SUBMITTED BY

**Keyboard functions in Jack**

A PROJECT REPORT

Submitted by – Batch A TEAM\_7

**01**

**02**

M.PRASANNA TEJA G. HIMAMSH

(CB.EN.U4AIE21035) ( (CB.EN.U4AIE21014)

**03**

**04**

G.SRI VATSANKA VIKHYAT BANSAL

(CB.EN.U4AIE21010) ( (CB.EN.U4AIE21076)

### ELEMENTS OF COMPUTING-II

### 

**Centre for Computational Engineering and Networking**

**AMRITA SCHOOL OF ENGINEERING**

## AMRITA VISHWA VIDYAPEETHAM

COIMBATORE - 641 112

SUPERVISED BY – Prof.Sreelakshmi K (Asst.Professor )

**TABLE OF CONTENT**

**1)AIM**

**2)ACKNOWLEDGEMENT**

**3)DECLARATION**

**4)ABSTRACT**

**5)INTRODUCTION**

**6)JACK LANGUAGE**

6.1- Jack OS . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

6.2-Standard Library. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .7

6.3-Keyboard class . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

6.3.1-Conversions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

**7)ASCII CODE**

7.1- Theory . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

7.2-ASCII Table . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

8)Code

8.1- Keyboard Code . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .11

8.2-Main Jack Code . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .13

**9)Implementation**

**10) Extras**

**11)Conclusion**

**AIM**

**Implementing the keyboard library in Jack:**

1. Returns the ASCII code (as char) of the currently pressed key, or 0 if no key is currently pressed.

2. Selects the following key from the keyboard: awaits the moment a key is pressed before

the key is released, after which it is echoed to the screen and its value is returned.  
  
3. Prints the message on the screen, reads the next line (until a newline

character) from the keyboard, and returns its value.

4.Prints the message on the screen, reads the next line (until a newline

character) from the keyboard, and returns its integer value (until the first nonnumeric

character

**ACKNOWLEDGEMENT**

We are deeply thankful to Center for Excellence in **Computational Engineering and Networking (CEN) at Amrita Vishwa Vidyapeetham, Coimbatore** for providing us such a wonderful environment to peruse our research. We would like to express our sincere gratitude to **Prof.SreeLakshmi. K (Asst. Prof) Department of CEN, Amrita Vishwa Vidyapeetham Coimbatore**. We have completed our research under her guidance we found the research area, topic and problem with her suggestions she guided us with our study and supplied us as many research papers and academic resources in this area.

We would also like to extend our gratitude to

**Jyothish Lal.G (Asst.prof) department of CEN, Amrita Vishwa Vidyapeetham Coimbatore** who provided us the basic knowledge of this field of this research. we would also like to acknowledge our team members for supporting each other and be grateful to university for providing this opportunity.

Finally, we would like to thank Noam Nisan, Shimon and their team for providing us wonderful tool and course named **Nand2tetris** which helped us a lot in research work.

**DECLARATION**

We, **The Team-7 of Batch-A** declare that the work embodied in this project

work hereby, titled “**Keyboard functions in Jack**”, forms our team contribution to the research work carried out under the guidance of **Prof.SreeLakshmi. K (Asst. Prof) Department of CEN, Amrita Vishwa Vidyapeetham Coimbatore.**

It is a result of our team research work and has not been previously submitted to any other University for any other Degree/ Diploma to this or any other University.

Wherever reference has been made to previous works of others, it has been clearly indicated as such and included. We, here by further declare that all information of this document has been obtained and presented in accordance with academic rules and ethical conduct. So, we hope you will like the project which we had done.

**From groupmates of Team-7**,

M. Prasanna Teja (CB.EN.U4AIE21035)

G. Himamsh (CB.EN.U4AIE21014)

G. Sri Vatsanka (CB.EN.U4AIE21010)

Vikhyat Bansal (CB.EN.U4AIE21076)

#### **Place: Ettimadai**

Date: XX-7-2022

**ABSTRACT**

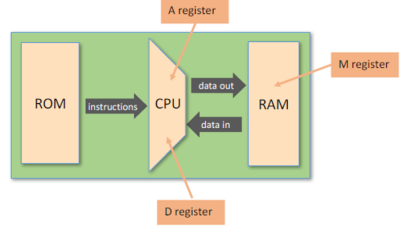
* We must develop a Jack code and use the Jack Compiler to compile the Jack file to VM file. Later, the VM file will be translated to an assembly language code file, which is abbreviated as ASM.
* After developing the ASM file, that ASM file will be converted to a HACK file with help of ASSEMBLER.
* The appropriate files, such as VM, ASM, and hack files, must be examined in the appropriate platforms, such as VM emulator, CPU emulator, hardware simulator, and assembler. The nand2tetris webpages are the source of this program.

**Required Steps:**

* Step-1: We will convert .Jack files to .VM files.
* Step-2: We will convert .Jack files to .VM files.
* Step-3: We will convert .Jack files to .VM files.

**Introduction**

1. Keyboard functions are provided, and we are requested to create a Jack programme that completes the task, compiles, and runs successfully on a VM emulator.
2. High-level programmers can be created using Jack, a straightforward object-based language.
3. We have to write Jack code, and with help of Jack Compiler, we have to compile the jack file to VM file, later VM file is to be converted to assembly language code file, simply called ASM file.
4. Finally, that assembly code file is to be converted to 16- bit machine code i.e., hack file with the help of an assembler.
5. We have to make computer chip. Then to check hack files in hardware simulator we have to load computer chip first.
6. The final hack is to be loaded in the computer chip in hardware simulator. We have to check the respective files, like VM, ASM file, hack file in the respective platforms like VM emulator, CPU emulator, hardware simulator, assembler.
7. This software is provided by the nand2tetris websites.



**6.JACK LANGUAGE**

**6.1-JackOS:**

1. Writing Jack programs requires working with the Jack OS, just like writing Java programs requires working with the Java class library.
2. The Jack OS can be viewed as a collection of software services that extend the basic language's capabilities and close gaps between it and the underlying hardware. Here is the [Jack OS API.](https://www.nand2tetris.org/_files/ugd/44046b_a89c06efcef3491cafb42fb76cc95dc0.pdf)
3. The Nand2tetris Software Suite includes two Jack OS implementations.
4. One OS implementation was written in Jack, and was then translated using a Jack compiler into the set of 8 class files Math.vm, Screen.vm, Output.vm, Keyboard.vm, Memory.vm, String.vm, Array.vm, and Sys.vm.
5. We have to further study on library of Keyboard which will be discussed further.
6. These files are stored in the nand2tetris/tools/os folder in your computer.
7. The second OS implementation was written in Java, and is an integral part of the supplied VM emulator, also available in nand2tetris/tools.
8. Below we explain how to use each OS version.

**6.2-Jack standard Library**:

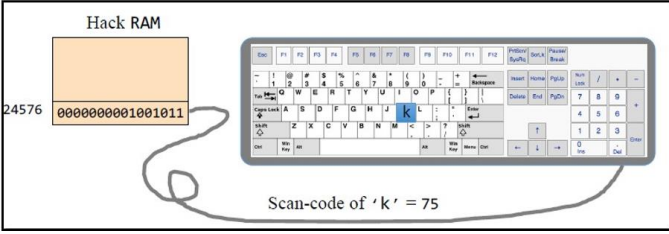
* The jack programming language includes a standard library which can be seen as an interface to an underlying operating system.
* The library is a set of jack classes that must be included in every jack implementation.
* The following classes are included in a standard Library:

1. **Math**: Provides basic Mathematical operations.
2. **String**: Implements the string type & basic string-related operation
3. **Array**: Defines array type, array construction & disposal of array
4. **Output**: Handles text-based output.
5. **Screen**: Handles Graphic screen output.
6. **Keyboard**: Handles user input from the keyboard
7. **Memory**:    Handles memory operations.
8. **Sys**: Provides some execution-related services

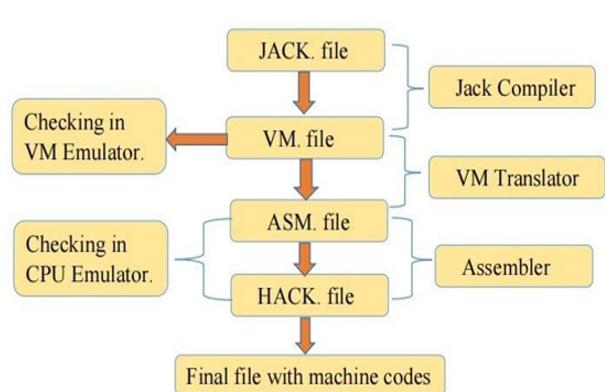
**6.3-Keyboard classes:**

This class allows reading the input from the keyboard.

1. **Function char keyPressed():** Returns the character of the currently pressed key on the keyboard ;if no key is currently pressed, returns 0.
2. **Function char readChar():** Waits until a key is pressed on the keyboard and released and echoes the key to the screen and returns the character of the pressed key.
3. **Function string readLine(string message):** Prints the message on the screen, reads the next line from the keyboard ,echoes the line to the  screen, and returns it's value. This method handles user backspaces.
4. **Function int readInt(string message):**Prints the message on the screen, reads the next line from the keyboard ,echoes the line to the  screen, and returns the integer until the first non-numeric character in the line. This method handles user backspaces.



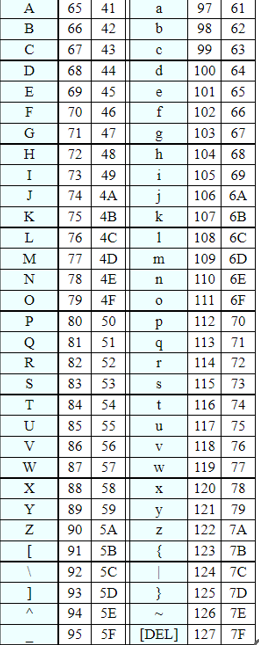
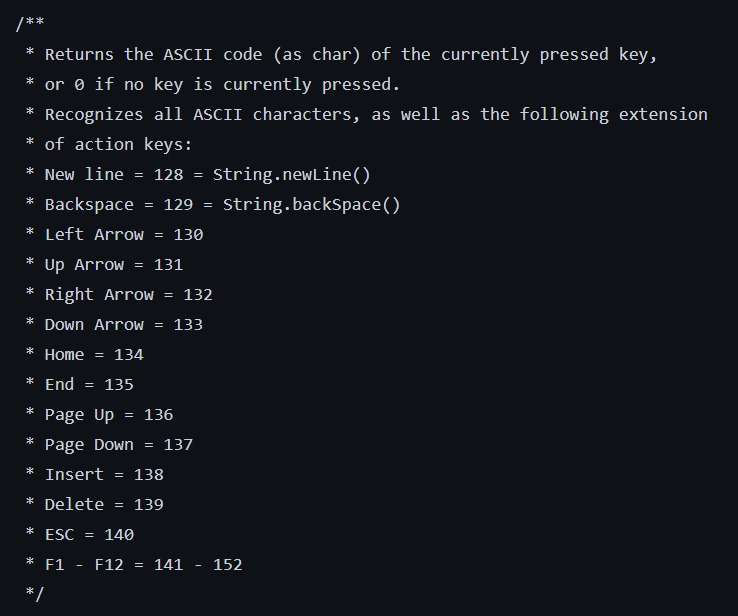
**6.3.1 - Conversion of files:**

****

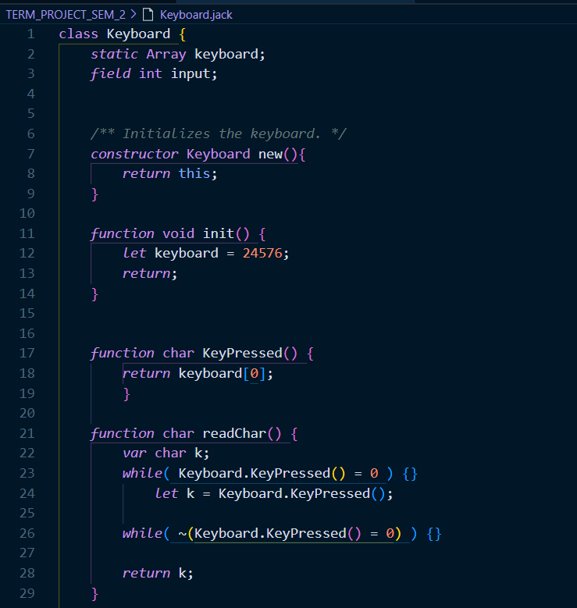
**ASCII CODE**

* ASCII stands for "**American standard code for information interchange**".
* ASCII is 7-bit character set containing 128 character.
* It contains numbers from 0-9 ,the upper and lower cases English  letter from A to Z and some special [character(!,@,#,%,\*...etc](mailto:character(!,@,)).
* The characters are used in modern computers ,Html and on internet are based on ASCII code.

