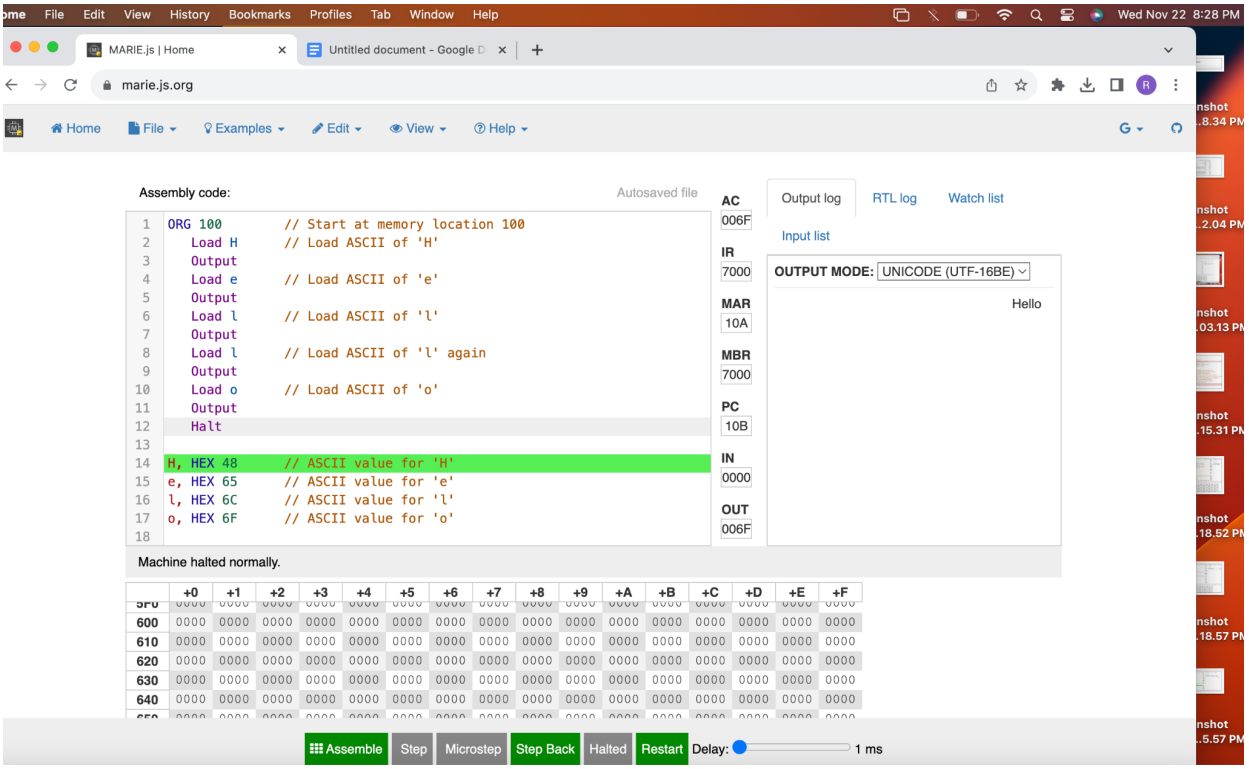
Q.no.1) Write the program to print the string "Hello" in MARIE assembly language.

