0,10
LDR 2,2,10
LDA 0,0,6
LDX 1,9
JZ 2,1,6
AMR 3,0,10
SMR 2,0,5
AIR 1,7
SIR 2,3
MLT 0,2
DVD 2,0
TRR 1,2
AND 0,2
ORR 0,1
NOT 3
SRC 1,2,1,1
RRC 0,1,0,1
IN 1,0
OUT 1,1
LOC 1024
End: HALT
```



j. Content of "CustomInput.txt" (Generated File)



The image shows a screenshot of a Java IDE with a dark theme. The top toolbar contains various icons for file operations, editing, and running. The tab bar at the top shows five open files: MainFrame.java, OpCodes.java, Simulator.java, Input.txt, and CustomInput.txt. The CustomInput.txt tab is currently selected and active. The editor area displays the content of this file, which consists of 28 lines of text. Each line is numbered from 1 to 28 on the left margin. The text on each line represents a memory address and its corresponding value in hexadecimal. Line 12 is currently selected, and the cursor is positioned at the end of the line.

```
1 0006 0005
2 0007 000A
3 0008 000C
4 0009 0011
5 000A 0001
6 000B 0008
7 000C 8487
8 000D 070A
9 000E 068A
10 000F 0C06
11 0010 8449
12 0011 2246
13 0012 130A
14 0013 1605
15 0014 1907
16 0015 1E03
17 0016 4080
18 0017 4600
19 0018 4980
20 0019 4C80
21 001A 5040
22 001B 5700
23 001C 65C2
24 001D 6881
25 001E C500
26 001F C901
27 0400 0000
28
```