**ASCII Table**

****

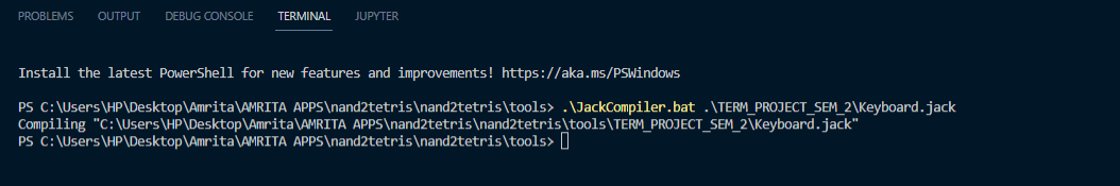
**Keyboard jack code**

****

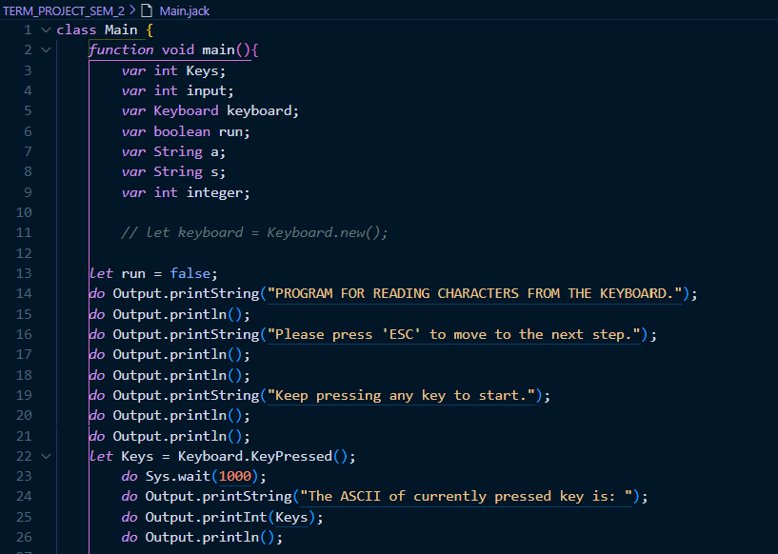
****

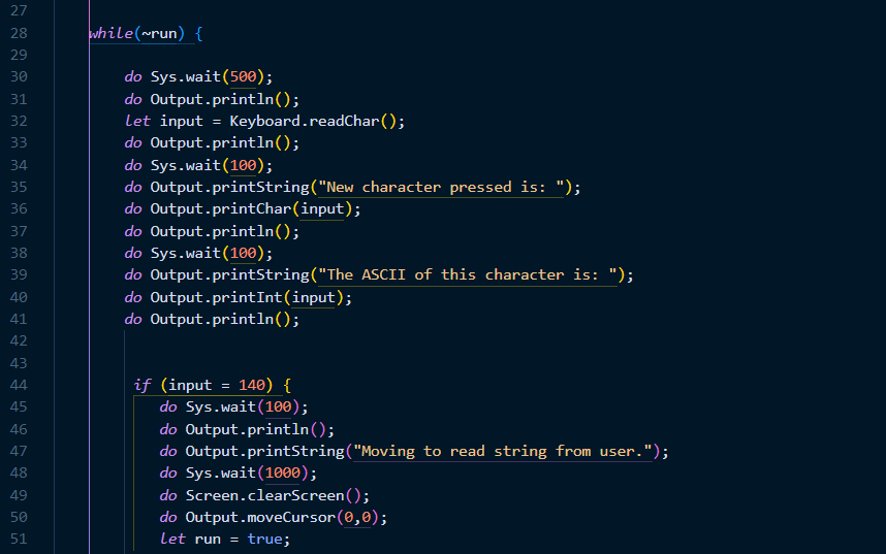
****

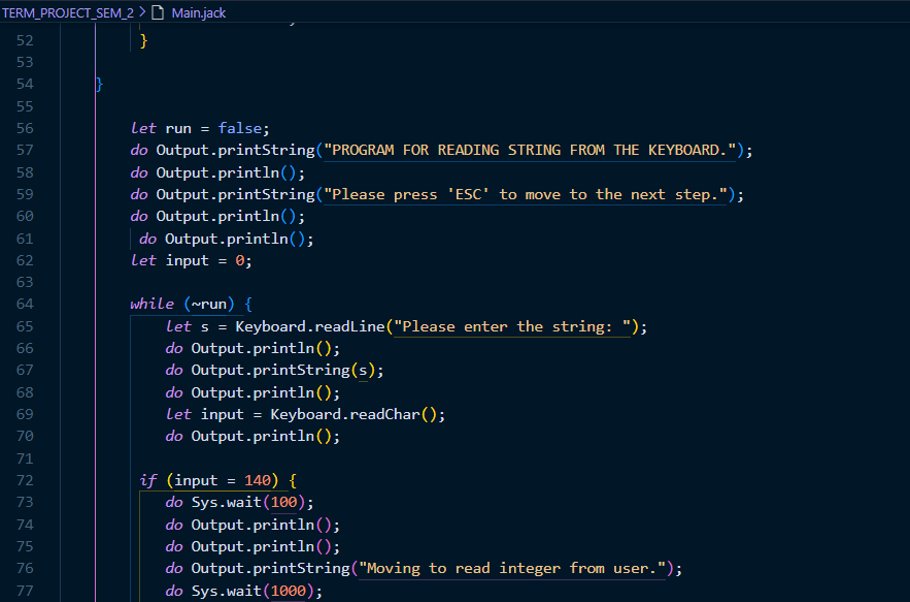
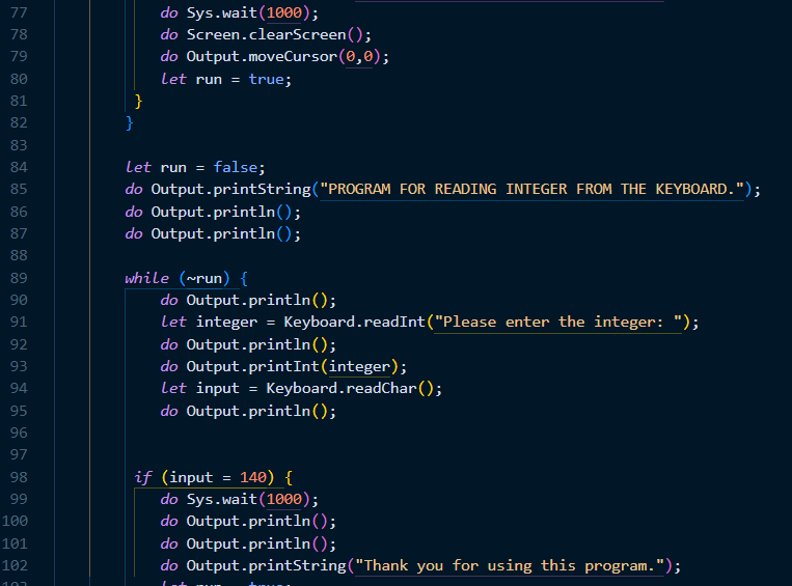
**Compiled keyboard code: -**

****

**Main Jack code: -**

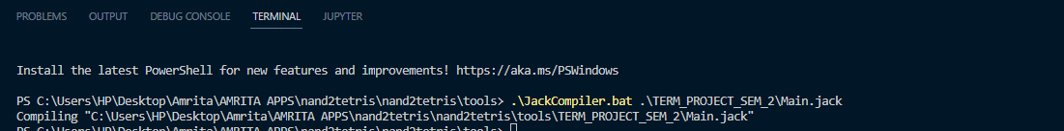






**Complied Main code : -**

****

**Main test file: -**

* We have a test code in nand2tertris to crosscheck the keyboard library we created.
* Location nand2tetris🡪projects🡪12🡪keyboardtest.



