ELEMENTS OF COMPUTING SYSTEMS - 2

TEAM MEMBERS

BATCH-A	TEAM-7
GAJULA SRI VATSANKA	CB.EN.U4AIE.21010
GUNNAM HIMAMSH	CB.EN.U4AIE.21014
M.PRASANNA TEJA	CB.EN.U4AIE.21035
VIKHYAT BANSAL	CB.EN.U4AIE.21076

1.

Write a Jack program to execute the following equation

If n <=10, y=Σ n

Else $y=\Sigma n^2$

CODE LOGIC

- We will take a variable n whose value we will take as input from user.
- If n is less than or equal to 10 then it will give summation of n from 0 to 10
- If n is more than 10 then output will be given as summation of n^2 from 11 to till number.

NOTE:

Value of n will be calculated till when n = 45 as soon as value of n is taken as n = 46. It will start the count in negative as 16 bit number integer range is from -32768 to +32767.

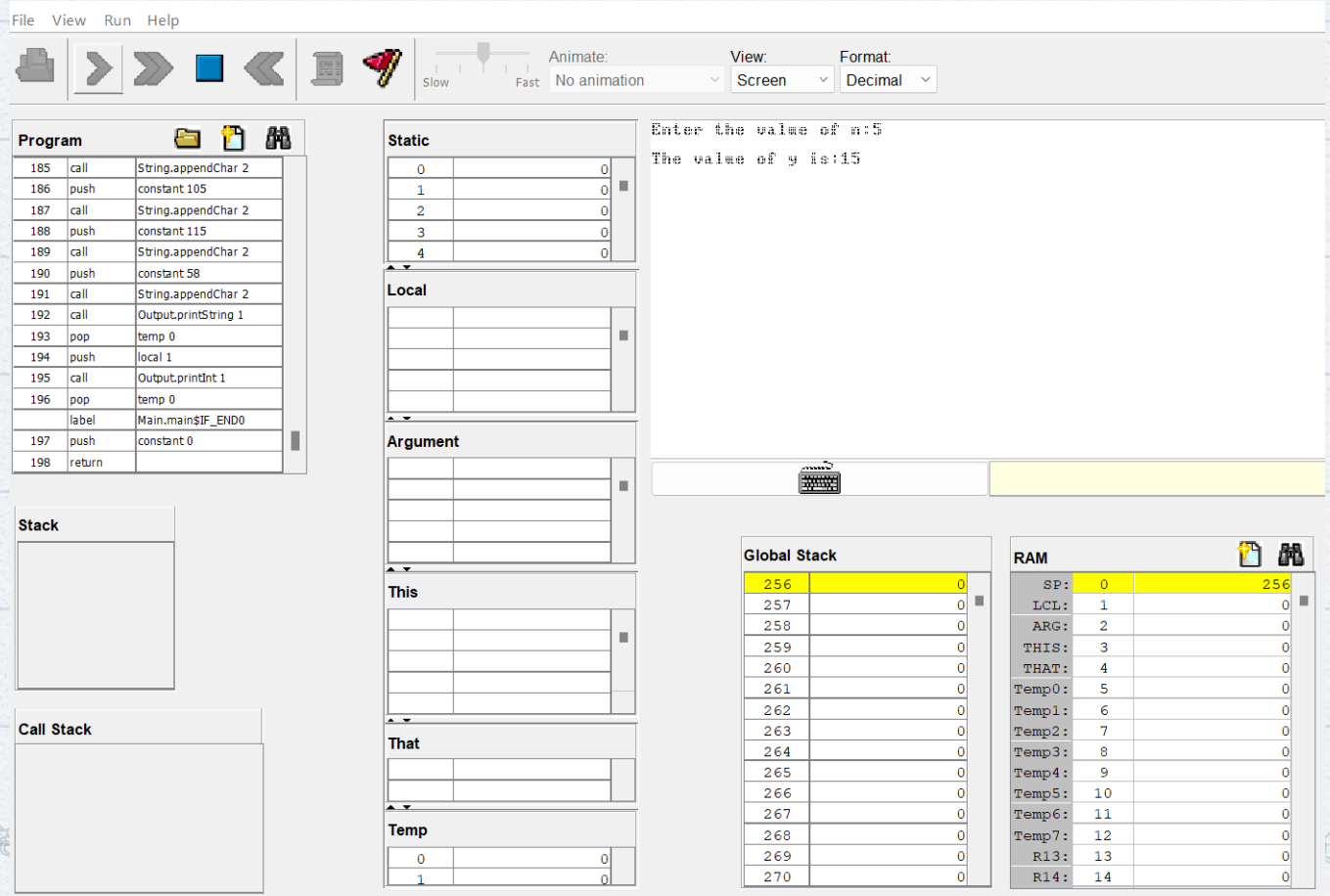
J ACK CODE

```
class Main {
    function void main() {
        var int n;
        var int y;
        var int i;
        //var int sum;

        let n = Keyboard.readInt("Enter the value of n:");
        do Output.println();

        let y = 0;
        if ((n<10) | (n=10)) {


            let i = 0;
            let y = 0;

            while (i>-1 & i<n) {
            let i = i+1;
            let y = y+i;

            }
            do Output.printString("The value of y is:");
            do Output.printInt(y);
            do Output.println();

        }
```

```
    else {
        let i = 0;
        let y = 0;

            while  (i>-1 & i<n) {
            let i = i+1;
            let y = y+(i*i);
        }
        do Output.printString("The value of y is:");
        do Output.printInt(y);

    }
    return;
    }
```

