Assignment 1- MARS ASSEMBLER

By – Vedant Mundada (BT2024268)

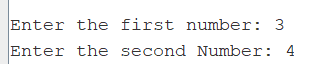
* CONTENTS-

1. **Assembler.py (to assemble the MIPS code)**
2. **Machine\_code.txt (the assembler output is stored in this file)**
3. **Binary\_search.asm**
4. **GCD.asm**
5. **GCD folder –**
6. **Code for GCD (GCD.cpp)**
7. **Assemble code for GCD (GCD.asm)**
8. **Machine code generated from MARS – (GCD\_MARS.bin)**
9. **Machine code generated form my assembler – (GCD\_mc.bin)**
10. **Binary Search-**
11. **Code for Binary Search (Binary\_search.cpp)**
12. **Assembly code for Binary Searcg (Binary\_search.asm)**
13. **Machine code generated from MARS – (Binary\_search\_MARS.bin)**
14. **Machine code generated form my assembler – (Binary\_search\_mc.bin)**
15. **Comparator Folder-**
16. **Assembled code to be placed here (Assembled.txt)**
17. **MARS code to be placed here (Original.txt)**
18. **Compare.py (To compare the codes in Assembled.txt and Original.txt and find any discrepancies)**

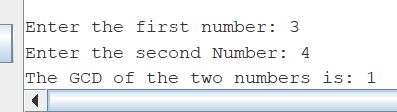
* GCD MIPS code-

**The code follows the Recursive Stein’s Algorithm-**

**After assembling it will ask to input the two numbers for which the we want to find the GCD-**

****

**After pressing enter, the output will be displayed-**

****

* Binary Search Code-

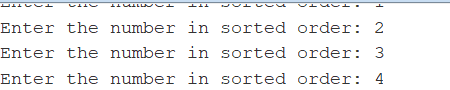
**After assembling, it will ask to input the number of integers in the** **array**

****

**After that it will ask to input the staring address of the array location (ex – 268501216)**



**After that we have to enter the numbers one by one in a sorted manner**



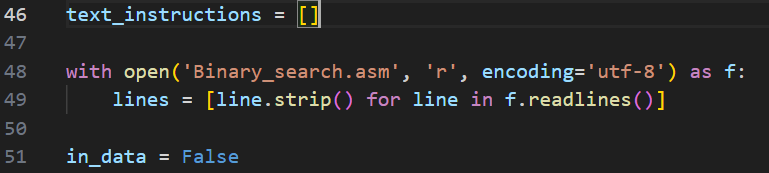
**Then we have to input the number we want to search for-**



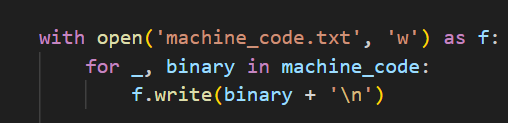
**The output will give the index (0 – indexed) if the number exists or it will say that the number is not found-**



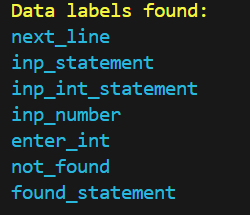
* Assembler  
  **The assembler will take a .asm or .txt file as input and put the output in a file called “machine\_code.txt”**

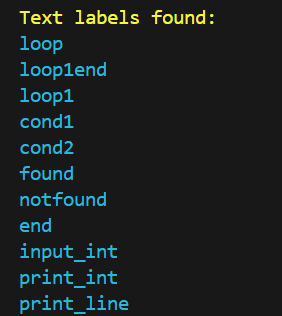
**For giving the input file, we have to put the file name at this location and compile the code:**  


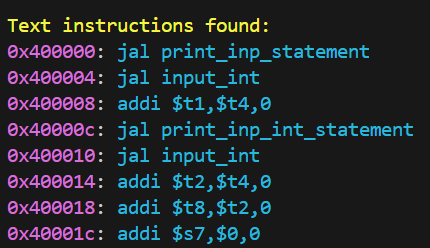
**Output file:**

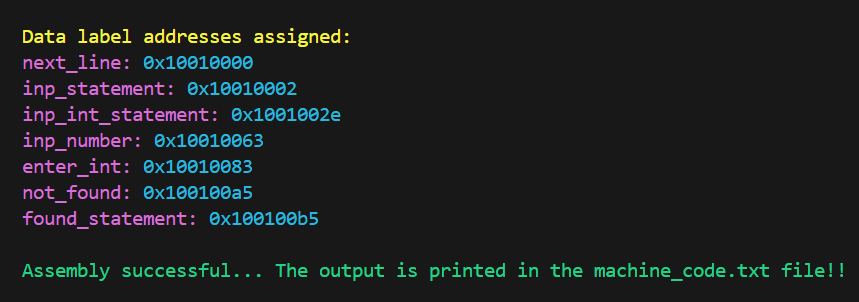


**After compiling, the machine code will be available in the output file and the process the assembler went with will be displayed in the cmd lines:**



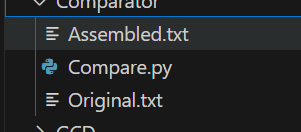


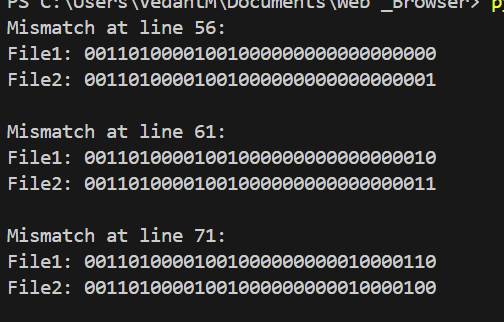




* Comparator

**It just a checking tool that can be used to check two text files Assembled.txt and Original.txt, if the codes are exactly the same then it will output that the files are same, but it they are not then it will display the lines where there is a mismatch-**





***How to run-***

***Command = python .\Compare.py .\Original.txt .\Assembled.txt***

Disclaimer:

**While giving the input to the assembler, we have to give the entire file including the .data segment as well, as it will use that to make address references for the data segments as well, and when we compare the assembled text with the MARS machine code, the mismatches shown are all address mis-matches of the .data segment as the assembler is not able to precisely calculate the address of all the .data segments as the MARS simulator, apart from that everything else should match.**  
The instruction included in the assembler are:

1. **Li**
2. **Ble**
3. **La**
4. **Jal**
5. **J**
6. **Addi**
7. **Or**
8. **Andi**
9. **Bne**
10. **Srl**
11. **Beq**
12. **Sub**
13. **Sllv**
14. **Jr**
15. **And**
16. **Sll**
17. **Lw**
18. **Sw**
19. **Slt**
20. **Syscall**