**Text

Description automatically generated**

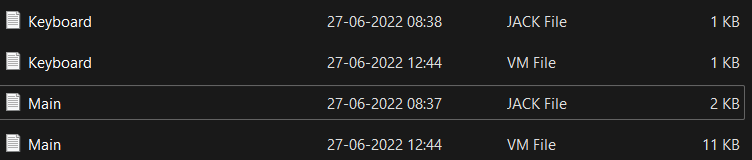
**Text

Description automatically generated**

**Text

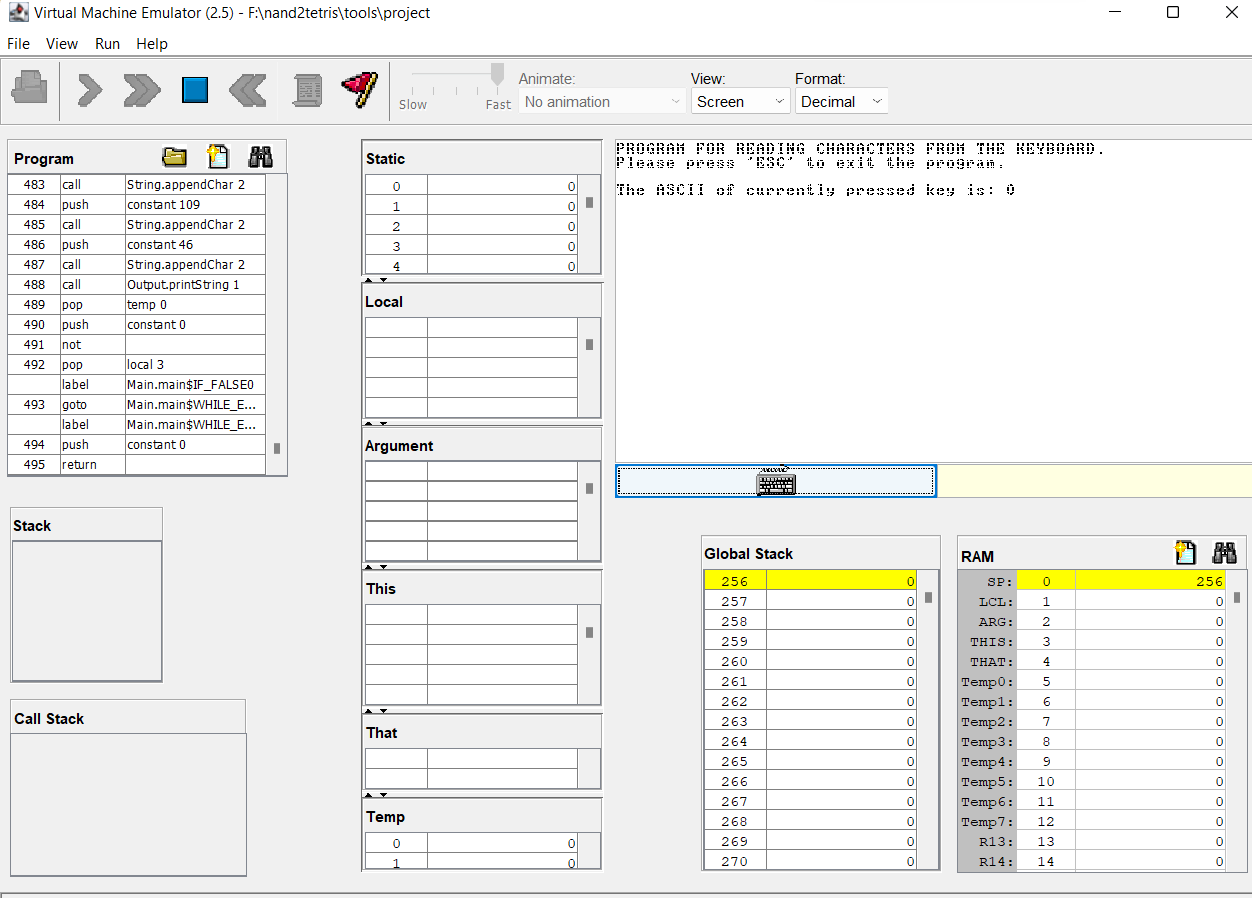
Description automatically generated**

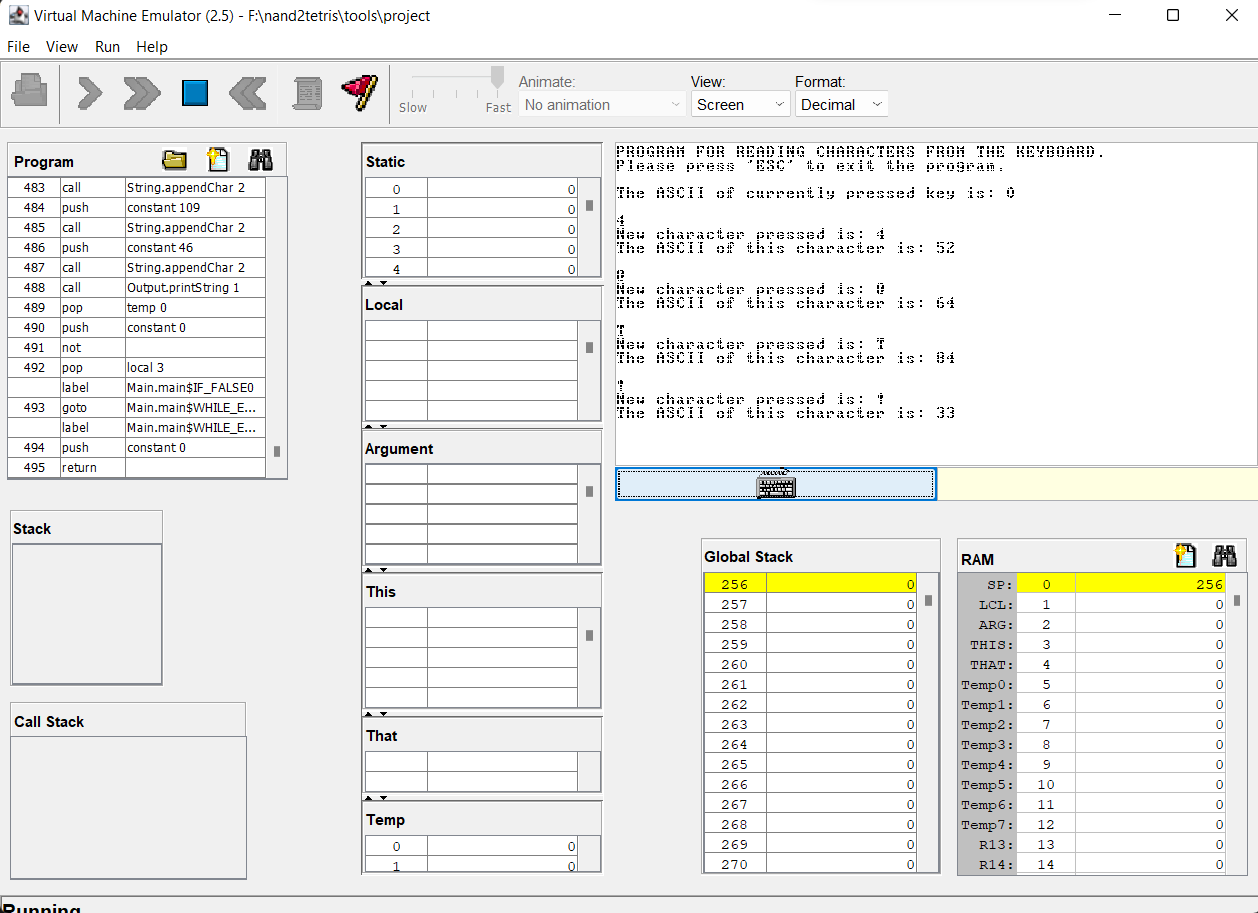
Keep all VM files in same folder

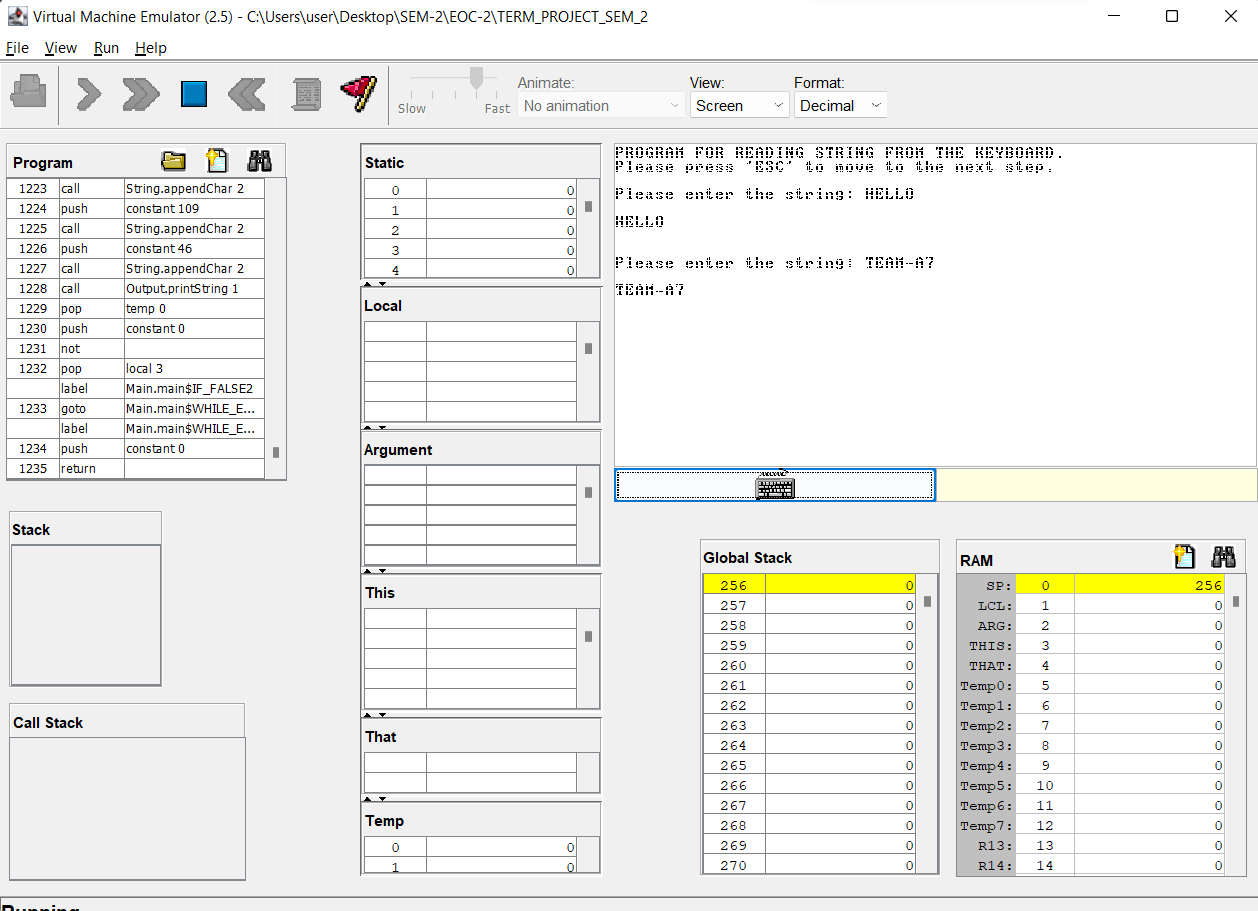


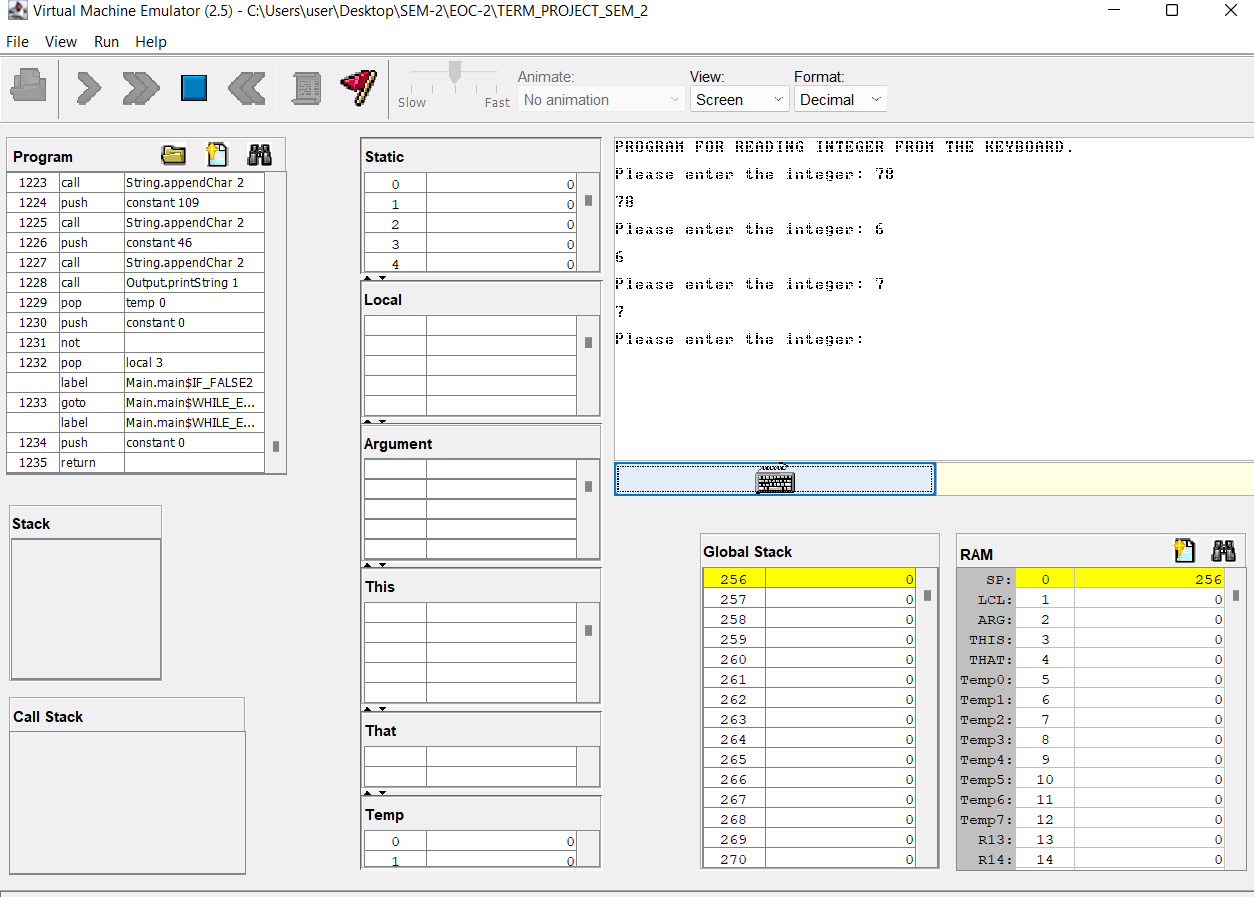
**VM Implementation**

* **Returns the ASCII code (as char) of the currently pressed key, or 0 if no key is currently pressed.**
* **Reads their next character from the keyboard: wait until a key is pressed and then released, then echos the key to the screen and returns the value of the pressed key.**
* **Reads their next character from the keyboard: wait until a key is pressed and then released, then echos the key to the screen and returns the value of the pressed key.**