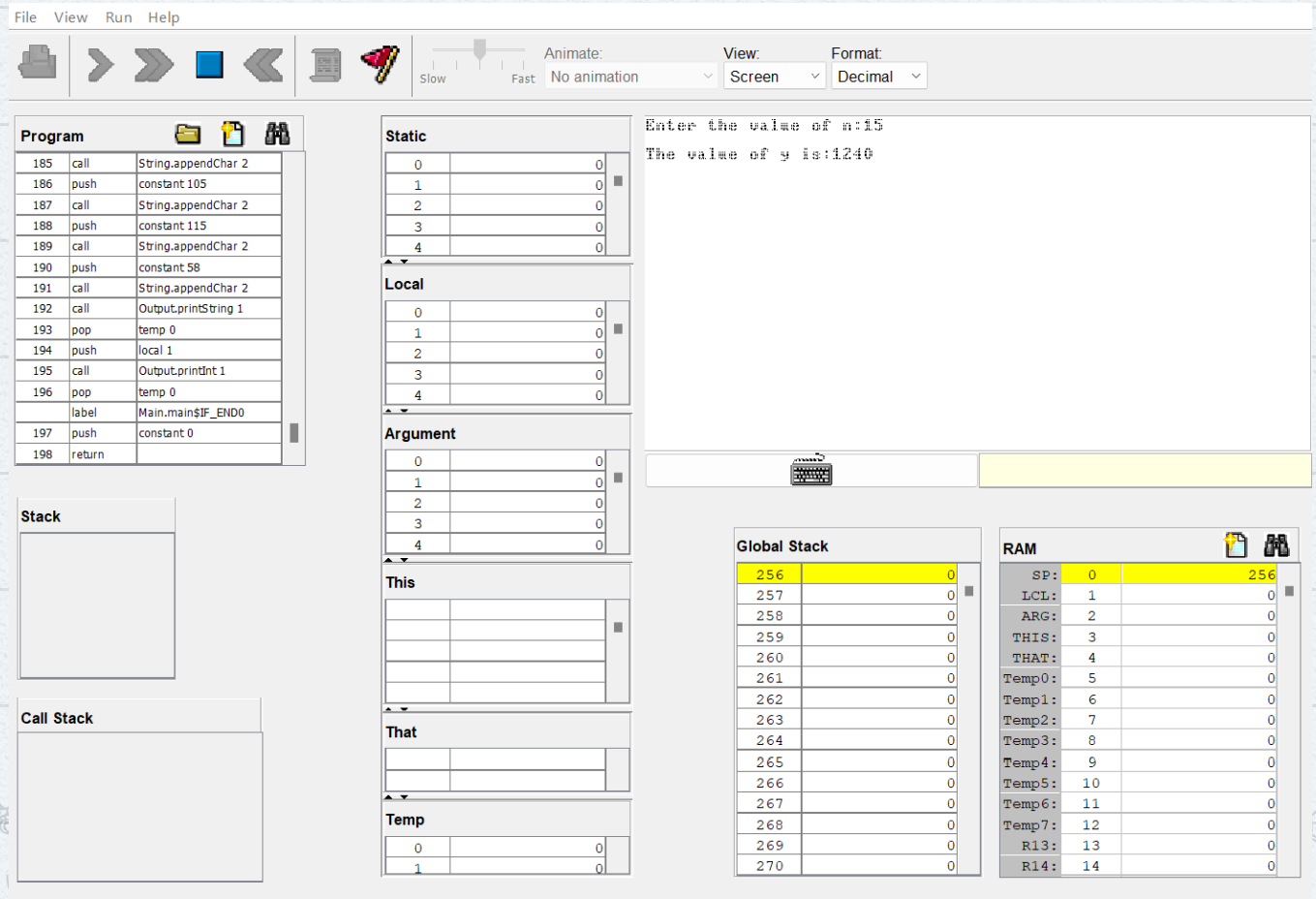
JACK COMPILATION

OUTPUT



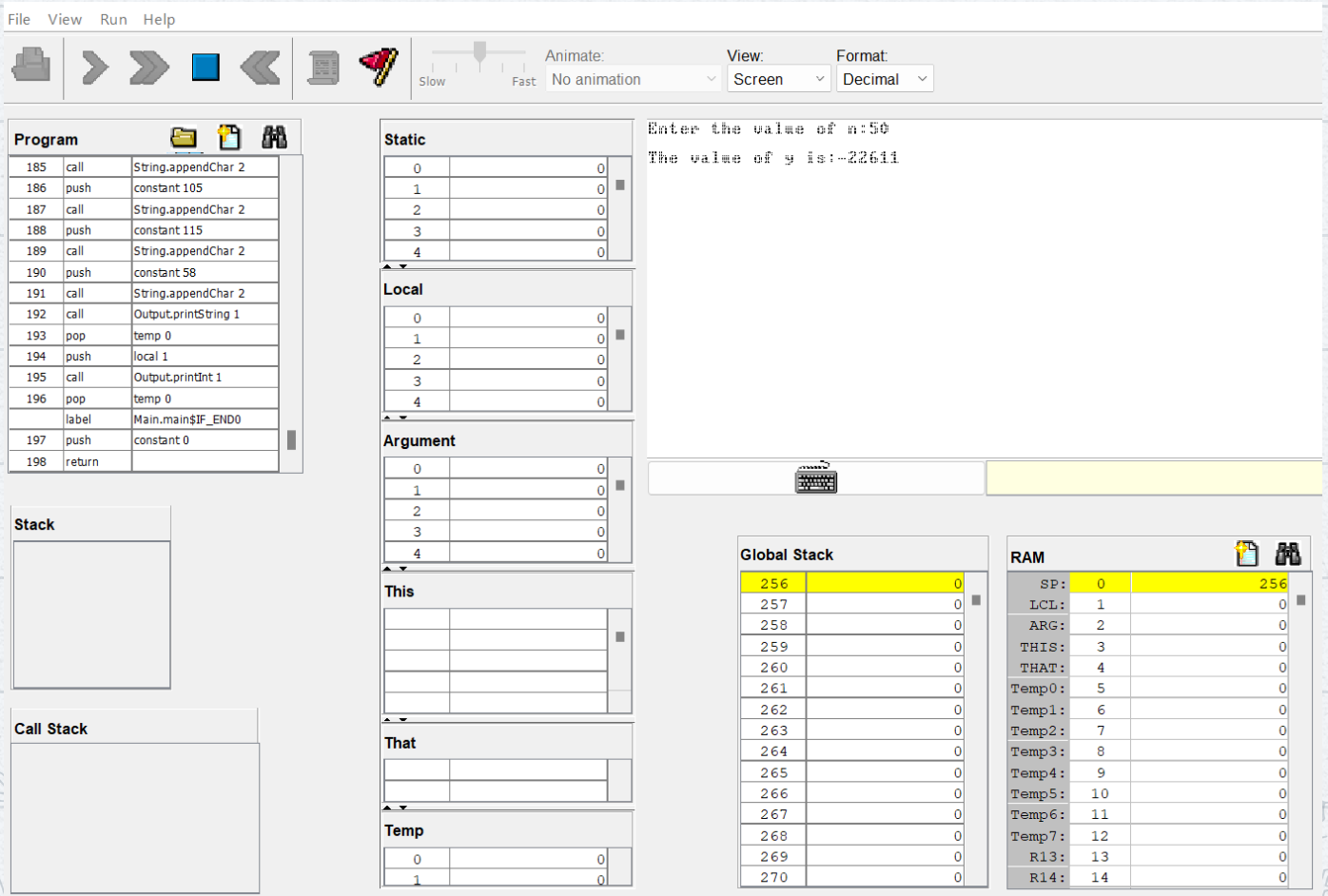
When n = 5

OUTPUT



When n = 15

OUTPUT



When n = 50

2. Write a Jack program which counts up to 30 sec and automatically resets to 0.

CODE LOGIC

- We will write a jack code which will count from 0 to 30 with a delay of 1000 millisecond (1second).
- When counter hits 30 the counter will reset and again start from 0.

JACK CODE

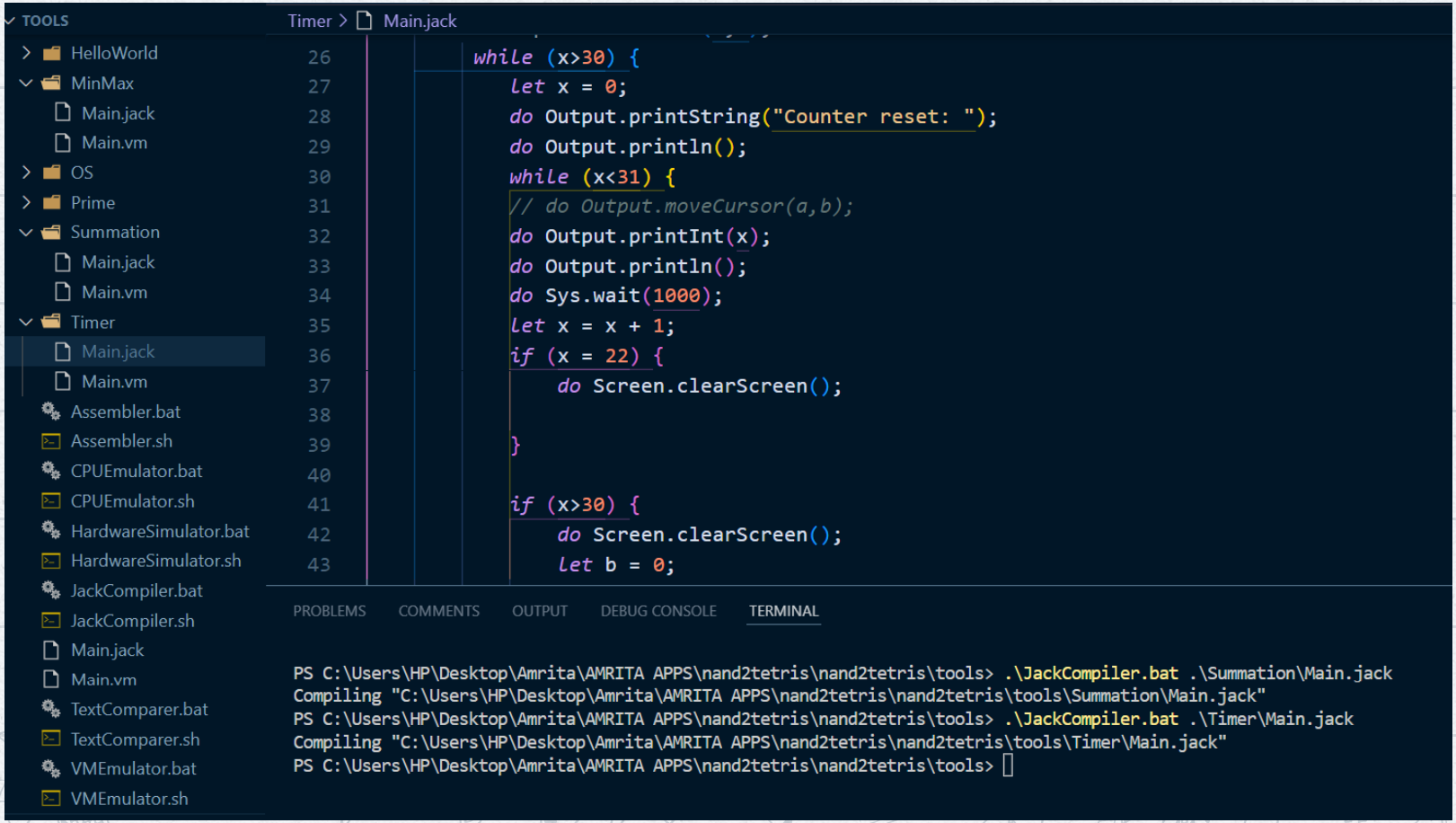
Timer > Main.jack

```
class Main {
    function void main() {
        var int a, b, c;
        var int x, y, z;
        let x = 0;
        let a = 1;
        let b = 0;
        do Output.printString("Counter Started: ");
        do Output.println();
        while (x<31) {
            // do Output.moveCursor(a,b);
            do Output.printInt(x);
            do Output.println();
            do Sys.wait(1000);
            let x = x + 1;

            if (x = 22) {
                do Screen.clearScreen();

            }
        }
        do Screen.clearScreen();
        let b = 0;
        let a = 0;
        do Output.moveCursor(a,b);
```