```
ORG 100        // Start at memory location 100
   Load H      // Load ASCII of 'H'
   Output
   Load e      // Load ASCII of 'e'
   Output
   Load l      // Load ASCII of 'l'
   Output
   Load l      // Load ASCII of 'l' again
   Output
   Load o      // Load ASCII of 'o'
   Output
   Halt

H, HEX 48      // ASCII value for 'H'
e, HEX 65      // ASCII value for 'e'
l, HEX 6C      // ASCII value for 'l'
o, HEX 6F      // ASCII value for 'o'\\
```

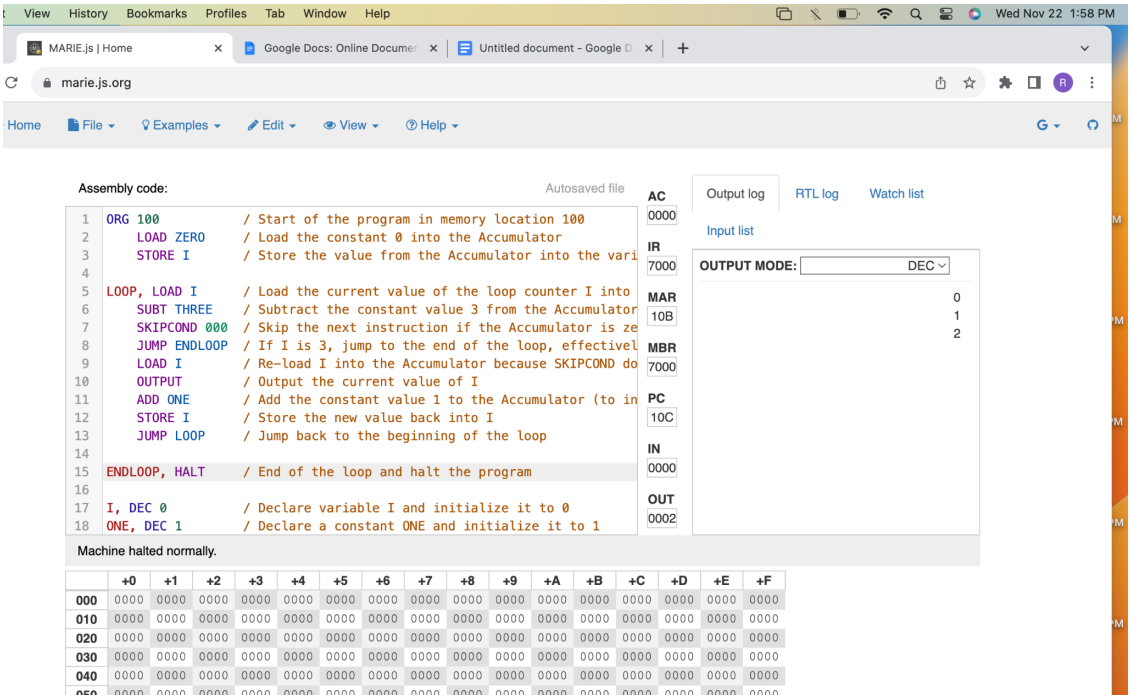
Output :



Q.no.2) Write the MARIE assembly program to implement "break" statement in for-loop shown as follows in Python program. for i in range(5): if i == 3: break print(i) 0 1 2

```
ORG 100          / Start of the program in memory location 100
   LOAD ZERO     / Load the constant 0 into the Accumulator
   STORE I       / Store the value from the Accumulator into the variable I (initializes loop counter to 0)

LOOP, LOAD I     / Load the current value of the loop counter I into the Accumulator
   SUBT THREE    / Subtract the constant value 3 from the Accumulator (to check for the break condition)
   SKIPCOND 000  / Skip the next instruction if the Accumulator is zero (i.e., I is 3)
   JUMP ENDLOOP  / If I is 3, jump to the end of the loop, effectively breaking it