****

* **Reads their next character from the keyboard: wait until a key is pressed and then released, then echos the key to the screen and returns the value of the pressed key.**
* **Prints the message on the screen, reads the next line from the keyboard ,echoes the line to the  screen, and returns it's value. This method handles user backspaces.**

****

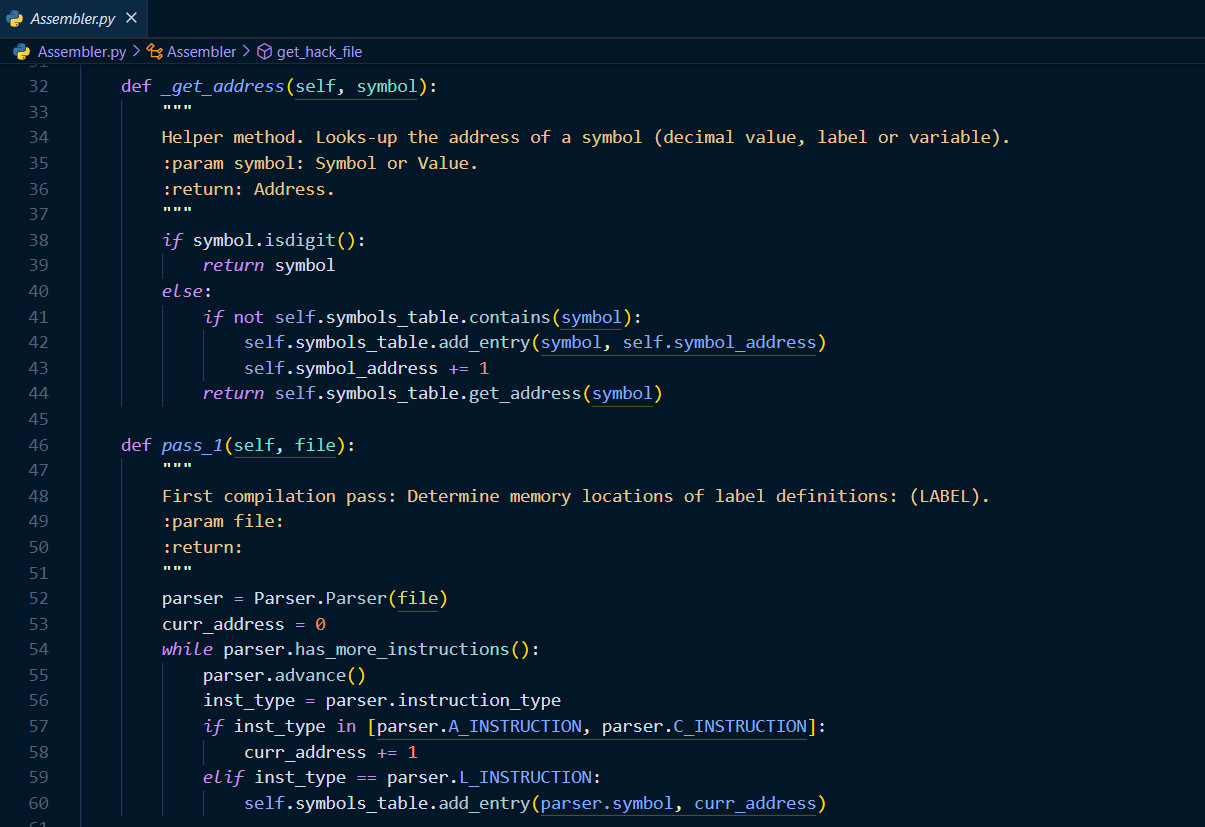
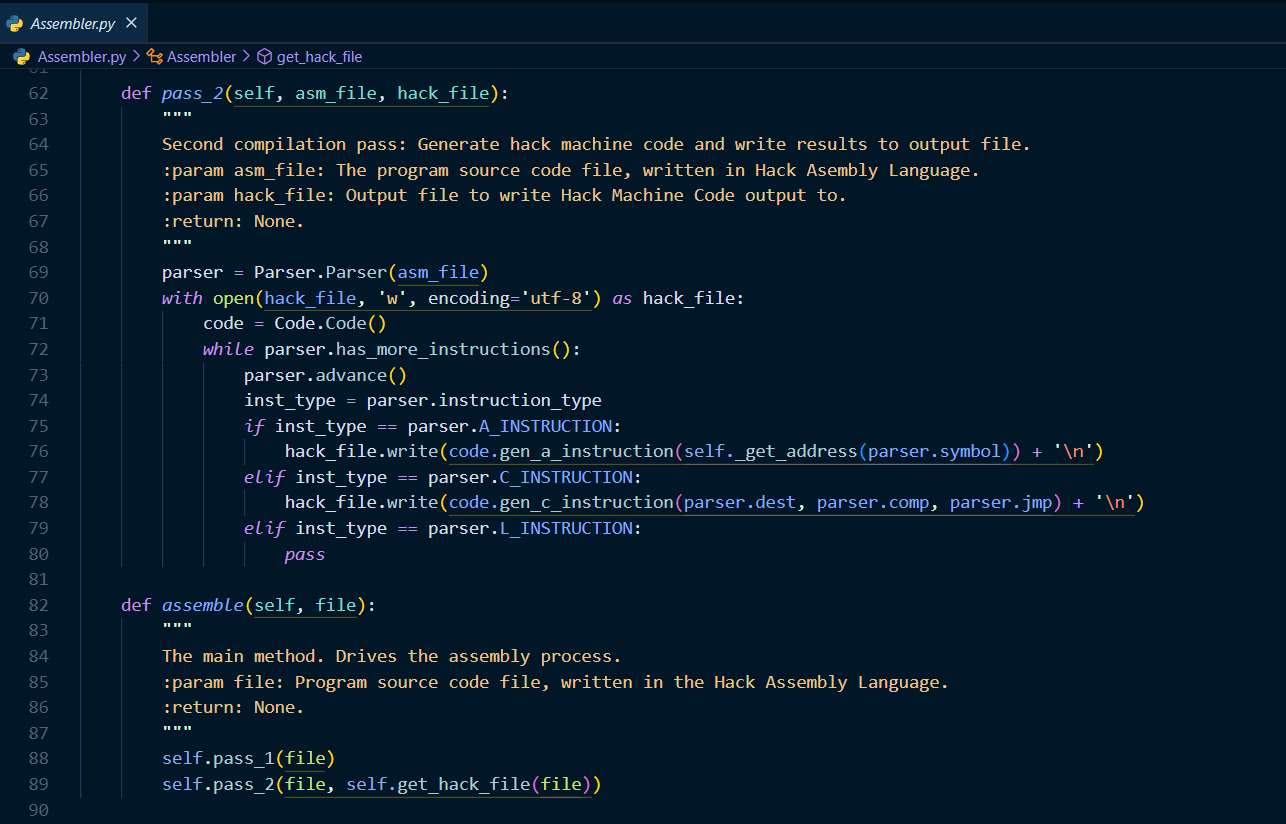
**Prints the message on the screen, reads the next line from the keyboard , echoes the line to the  screen, and returns the integer until the first non-numeric character in the line. This method handles user backspaces**

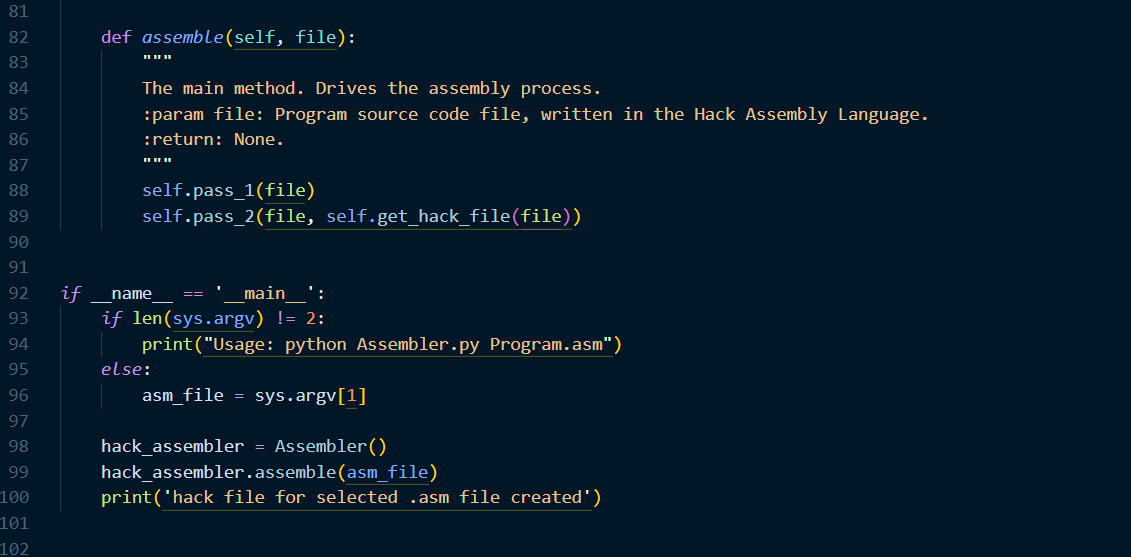
**Part-2**

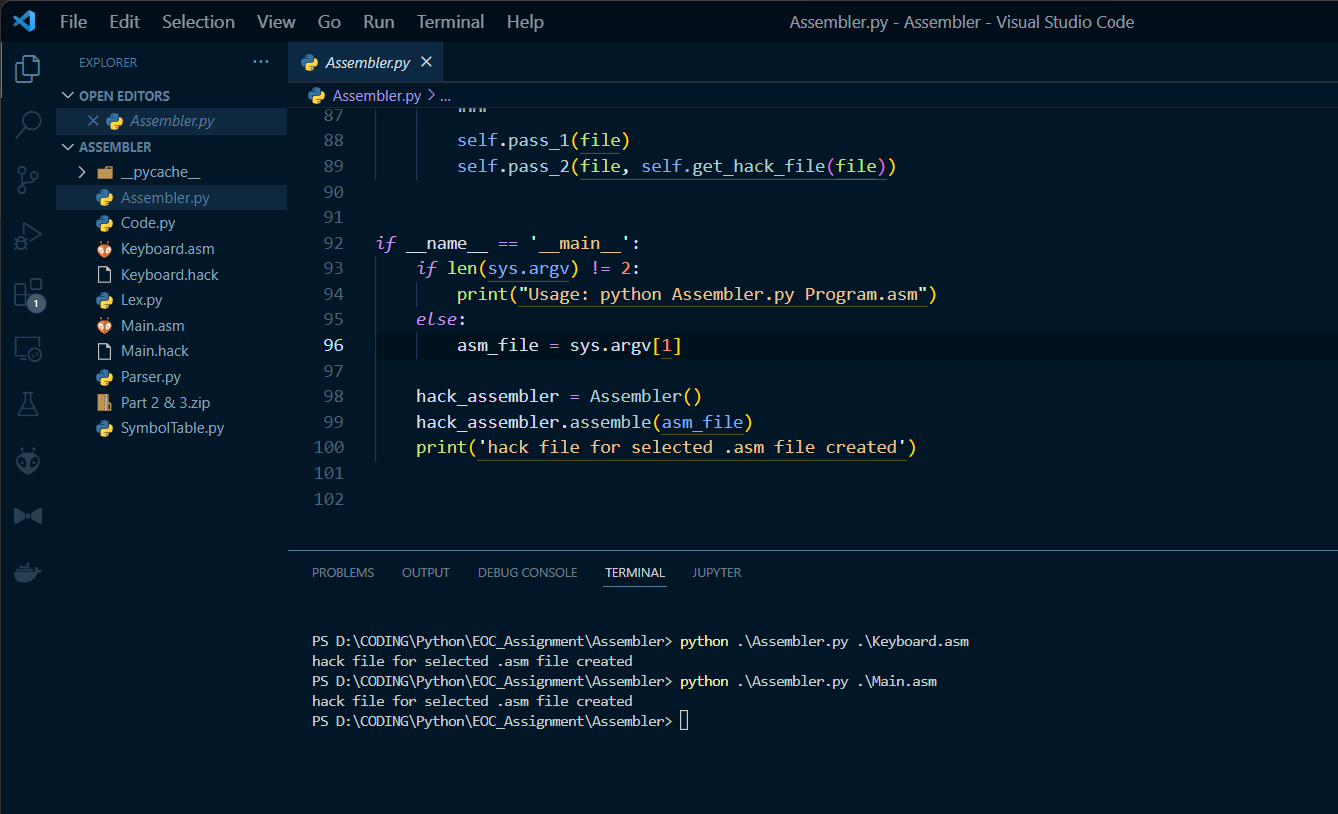
* **Write your own general purpose assembler to convert the assembly program into 16 bit machine codes also use the Nand2tetris assembler for conversion. (Your own assembler should be developed by you and is different from the assembler tool supplied along with the book) .**