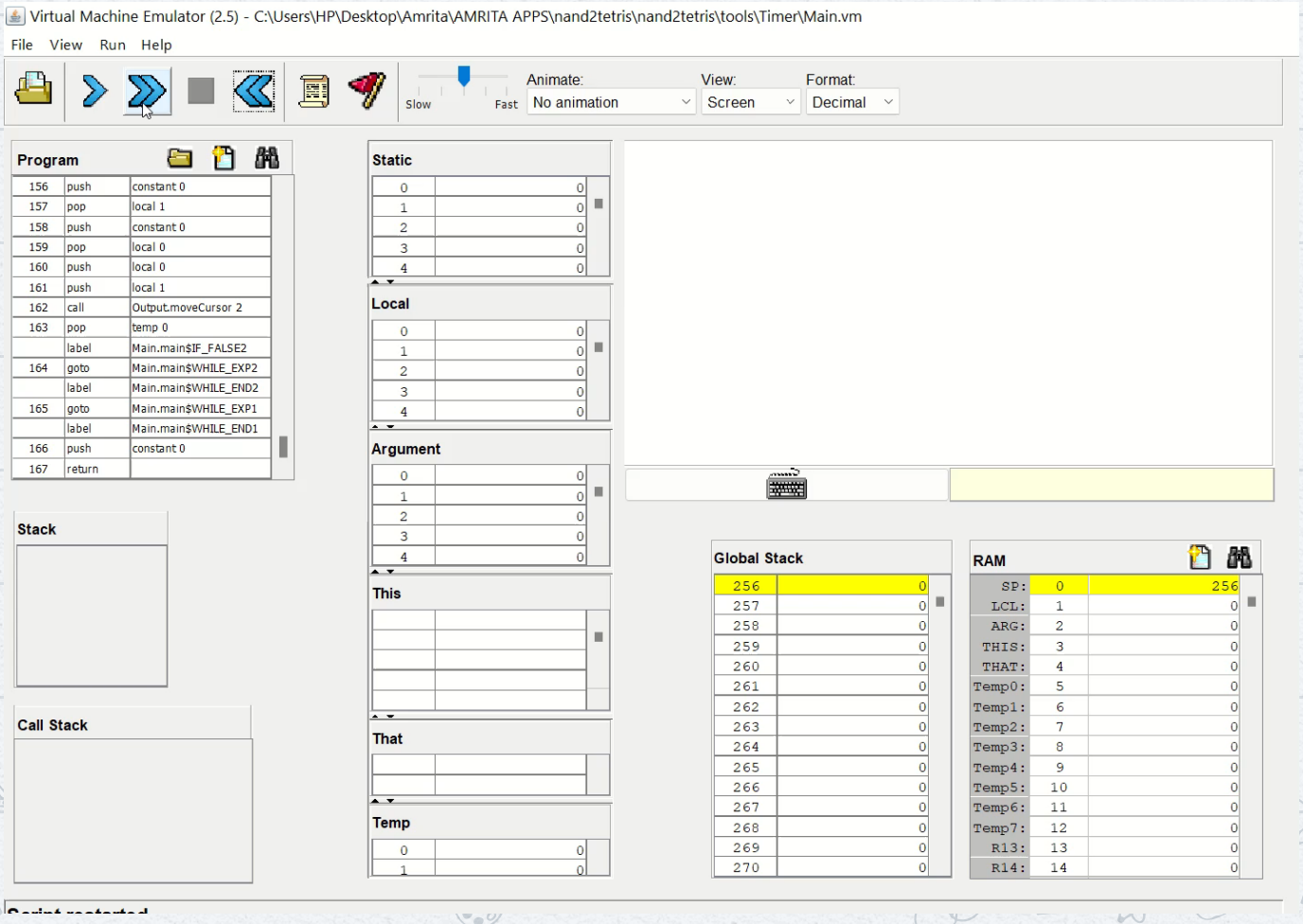
Timer > Main.jack

```
26          while (x>30) {
27              let x = 0;
28              do Output.printString("Counter reset: ");
29              do Output.println();
30              while (x<31) {
31              // do Output.moveCursor(a,b);
32              do Output.printInt(x);
33              do Output.println();
34              do Sys.wait(1000);
35              let x = x + 1;
36              if (x = 22) {
37                  do Screen.clearScreen();
38
39              }
40
41              if (x>30) {
42                  do Screen.clearScreen();
43                  let b = 0;
44                  let a = 0;
45                  do Output.moveCursor(a,b);
46
47              }
48
49              }
50      }
51      return;
52  }
53  }
```

JACK COMPILATION



OUTPUT


Virtual Machine Emulator (2.5) - C:\Users\HP\Desktop\Amrita\AMRITA APPS\nand2tetris\nand2tetris\tools\Timer\Main.vm
File View Run Help
Slow Fast
Animate: No animation
View: Screen
Format: Decimal
Program
156 push constant 0
157 pop local 1
158 push constant 0
159 pop local 0
160 push local 0
161 push local 1
162 call Output.moveCursor 2
163 pop temp 0
label Main.main$IF_FALSE2
164 goto Main.main$WHILE_EXP2
label Main.main$WHILE_END2
165 goto Main.main$WHILE_EXP1
label Main.main$WHILE_END1
166 push constant 0
167 return
Static
Local
Argument
This
That
Temp
Stack
Call Stack
Global Stack
RAM
SP: 0 256
LCL: 1 0
ARG: 2 0
THIS: 3 0
THAT: 4 0
Temp0: 5 0
Temp1: 6 0
Temp2: 7 0
Temp3: 8 0
Temp4: 9 0
Temp5: 10 0
Temp6: 11 0
Temp7: 12 0
R13: 13 0
R14: 14 0
Script restarted

ALTERNATIVE JACK CODE

Main - Notepad

File Edit View

```
class Main  {
function void main() {
  var int k;
  let k=1;


  while (true) {
        if (k=31){
         let k=0;
         do Screen.clearScreen();
}

do Output.printInt(k);
do Output.printString(" ");
do Sys.wait(1000);

if (k=0) {
 do Sys.halt();
}

let k=k+1;
}

return;

  }
}
```