   LOAD I        / Re-load I into the Accumulator because SKIPCOND does not alter the Accumulator
   OUTPUT        / Output the current value of I
   ADD ONE       / Add the constant value 1 to the Accumulator (to increment I)
   STORE I       / Store the new value back into I
   JUMP LOOP     / Jump back to the beginning of the loop

ENDLOOP, HALT    / End of the loop and halt the program

I, DEC 0         / Declare variable I and initialize it to 0
ONE, DEC 1       / Declare a constant ONE and initialize it to 1
THREE, DEC 3     / Declare a constant THREE and initialize it to 3 (used for our break condition)
ZERO, DEC 0      / Declare a constant ZERO and initialize it to 0 (used for initializing I)
```

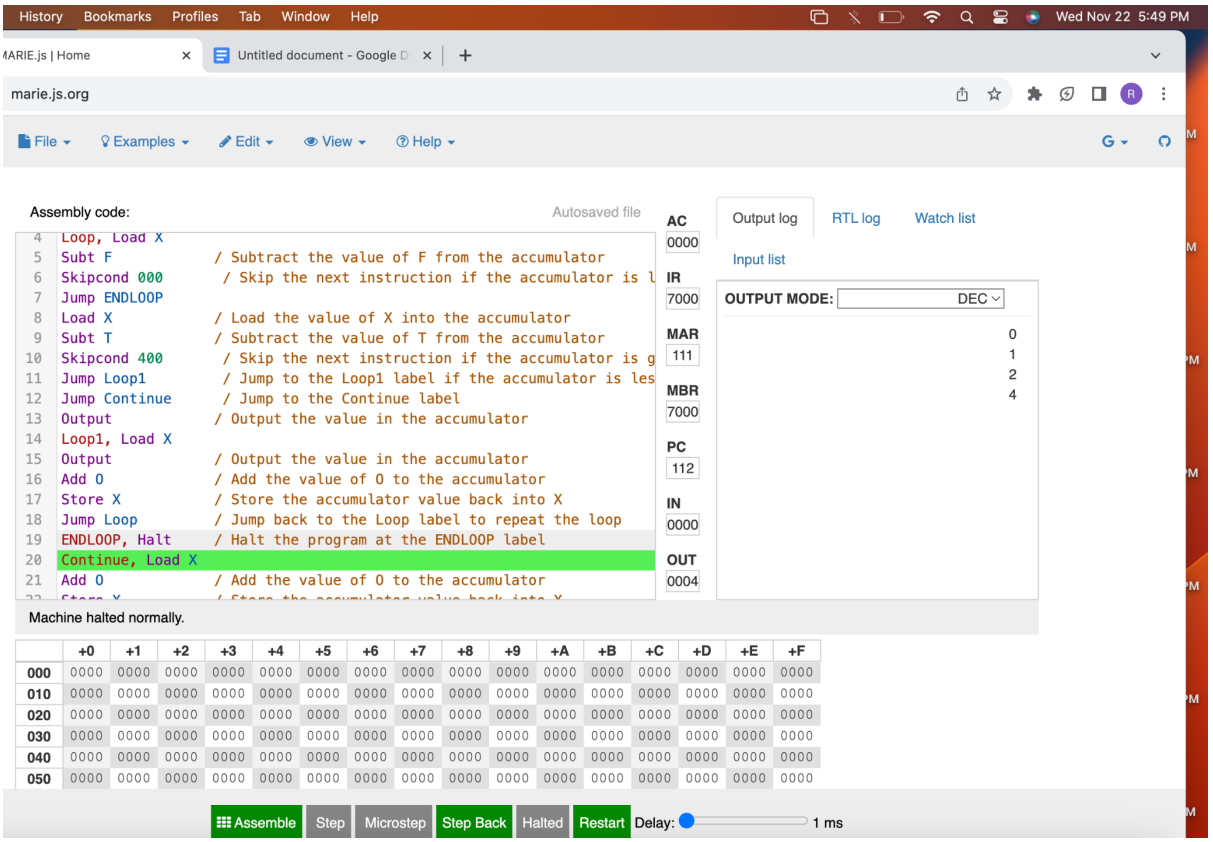
Output:



Q.no.3)

```
ORG 100
Load X          / Load value of x
Store X
Loop, Load X
Subt F          / Subtract the value of F from the accumulator
Skipcond 000    / Skip the next instruction if the accumulator is less than 0
Jump ENDLOOP
Load X          / Load the value of X into the accumulator
Subt T          / Subtract the value of T from the accumulator
Skipcond 400    / Skip the next instruction if the accumulator is greater than or equal to 0