**ASSEMBLER CODE**

****

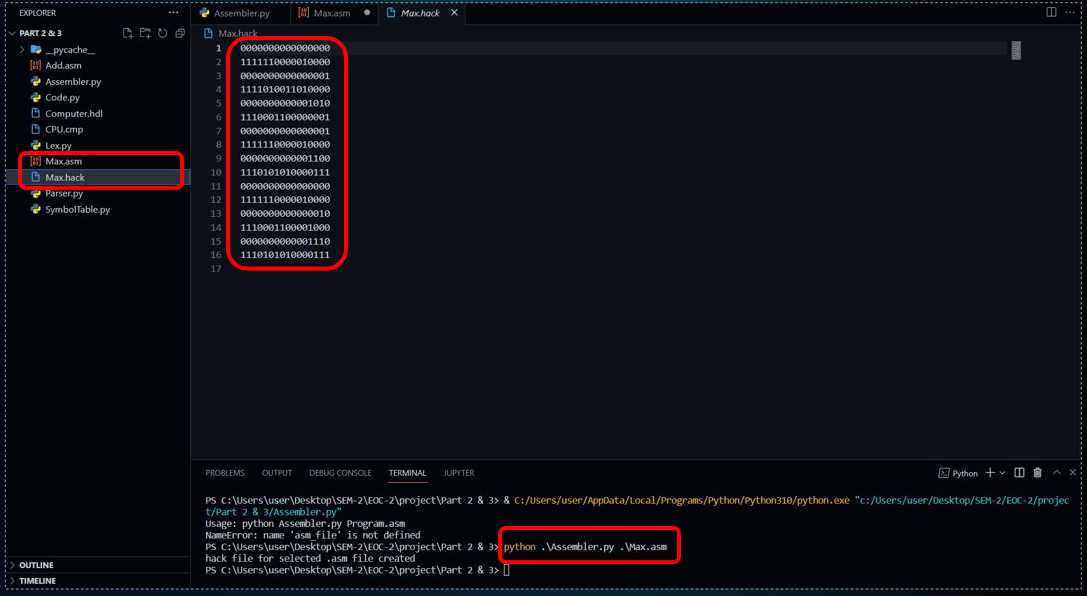
****



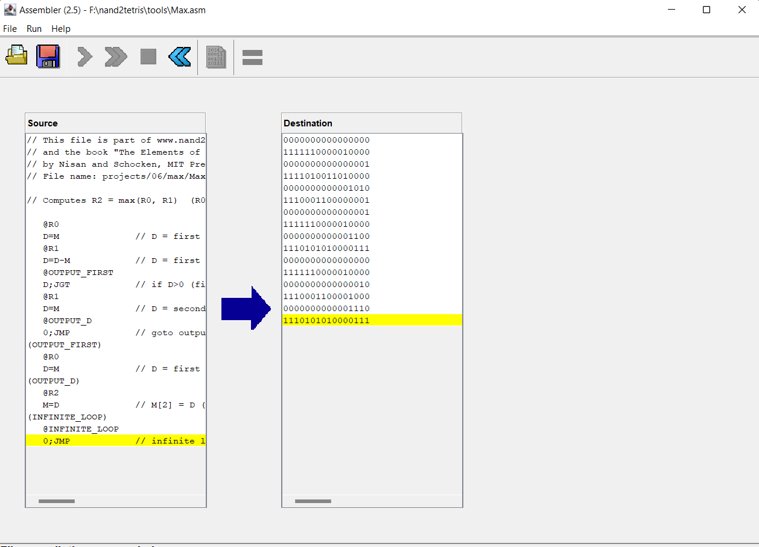
****

* Here we created our own assembler to convert .asm files to .hack files.
* We need to keep the files in the same folder in order to convert .asm files to .hack files.

Now in the terminal we use the command "python .\Assembler.py .\Max.asm" to convert asm.file to hack.file.

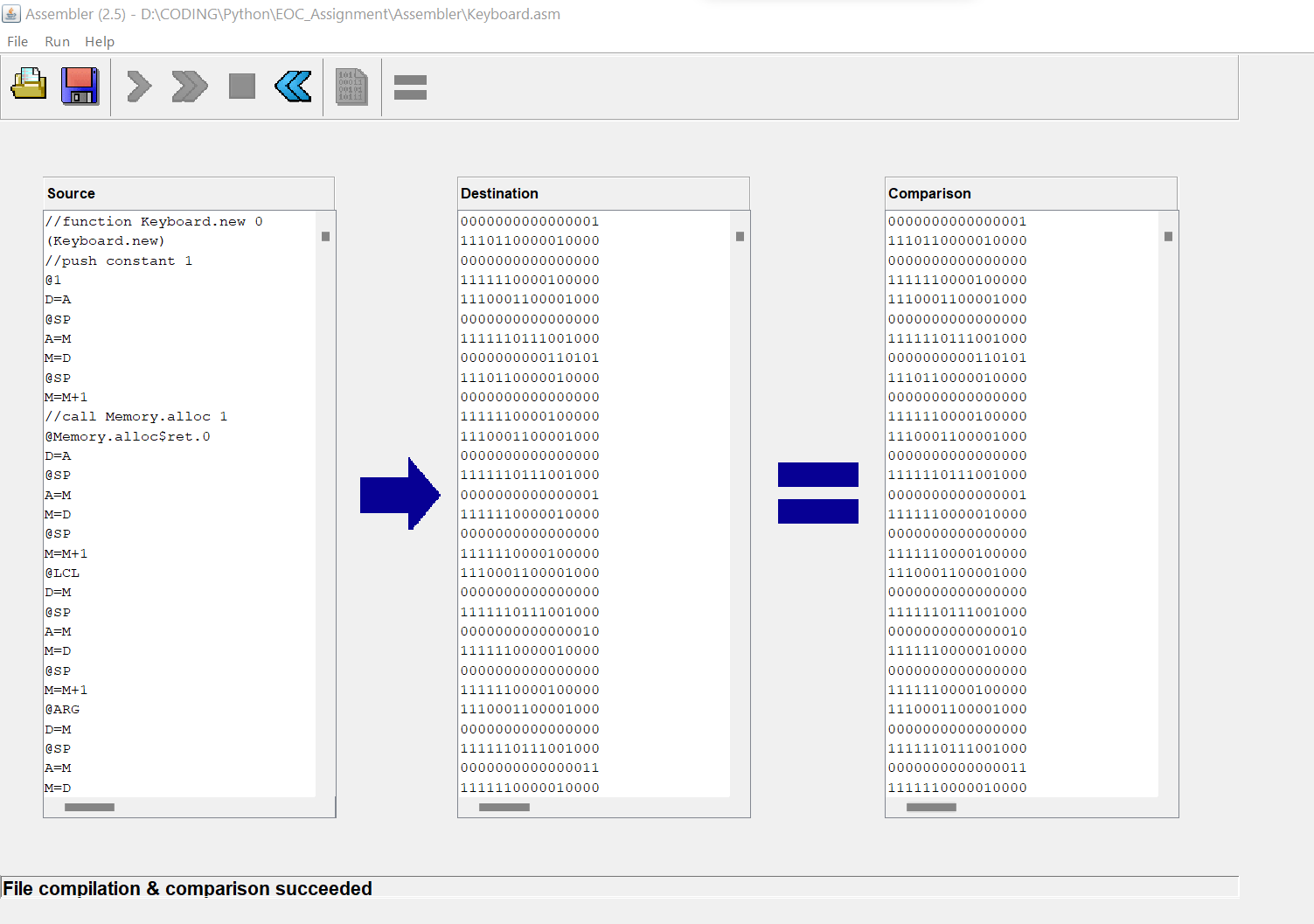
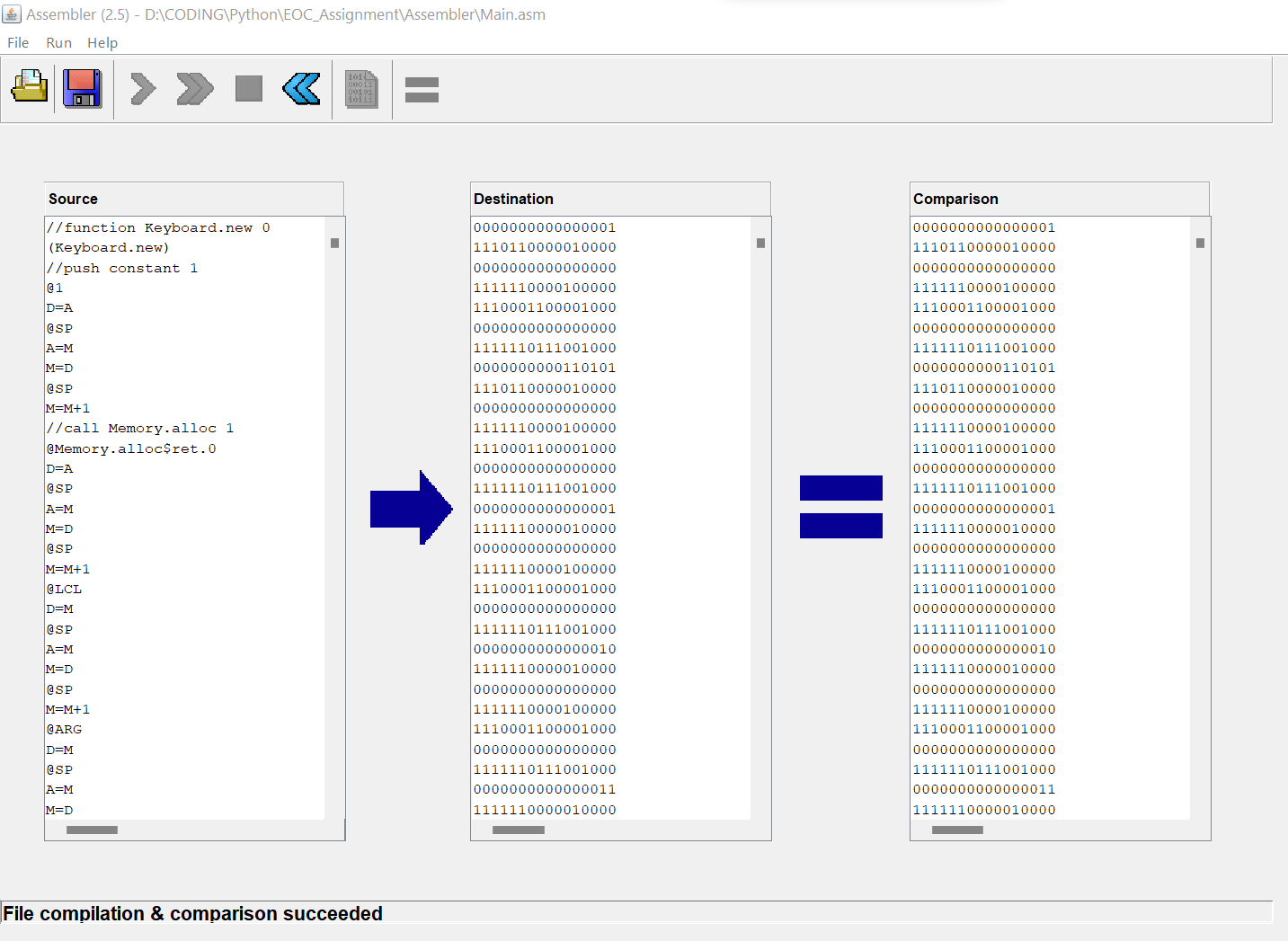
****

* After doing the above process we can see a new file in the folder named as “Main.hack” which is a hack file.
* Now we are going to use assembler tool in nand2tetris to do the same process of converting asm file to hack file.
* Open the assembler and load the asm file and then click on the run button to convert it into hack language, then save the file.