Next we have to compile that in command prompt

Next we can find .vm file in folder as below

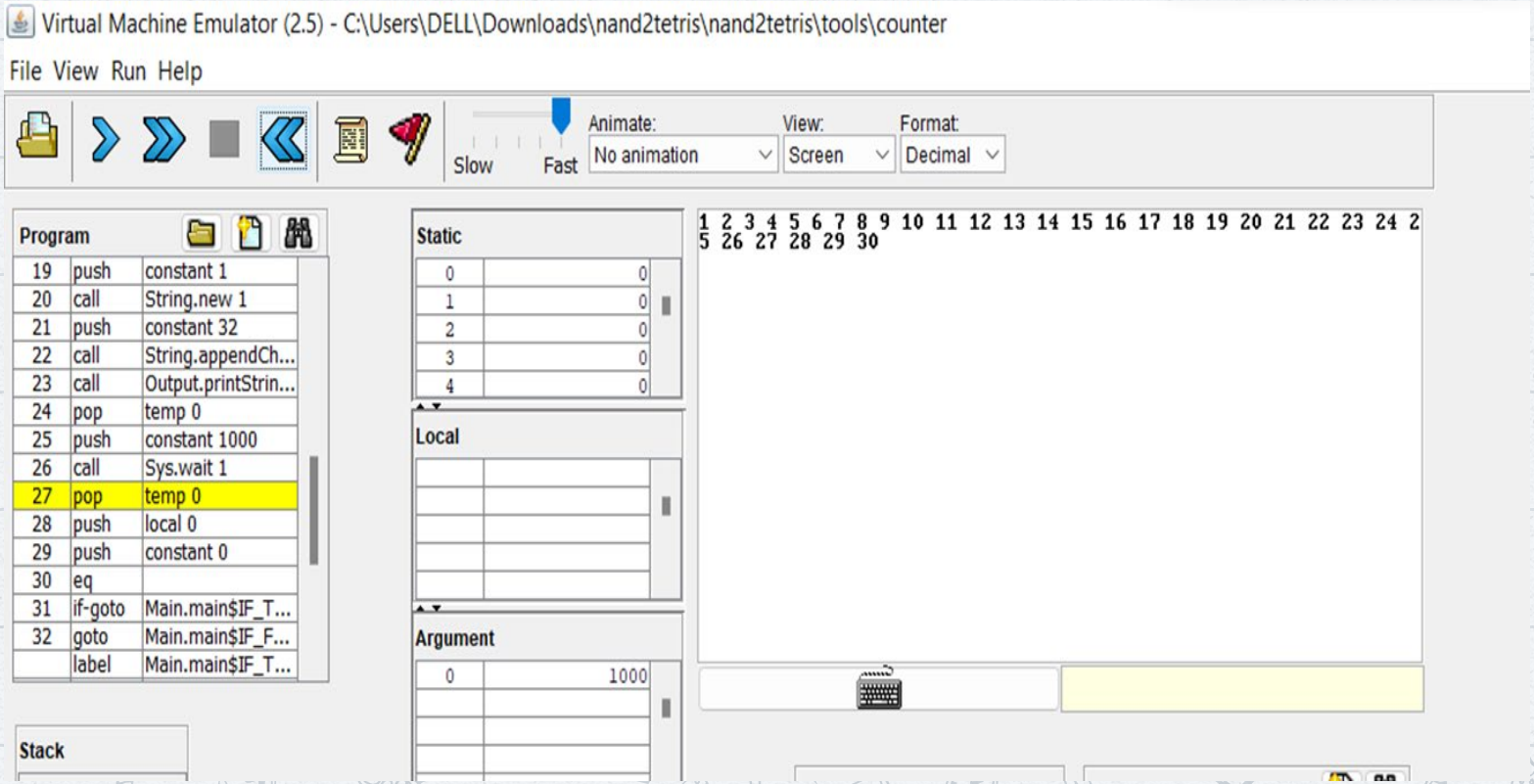
Main.vm - Notepad

File Edit View

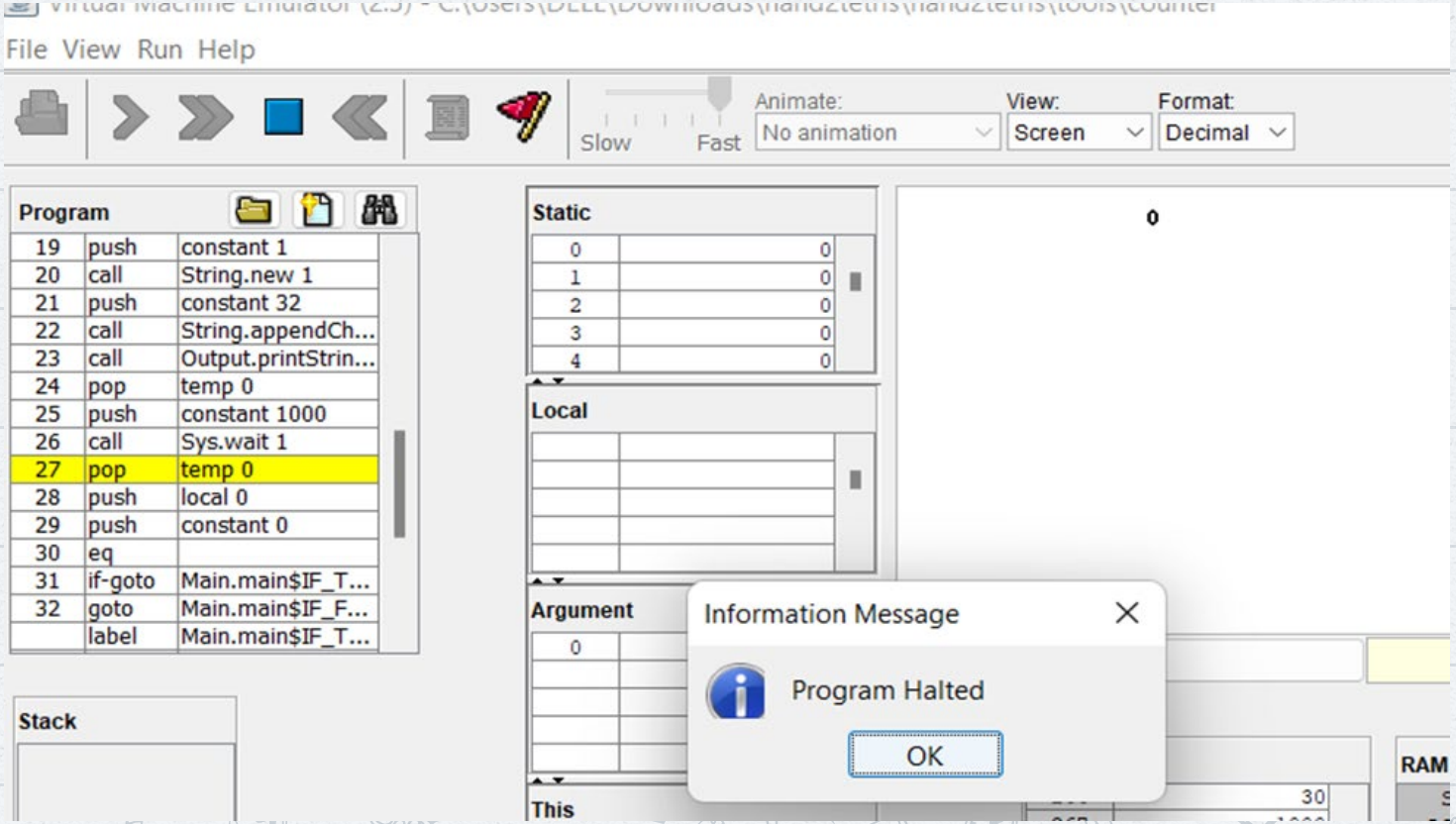
```
function Main.main 1
push constant 1
pop local 0
label WHILE_EXP0
push constant 0
not
not
if-goto WHILE_END0
push local 0
push constant 31
eq
if-goto IF_TRUE0
goto IF_FALSE0
label IF_TRUE0
push constant 0
pop local 0
call Screen.clearScreen 0
pop temp 0
label IF_FALSE0
push local 0
call Output.printInt 1
pop temp 0
```

```
push constant 1000
call Sys.wait 1
pop temp 0
push local 0
push constant 0
eq
if-goto IF_TRUE1
goto IF_FALSE1
label IF_TRUE1
call Sys.halt 0
pop temp 0
label IF_FALSE1
push local 0
push constant 1
add
pop local 0
goto WHILE_EXP0
label WHILE_END0
push constant 0
return
```