Jump Loop1      / Jump to the Loop1 label if the accumulator is less than 0
Jump Continue   / Jump to the Continue label
Output          / Output the value in the accumulator
Loop1, Load X
Output          / Output the value in the accumulator
Add O           / Add the value of O to the accumulator
Store X         / Store the accumulator value back into X
Jump Loop       / Jump back to the Loop label to repeat the loop
ENDLOOP, Halt   / Halt the program at the ENDLOOP label
```

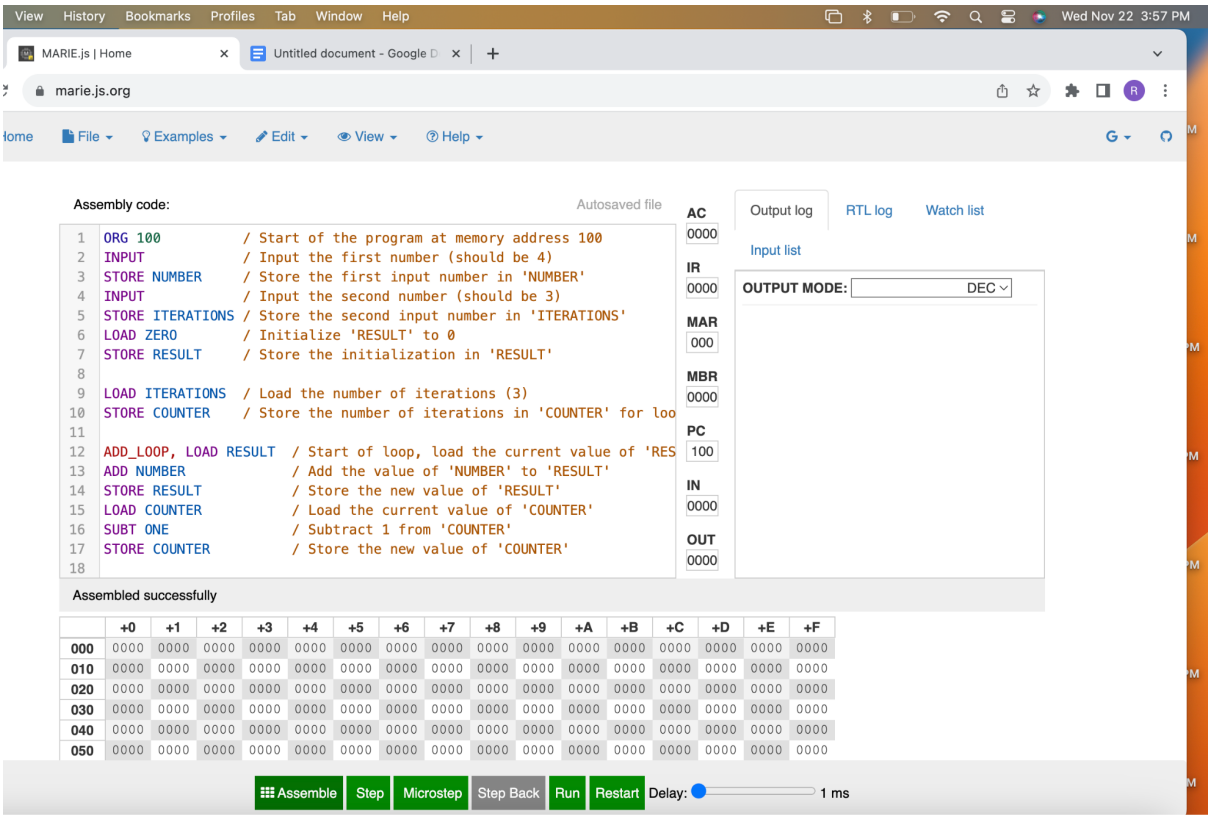
```
Continue, Load X
Add O           / Add the value of O to the accumulator
Store X         / Store the accumulator value back into X
Jump Loop
X, Dec 0        / Initialize X to 0
T, Dec 3
O, Dec 1
F, Dec 5
END 100         / End of the program
```

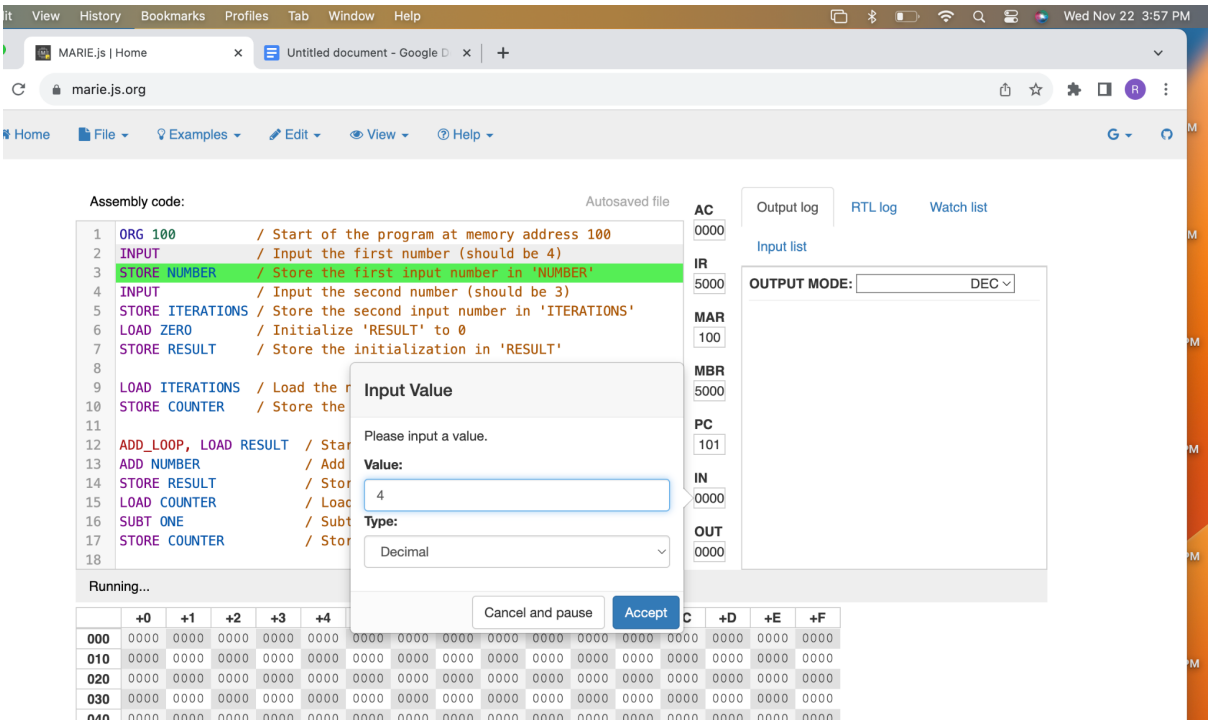


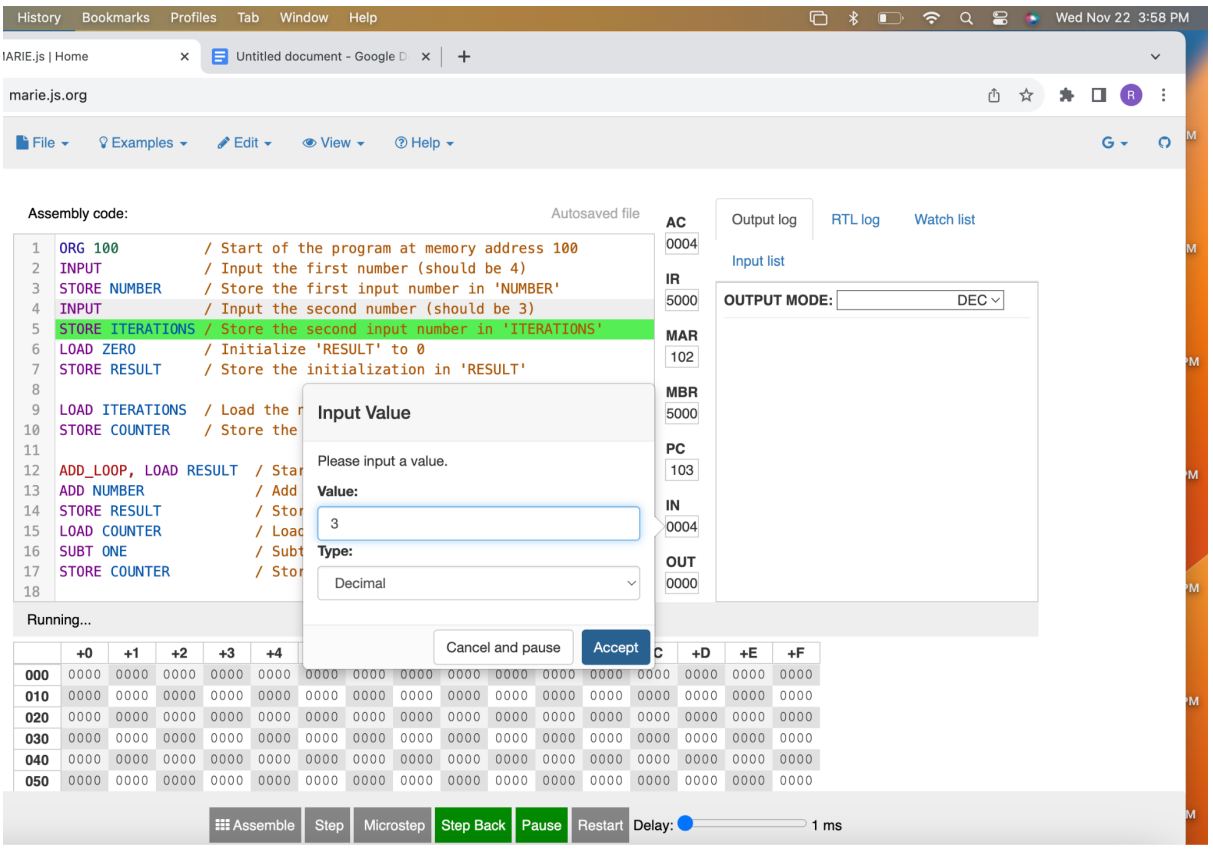
Q.No.4) Since there is not a multiplication instruction in ISA of MARIE, two integers multiplication operation, for instance, 4×3, must be done by the addition operation, like 4×3 = 4 + 4 + 4. Write the MARIE assembly program to find the product of two integers m×n.

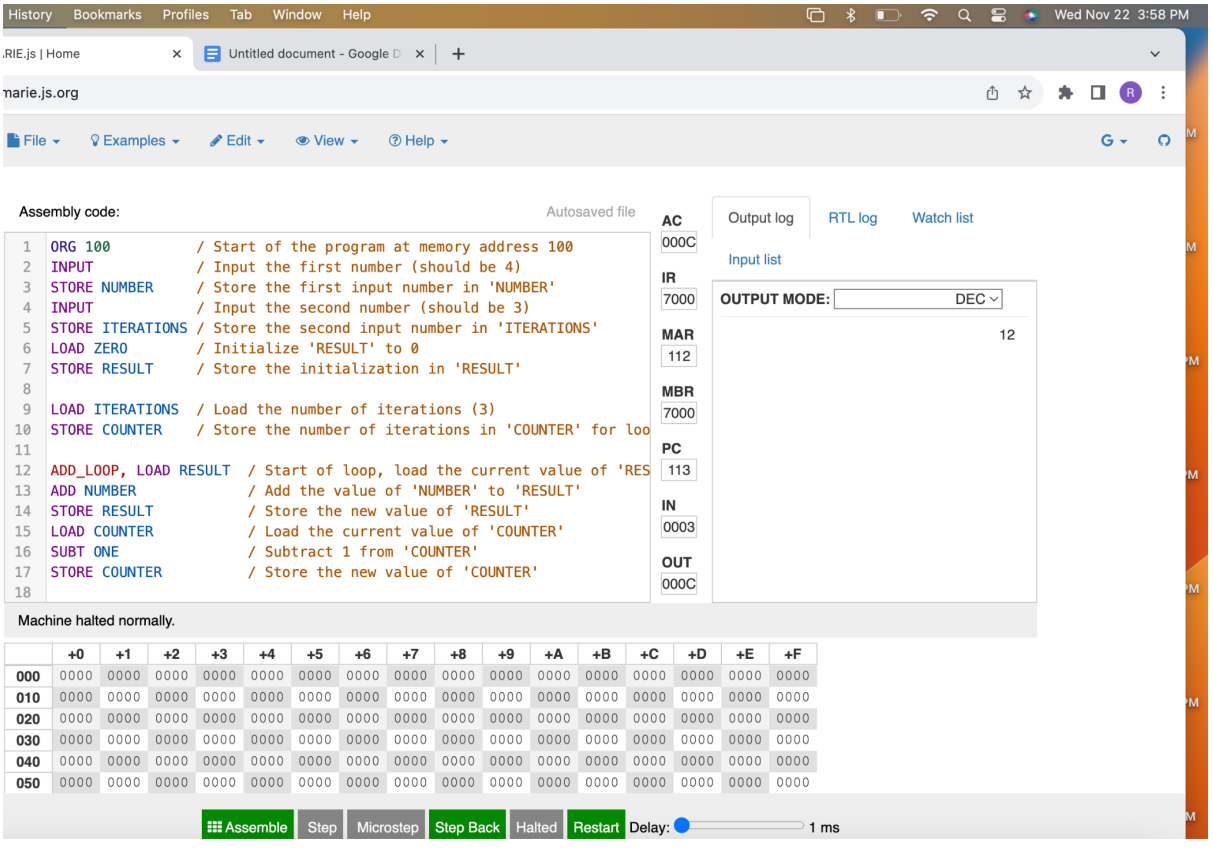
```
ORG 100         / Start of the program at memory address 100