* If we equate both the hack files that we obtained earlier using own assembler and nand2tertis are identical.

**COMPARISON PHASE**

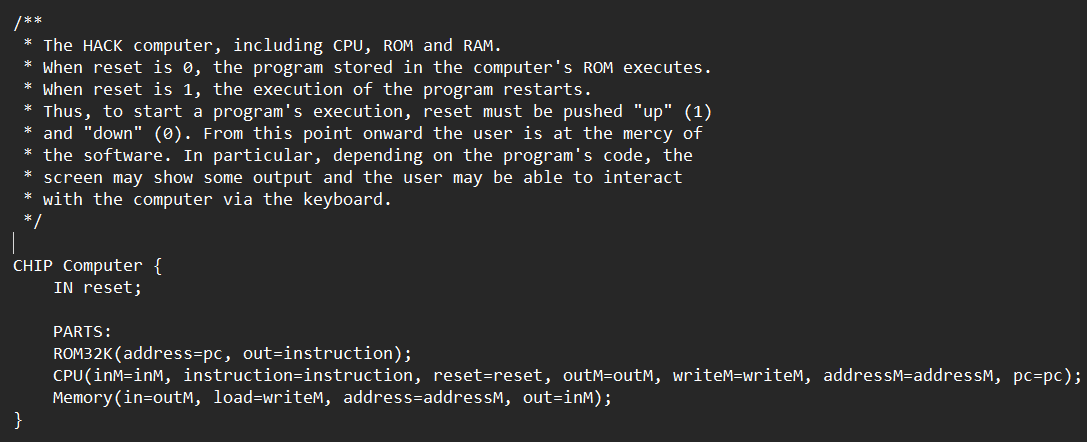
**** ****

**Part-3**

**Dump the machine codes generated by your own assembler into the RAM.hdl. Then design the architecture for implementing your machine instructions to get the result. (Here I presume that you already have the hdl scripts of all the hardware’s used by the CPU. Additional thing what you have to do is connecting the address and data buses during the execution).**

* **What is meant by 16 bit instructions?**

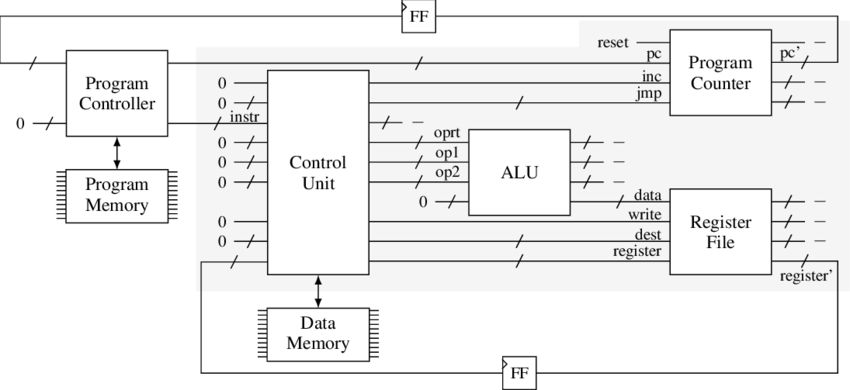
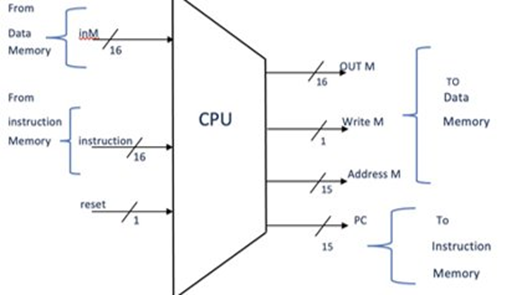
1. There are two sorts of 16-bit instructions: A instructions and C instructions. Now, from the 0th to the 15th bit, if the first bit (i.e., the T-pin) is zero, it is an A instruction; if the first three bits are 1 1 1, or if the addition of the first bit is certain, it is a C instruction.
2. If the first bit is zero, the instruction is an A; if the first bit is one, the instruction is a C-instruction.
3. For the A instruction, the remaining 15 bits are value, which is saved in the A register and then outputs the value.
4. However, if I have a C instruction, then everything is entirely different. In this case, the 13 instruction bits are decoded into the Op-code (treat it as letter "a"), ALU control bits, Destination load bits, and Jump bits.

**Computer hdl file**

**Central Processing Unit(C.P.U)**

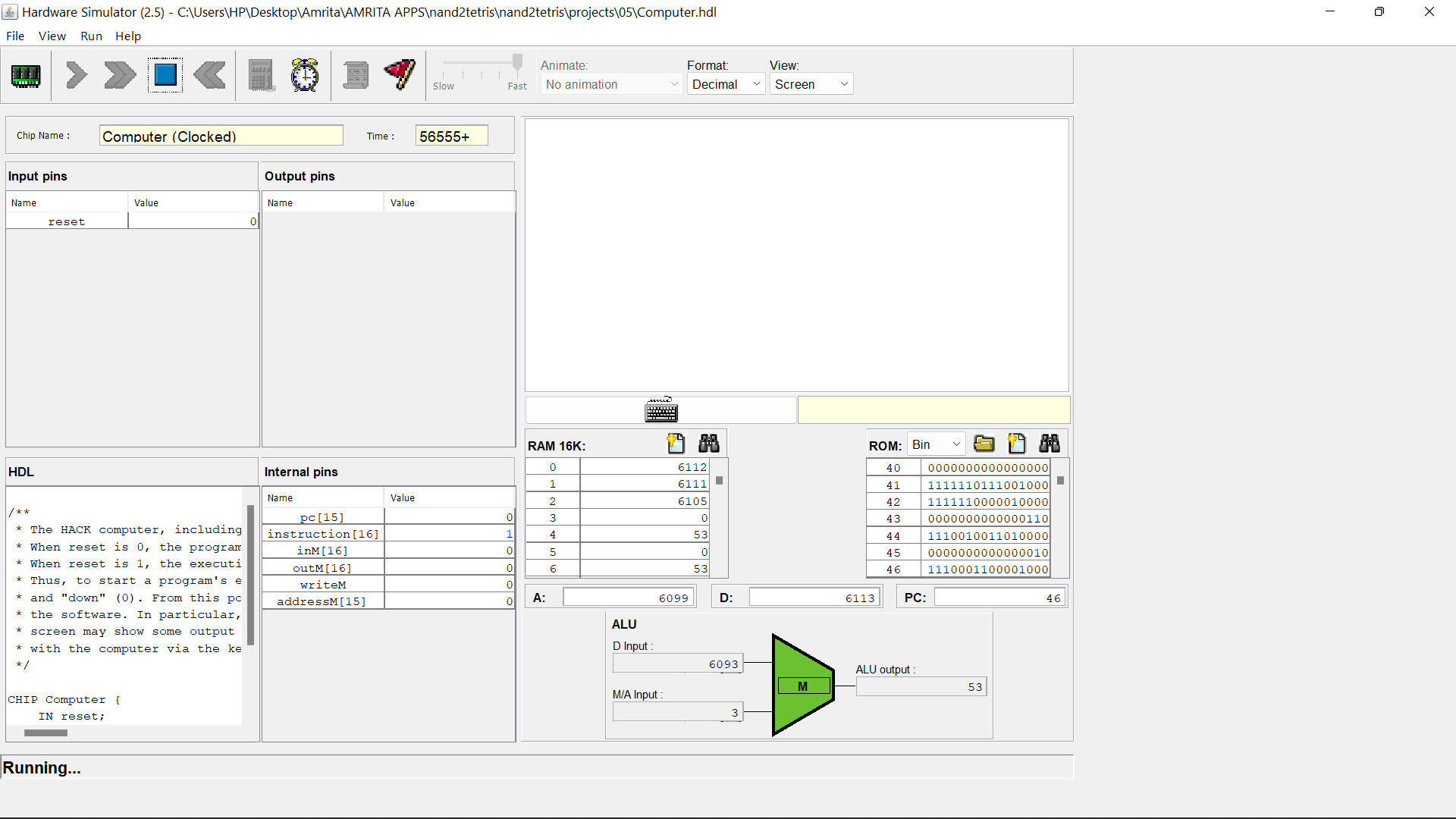
* The control unit is in charge of all CPU functions, including ALU operations, data transfer inside the CPU, and the interchange of data and control signals across external interfaces.
* The control unit is in charge of all CPU functions, including ALU operations, data transfer inside the CPU, and the interchange of data and control signals across external interfaces.