ALTERNATIVE OUTPUT



ALTERNATIVE OUTPUT



3.

Write a Jack program to find the sum of prime numbers from 50 to 100.

CODE LOGIC

- We will take input from user of two number that are ranging from n=50 to n=100.
- Then we will check whether number starting from n = 50 till n=100 contains how many prime numbers and then add those prime numbers.

JACK CODE

Prime > Main.jack

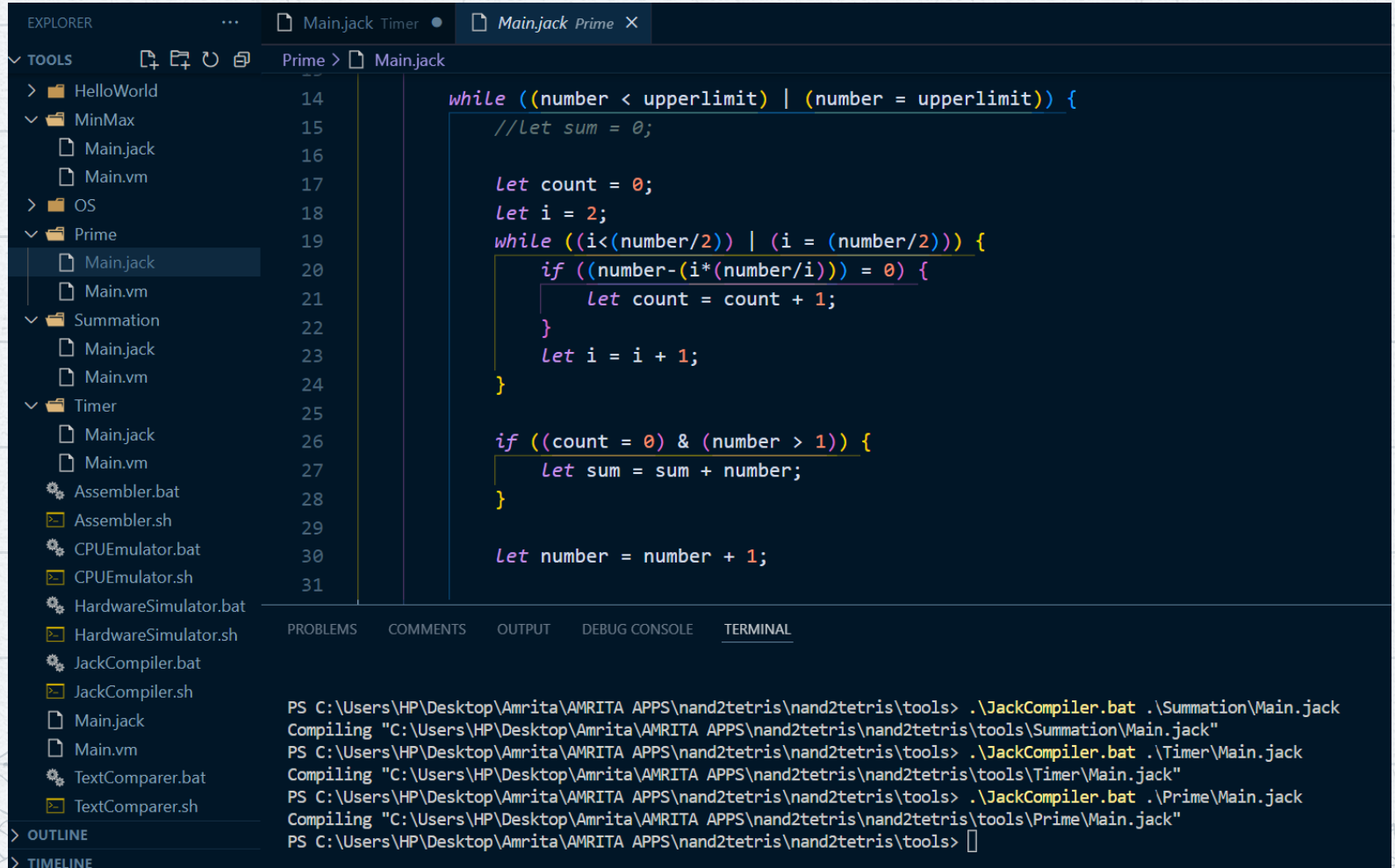
```
class Main {
    function void main () {
        var int count, sum, number, i;
        var boolean k;
        var int upperlimit;
        var int number; // Starting range of prime number to be checked and keep number less than or equal
        to final number
        let sum = 0;
        let count = 0;


        let number = Keyboard.readInt("Enter the first number: ");
        let upperlimit = Keyboard.readInt("Enter the second number: ");

        while ((number < upperlimit) | (number = upperlimit)) {
            //Let sum = 0;

            let count = 0;
            let i = 2;
            while ((i<(number/2)) | (i = (number/2))) {
                if ((number-(i*(number/i))) = 0) {
                    let count = count + 1;
                }
                let i = i + 1;
            }

            if ((count = 0) & (number > 1)) {
                let sum = sum + number;
```