INPUT           / Input the first number (should be 4)
STORE NUMBER    / Store the first input number in 'NUMBER'
INPUT           / Input the second number (should be 3)
STORE ITERATIONS / Store the second input number in 'ITERATIONS'
```

```
LOAD ZERO        / Initialize 'RESULT' to 0
STORE RESULT     / Store the initialization in 'RESULT'

LOAD ITERATIONS  / Load the number of iterations (3)
STORE COUNTER    / Store the number of iterations in 'COUNTER' for loop control

ADD_LOOP, LOAD RESULT  / Start of loop, load the current value of 'RESULT'
ADD NUMBER             / Add the value of 'NUMBER' to 'RESULT'
STORE RESULT           / Store the new value of 'RESULT'
LOAD COUNTER           / Load the current value of 'COUNTER'
SUBT ONE               / Subtract 1 from 'COUNTER'
STORE COUNTER          / Store the new value of 'COUNTER'

SKIPCOND 400           / Check if 'COUNTER' is zero
JUMP ADD_LOOP          / If not, repeat the loop

LOAD RESULT            / Load the final result
OUTPUT                 / Output the result
HALT                   / Stop the program

ZERO, DEC 0            / Constant value 0
ONE, DEC 1             / Constant value 1
NUMBER, DEC 0          / Will hold the first input number
ITERATIONS, DEC 0      / Will hold the second input number