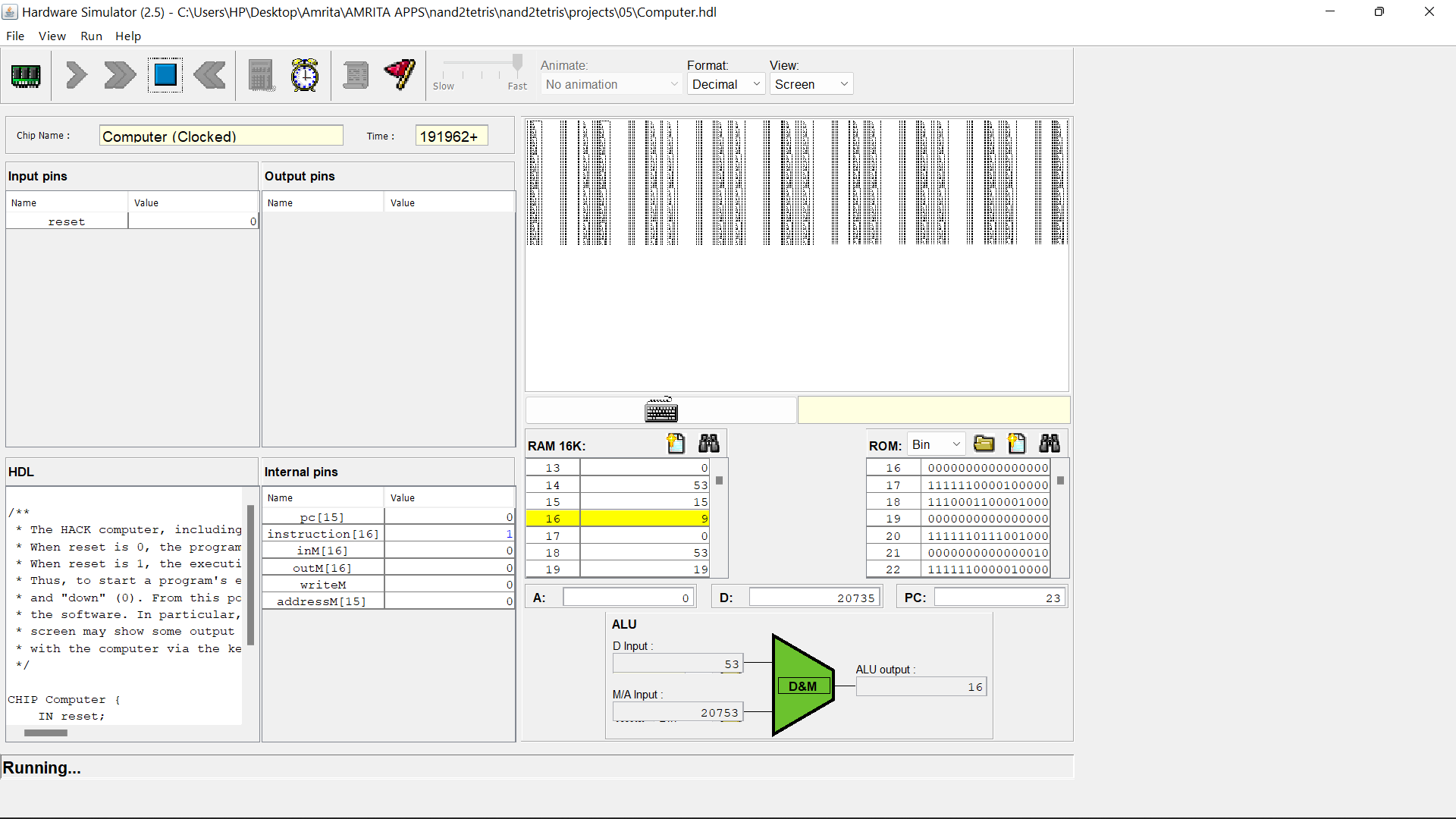
**CPU DESIGN**

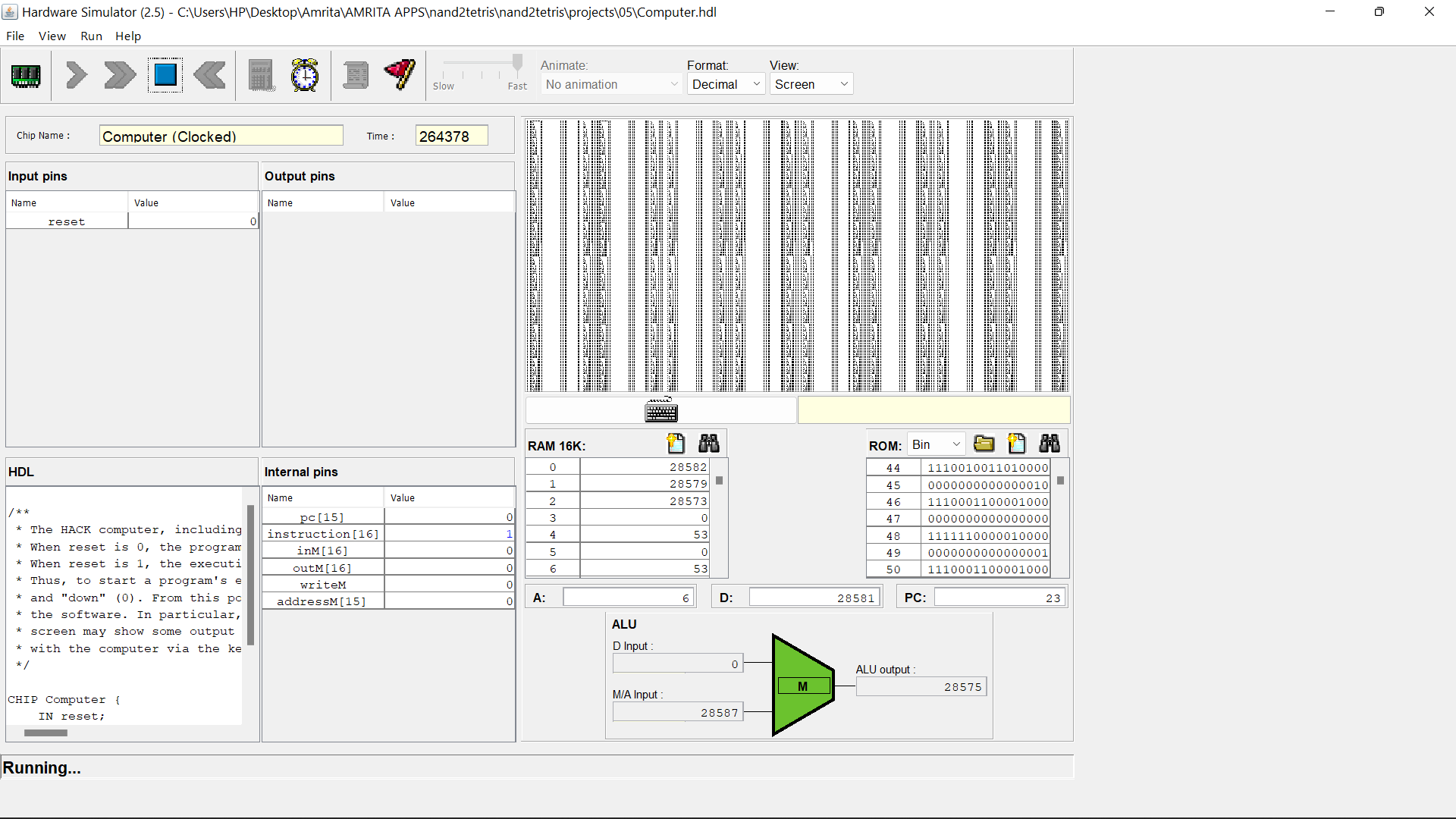
** **

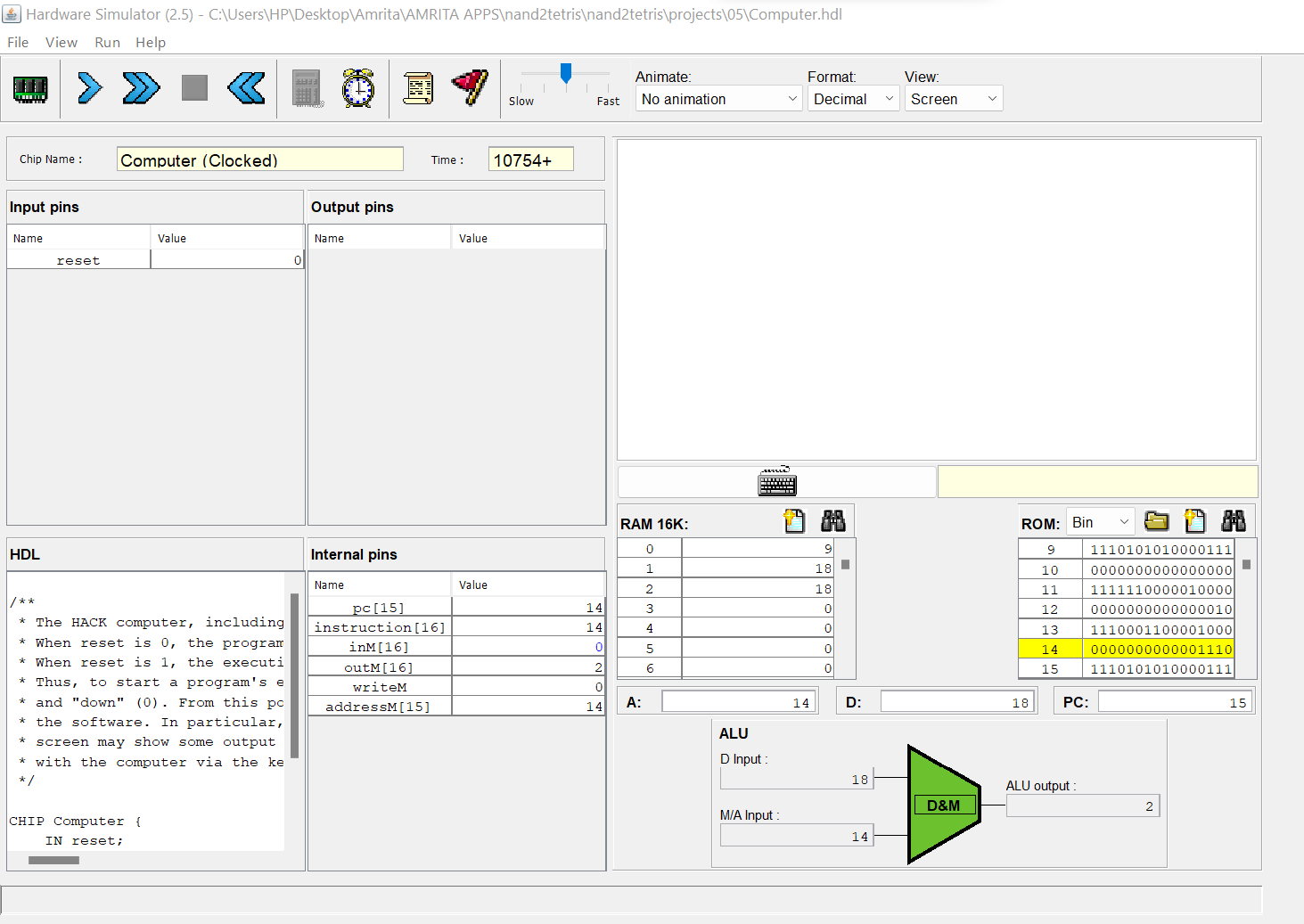
* **Instruction Memory -**It contains a 16 bit ‘instruction’ as input and the output of PC can be explained based on Instruction memory.