JACK CODE

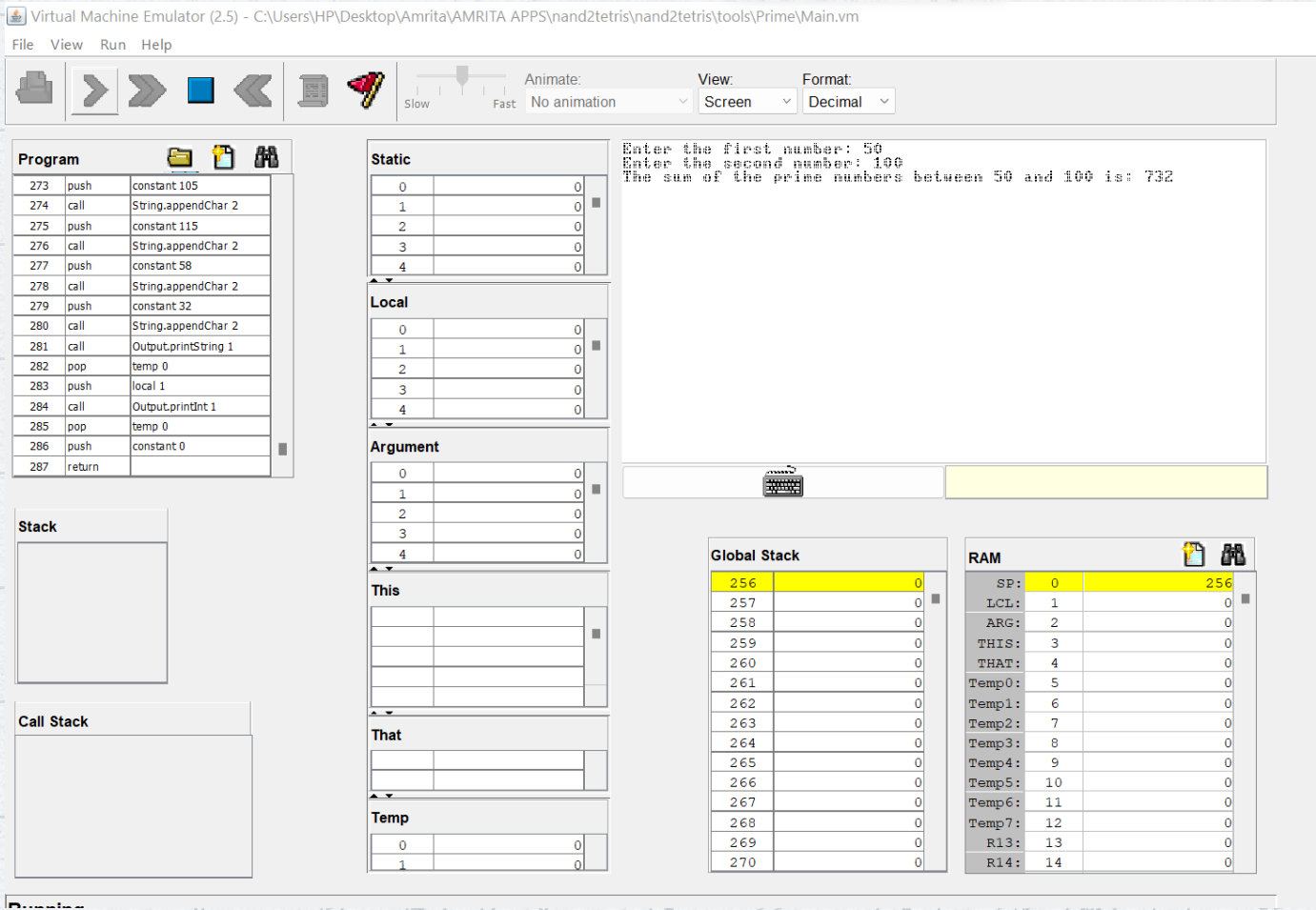
Prime > Main.jack

```
        while ((number < upperlimit) | (number = upperlimit)) {
            //let sum = 0;

            let count = 0;
            let i = 2;
            while ((i<(number/2)) | (i = (number/2))) {
                if ((number-(i*(number/i))) = 0) {
                    let count = count + 1;
                }
                let i = i + 1;
            }

            if ((count = 0) & (number > 1)) {
                let sum = sum + number;
            }

            let number = number + 1;

        }
        do Output.printString("The sum of the prime numbers between 50 and 100 is: ");
        do Output.printInt(sum);
        return;

    }
}
```