RESULT, DEC 0          / Will hold the final result
COUNTER, DEC 0         / Used for loop control
```


View History Bookmarks Profiles Tab Window Help
Wed Nov 22 3:57 PM
MARIE.js | Home
Untitled document - Google D
marie.js.org
Home File Examples Edit View Help
Assembly code:
Autosaved file
1 ORG 100 / Start of the program at memory address 100
2 INPUT / Input the first number (should be 4)
3 STORE NUMBER / Store the first input number in 'NUMBER'
4 INPUT / Input the second number (should be 3)
5 STORE ITERATIONS / Store the second input number in 'ITERATIONS'
6 LOAD ZERO / Initialize 'RESULT' to 0
7 STORE RESULT / Store the initialization in 'RESULT'
8
9 LOAD ITERATIONS / Load the number of iterations (3)
10 STORE COUNTER / Store the number of iterations in 'COUNTER' for loo
11
12 ADD_LOOP, LOAD RESULT / Start of loop, load the current value of 'RES
13 ADD NUMBER / Add the value of 'NUMBER' to 'RESULT'
14 STORE RESULT / Store the new value of 'RESULT'
15 LOAD COUNTER / Load the current value of 'COUNTER'
16 SUBT ONE / Subtract 1 from 'COUNTER'
17 STORE COUNTER / Store the new value of 'COUNTER'
18
AC 0000
IR 0000
MAR 000
MBR 0000
PC 100
IN 0000
OUT 0000
Output log
RTL log
Watch list
Input list
OUTPUT MODE: DEC
Assembled successfully
+0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F
000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
020 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
030 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
040 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
050 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
Assemble
Step
Microstep
Step Back
Run
Restart
Delay:
1 ms


View History Bookmarks Profiles Tab Window Help
Wed Nov 22 3:57 PM
MARIE.js | Home
Untitled document - Google D
marie.js.org
Home File Examples Edit View Help
Assembly code:
Autosaved file
1 ORG 100 / Start of the program at memory address 100
2 INPUT / Input the first number (should be 4)
3 STORE NUMBER / Store the first input number in 'NUMBER'
4 INPUT / Input the second number (should be 3)
5 STORE ITERATIONS / Store the second input number in 'ITERATIONS'
6 LOAD ZERO / Initialize 'RESULT' to 0
7 STORE RESULT / Store the initialization in 'RESULT'
8
9 LOAD ITERATIONS / Load the n
10 STORE COUNTER / Store the
11
12 ADD_LOOP, LOAD RESULT / Star
13 ADD NUMBER / Add
14 STORE RESULT / Stor
15 LOAD COUNTER / Loac
16 SUBT ONE / Subt
17 STORE COUNTER / Stor
18
Input Value
Please input a value.