**Working with .hack file of KEYBOARD & MAIN**

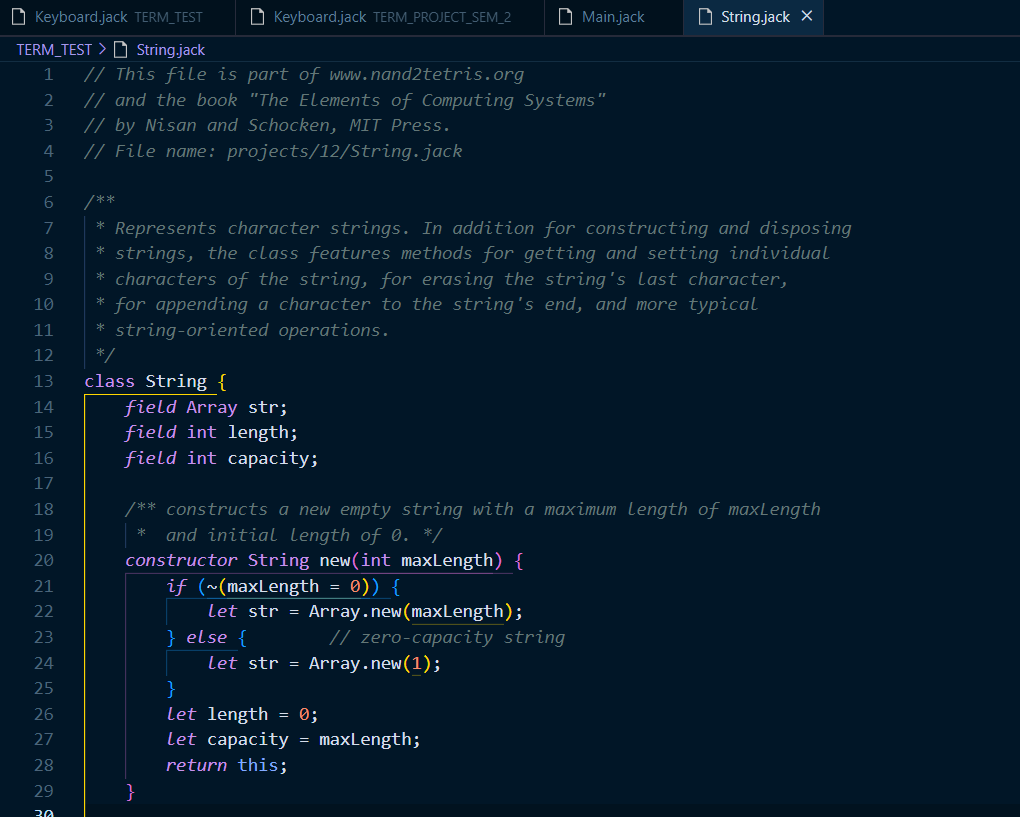
****

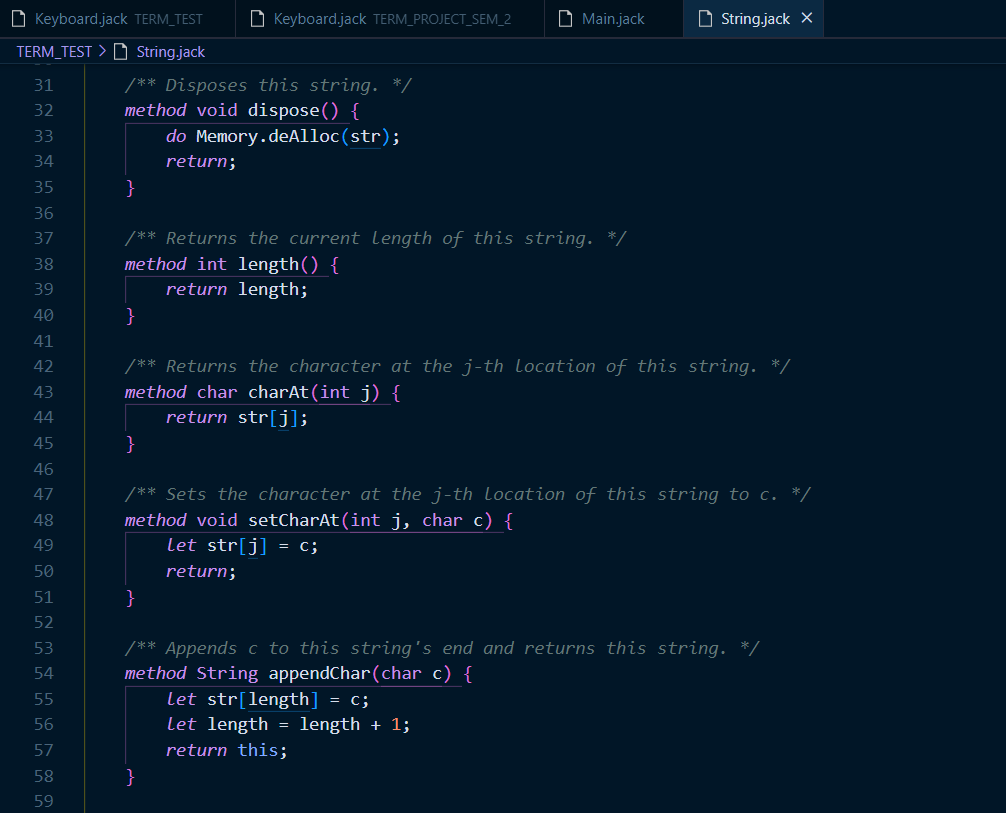
****

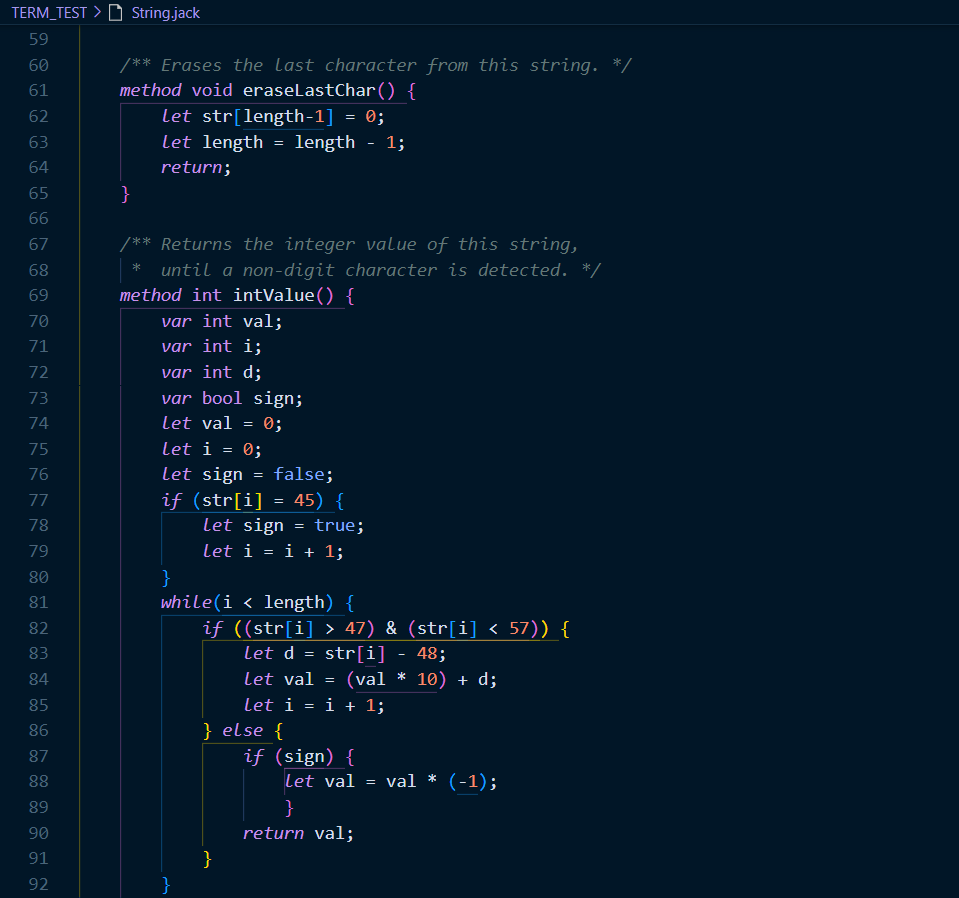
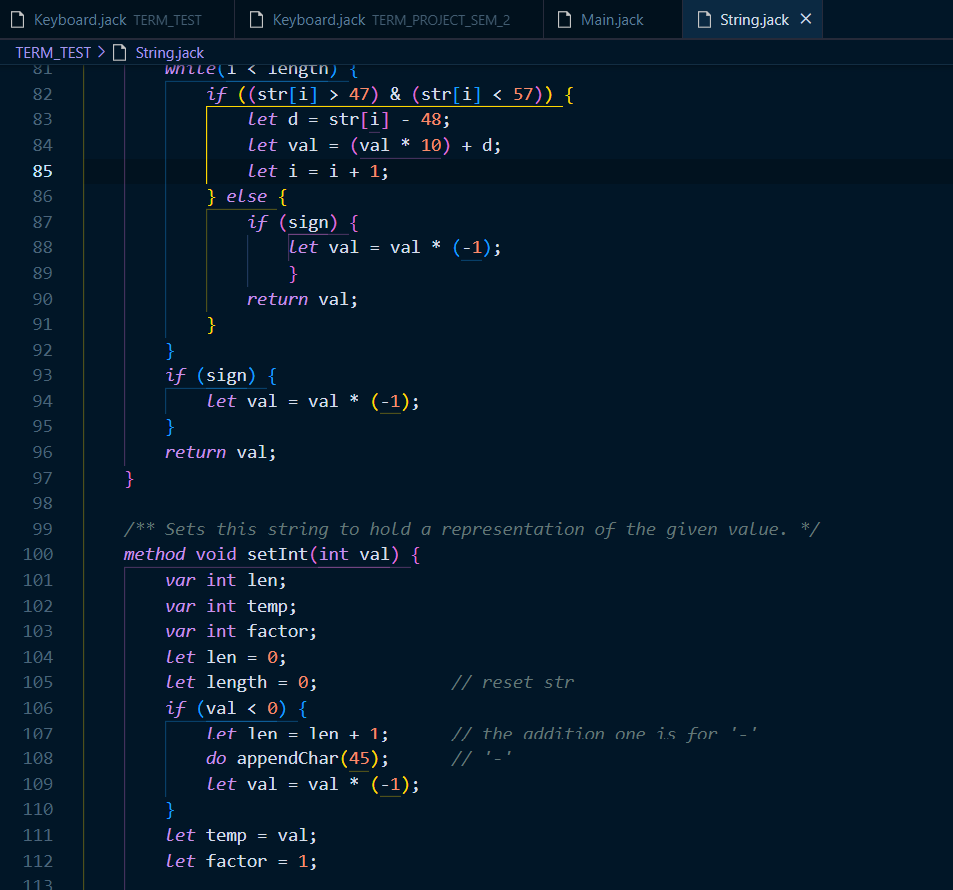
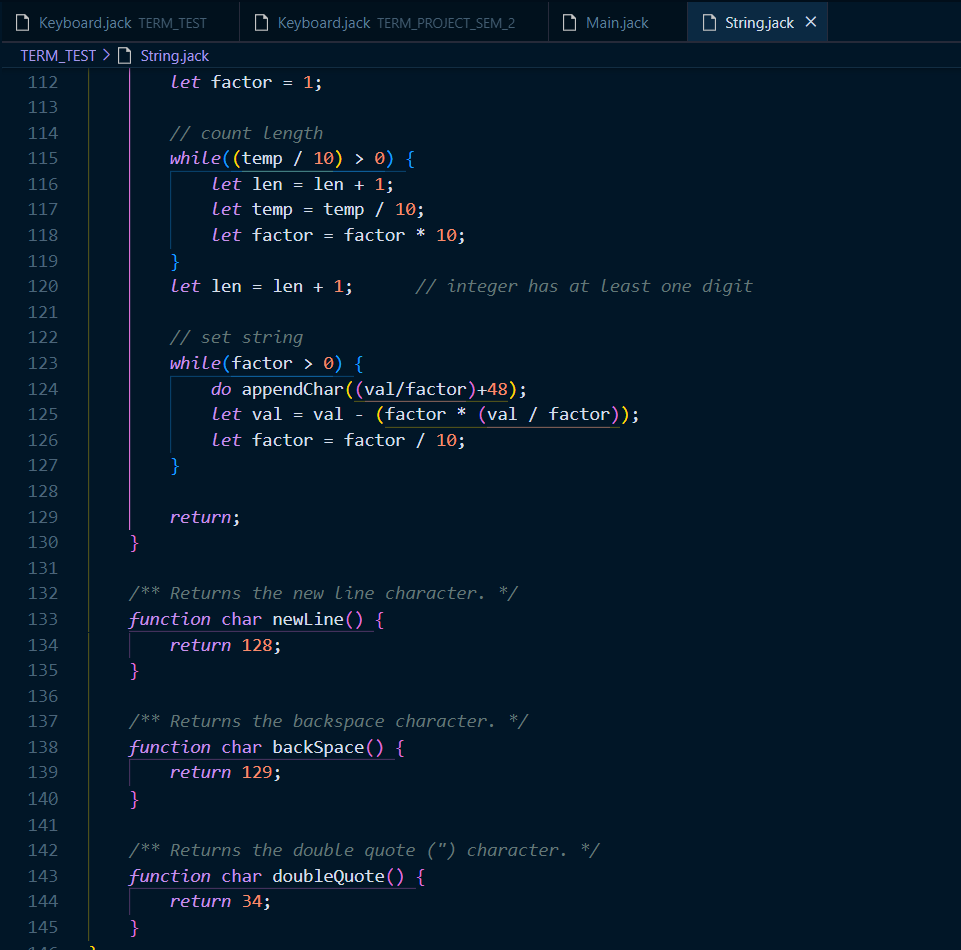
**Error**When we load the code on hardware simulator and run that code we can see that code will never stop, will continue in an infinite loop but will never give desired output.  
The reason behind this error is the information transfer and working of nand2tetris package as a whole, ASCII characters, calling of different files is not possible in terms of real time working, even though code gets converted into hack from .jack file.  
So we are going to use a simple alternate code of MAX which will compare two inputs and displays the maximum number out of the two input.  
  
  
  
  
  
  
  
  
  
 **ALTERNATE CODE[MAX HACK]**

****

**EXTRAS**

I (Vikhyat Bansal) and my friend (Rachit Agarwal) of BATCH\_B TEAM\_5 together created string and keyboard library. String library is an important library which can be used for implementation of keyboard library function readInt() directly using String.IntValue(). **String JACK CODE  
**

****

**** **** 

**Conclusion**

From this project we have learned working on JACK language and how JACK language can be used to create IN-BUILT library(keyboard) within a time frame with much more functionalities, without any prime limitations.  
Working of assembler is understandable on a whole new level where one can simply create a nand2tetris software with basic knowledge of HIGH-LEVEL programming language.  
Generating code in BITS and working with 16-BIT code is done on much easier level which can be tested using software available in nand2tetris folder.