JACK COMPILATION



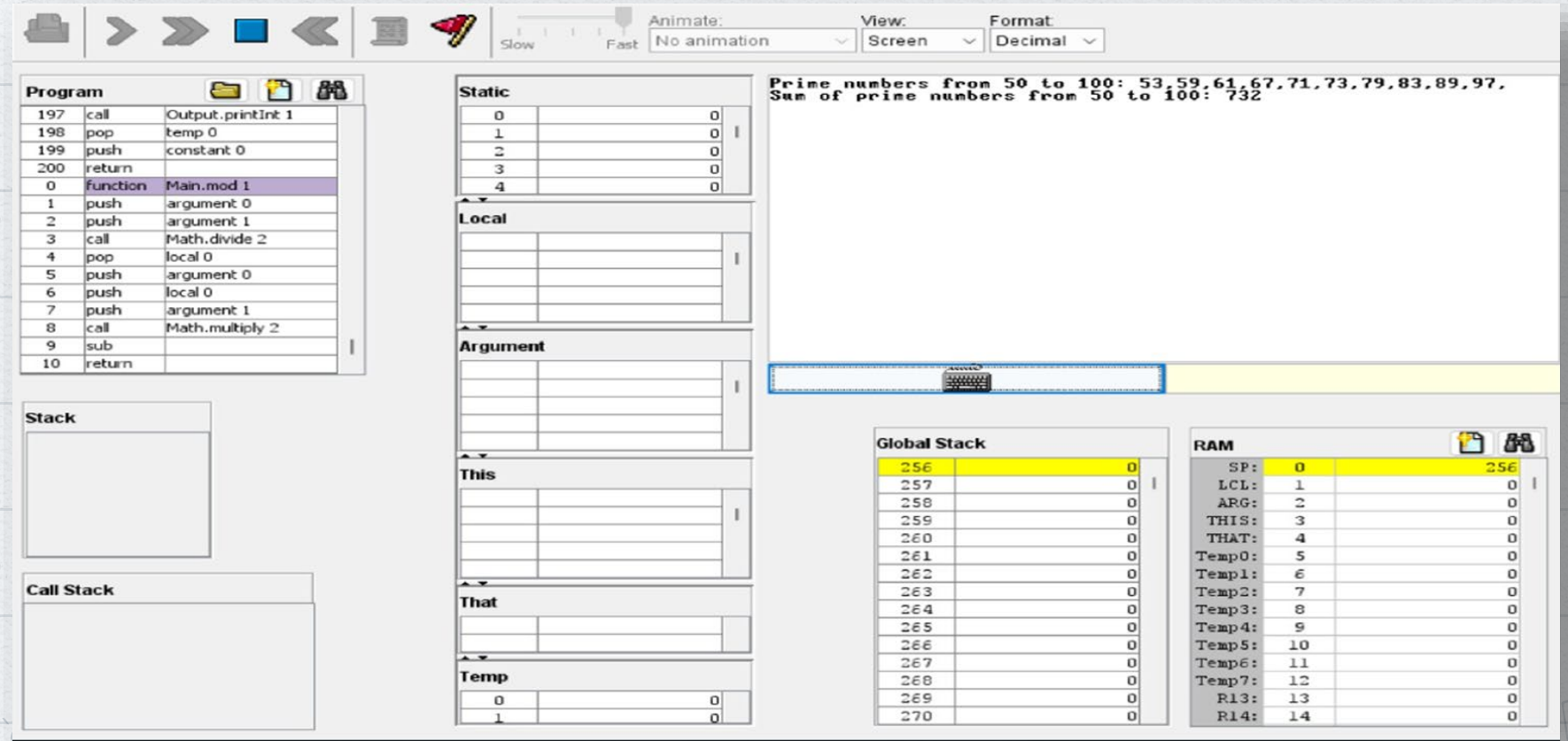
OUTPUT


Virtual Machine Emulator (2.5) - C:\Users\HP\Desktop\Amrita\AMRITA APPS\nand2tetris\nand2tetris\tools\Prime\Main.vm
File View Run Help
Animate: No animation
View: Screen
Format: Decimal
Slow Fast
Program
273 push constant 105
274 call String.appendChar 2
275 push constant 115
276 call String.appendChar 2
277 push constant 58
278 call String.appendChar 2
279 push constant 32
280 call String.appendChar 2
281 call Output.printString 1
282 pop temp 0
283 push local 1
284 call Output.printInt 1
285 pop temp 0
286 push constant 0
287 return
Static
Local
Argument
This
That
Temp
Enter the first number: 50
Enter the second number: 100
The sum of the prime numbers between 50 and 100 is: 732
Stack
Call Stack
Global Stack
256 0
257 0
258 0
259 0
260 0
261 0
262 0
263 0
264 0
265 0
266 0
267 0
268 0
269 0
270 0
RAM
SP: 0 256
LCL: 1 0
ARG: 2 0
THIS: 3 0
THAT: 4 0
Temp0: 5 0
Temp1: 6 0
Temp2: 7 0
Temp3: 8 0
Temp4: 9 0
Temp5: 10 0
Temp6: 11 0
Temp7: 12 0
R13: 13 0
R14: 14 0

ALTERNATIVE JACK CODE

```
class Main {
  function void main() {
    var int i;
    var int sum;   //We took 2 variables one for the loop and another to store the output.
    let i = 50;
do Output.printString("Prime numbers from 50 to 100: ");  //It prints the statement which represent the numbers we stored in the sum

    while (i < 100) {
      if (~((Main.rem(i, 2) = 0) | (Main.rem(i, 3) = 0) | (Main.rem(i, 5) = 0) | (Main.rem(i, 7) = 0))) {
          let sum = sum + i;

// We used the while loop to check the numbers which are prime between 50-100 and we used the condition that remainder should not be 0 when we divide it with 2,3,5,7

do Output.printInt(i);
do Output.printString(",");     //We printed all the prime numbers between 50-100

      }
      let i = i + 1;

    }

   do Output.println();
   do Output.printString("Sum of prime numbers from 50 to 100: ");
   do Output.printInt(sum);   //It prints sum
   return;
 }

  function int rem(int dividend, int divisor) {
    var int quotient;

    let quotient=dividend/divisor;
    return dividend - (quotient * divisor); // Since remainder operation is not available we used division formula to find the reminder.
  }
}
```

VM SIMULATION



4.

Write a brief description of the various Jack OS Services

INTRODUCTION

- The Jack language comes with a standard library that may also be viewed as an interface to an underlying operating system.

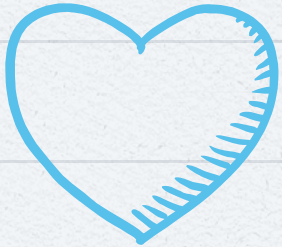
- The library is a collection of Jack classes, and must be provided in every implementation of the Jack language.

- The standard library includes the following classes:-

Math: Provides basic mathematical operations.
String: Implements the String type and basic.

- String-related operations:-

Array: Defines the Array type and allows construction and disposal of arrays.
Output: Handles text-based output.
Screen: Handles graphic screen output.
Keyboard: Handles user input from the keyboard.
Memory: Handles memory operations.
Sys: Provides some execution-related services.



THANKS!