Value:
4
Type:
Decimal
Cancel and pause
Accept
AC 0000
IR 5000
MAR 100
MBR 5000
PC 101
IN 0000
OUT 0000
Output log
RTL log
Watch list
Input list
OUTPUT MODE: DEC
Running...
+0 +1 +2 +3 +4 +C +D +E +F
000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
020 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
030 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000


History Bookmarks Profiles Tab Window Help
Wed Nov 22 3:58 PM
marie.js.org
File Examples Edit View Help
Assembly code:
Autosaved file
1 ORG 100 / Start of the program at memory address 100
2 INPUT / Input the first number (should be 4)
3 STORE NUMBER / Store the first input number in 'NUMBER'
4 INPUT / Input the second number (should be 3)
5 STORE ITERATIONS / Store the second input number in 'ITERATIONS'
6 LOAD ZERO / Initialize 'RESULT' to 0
7 STORE RESULT / Store the initialization in 'RESULT'
8
9 LOAD ITERATIONS / Load the r
10 STORE COUNTER / Store the
11
12 ADD_LOOP, LOAD RESULT / Star
13 ADD NUMBER / Add
14 STORE RESULT / Stor
15 LOAD COUNTER / Loac
16 SUBT ONE / Subt
17 STORE COUNTER / Stor
18
Input Value
Please input a value.
Value:
3
Type:
Decimal
Cancel and pause
Accept
AC 0004
IR 5000
MAR 102
MBR 5000
PC 103
IN 0004
OUT 0000
Output log
RTL log
Watch list
Input list
OUTPUT MODE: DEC
Running...
+0 +1 +2 +3 +4 +D +E +F
000 0000 0000 0000 0000 0000 0000 0000 0000
010 0000 0000 0000 0000 0000 0000 0000 0000
020 0000 0000 0000 0000 0000 0000 0000 0000
030 0000 0000 0000 0000 0000 0000 0000 0000
040 0000 0000 0000 0000 0000 0000 0000 0000
050 0000 0000 0000 0000 0000 0000 0000 0000
Assemble Step Microstep Step Back Pause Restart Delay: 1 ms


History Bookmarks Profiles Tab Window Help
Wed Nov 22 3:58 PM
marie.js.org
File Examples Edit View Help
Assembly code:
Autosaved file
1 ORG 100 / Start of the program at memory address 100
2 INPUT / Input the first number (should be 4)
3 STORE NUMBER / Store the first input number in 'NUMBER'
4 INPUT / Input the second number (should be 3)
5 STORE ITERATIONS / Store the second input number in 'ITERATIONS'
6 LOAD ZERO / Initialize 'RESULT' to 0
7 STORE RESULT / Store the initialization in 'RESULT'
8
9 LOAD ITERATIONS / Load the number of iterations (3)
10 STORE COUNTER / Store the number of iterations in 'COUNTER' for loo
11
12 ADD_LOOP, LOAD RESULT / Start of loop, load the current value of 'RES
13 ADD NUMBER / Add the value of 'NUMBER' to 'RESULT'
14 STORE RESULT / Store the new value of 'RESULT'
15 LOAD COUNTER / Load the current value of 'COUNTER'
16 SUBT ONE / Subtract 1 from 'COUNTER'
17 STORE COUNTER / Store the new value of 'COUNTER'
18
AC 000C
IR 7000
MAR 112
MBR 7000
PC 113
IN 0003
OUT 000C
Output log
RTL log
Watch list
Input list
OUTPUT MODE: DEC
12
Machine halted normally.
+0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F
000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
020 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
030 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
040 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
050 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
Assemble Step Microstep Step Back Halted Restart Delay: 1 ms