

1.

SOURCE CODE:

Assembly code: Autosaved file

```
1  ORG 100
2  Load HChar
3  Output
4  Load EChar
5  Output
6  Load LChar
7  Output
8  Load LChar
9  Output
10 Load OChar
11 Output
12 Halt
13
14 HChar, HEX 48 / ASCII for 'H'
15 EChar, HEX 65 / ASCII for 'e'
16 LChar, HEX 6C / ASCII for 'l'
17 OChar, HEX 6F / ASCII for 'o'
18 END
```

Machine halted normally.

| | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +A | +B | +C | +D |
|-----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 000 | 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 |

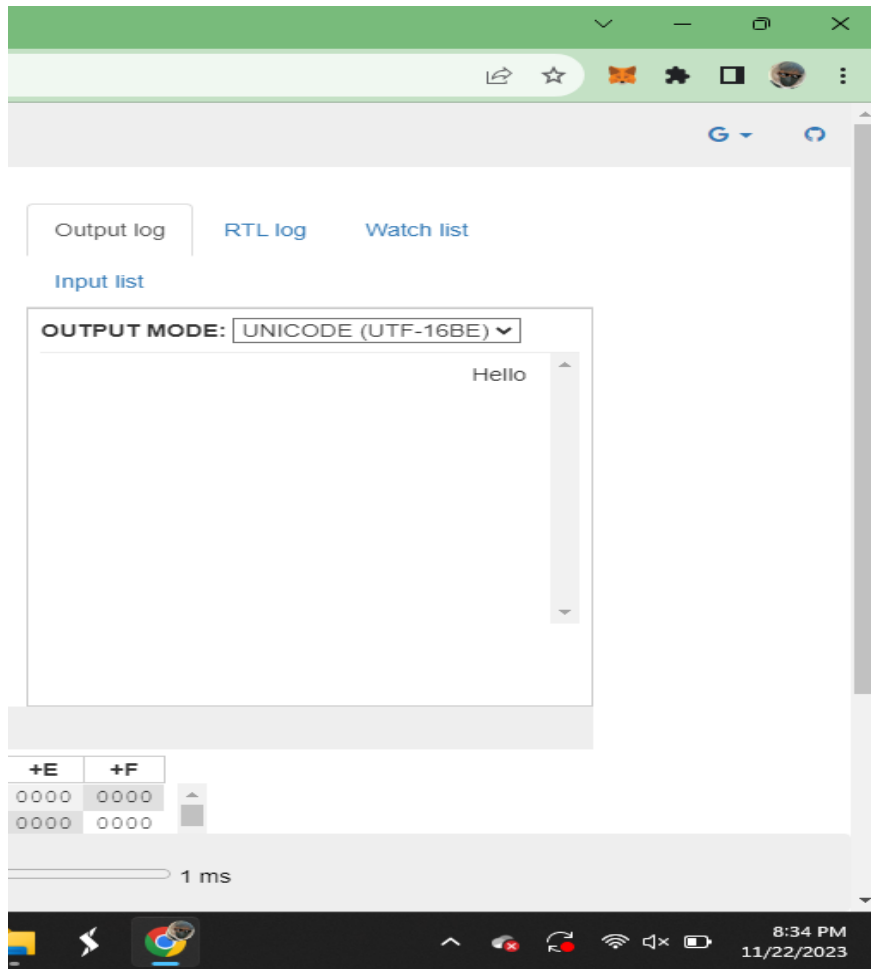
AC 006F
IR 7000
MAR 10A
MBR 7000
PC 10B
IN 0000
OUT 006F

Assemble Step Microstep Step Back Halted Restart Delay:

62°F Sunny

Search

OUTPUT:



2.

SOURCE CODE:

ORG 100

LOAD ZERO

STORE I

LOOP, LOAD I

SUBT THREE

SKIPCOND 000

JUMP PRINT

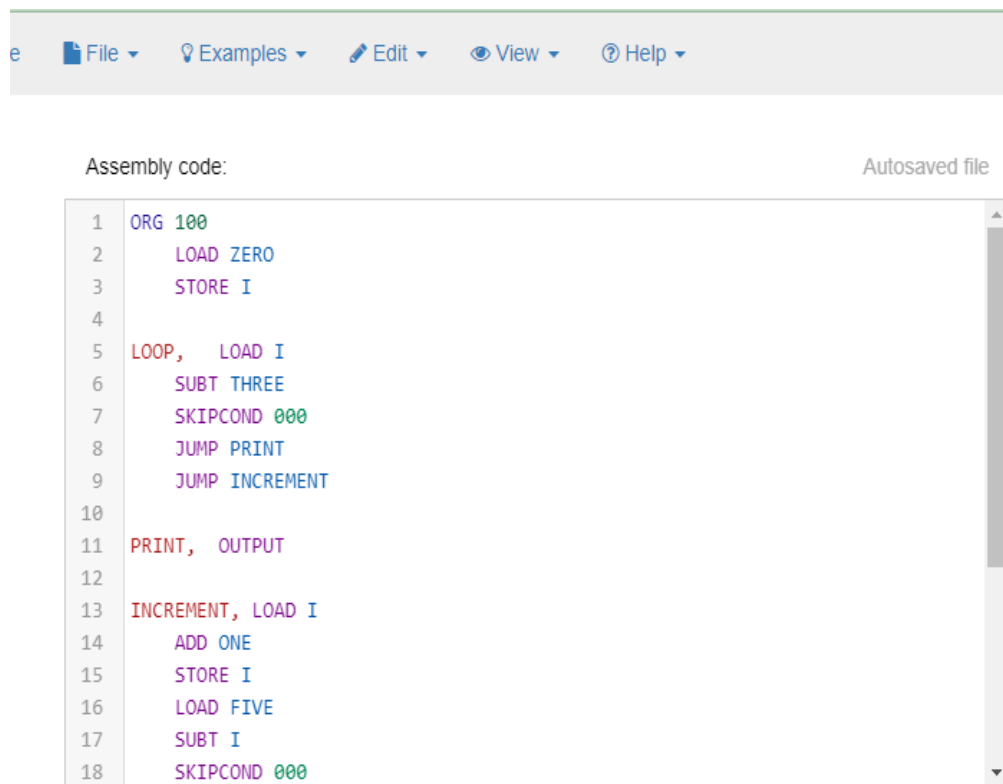
JUMP INCREMENT

PRINT, OUTPUT

```
INCREMENT, LOAD I
  ADD ONE
  STORE I
  LOAD FIVE
  SUBT I
  SKIPCOND 000
  JUMP LOOP
```

```
ENDLOOP, HALT
```

```
I,    DEC 0
ONE,   DEC 1
THREE, DEC 3
FIVE,  DEC 5
ZERO,  DEC 0
```



The screenshot shows a software interface for editing assembly code. At the top is a menu bar with the following items: 'File', 'Examples', 'Edit', 'View', and 'Help'. Below the menu bar, the text 'Assembly code:' is on the left and 'Autosaved file' is on the right. The main area contains a list of assembly instructions, each preceded by a line number from 1 to 18. The instructions are: 1. ORG 100, 2. LOAD ZERO, 3. STORE I, 4. (blank), 5. LOOP, LOAD I, 6. SUBT THREE, 7. SKIPCOND 000, 8. JUMP PRINT, 9. JUMP INCREMENT, 10. (blank), 11. PRINT, OUTPUT, 12. (blank), 13. INCREMENT, LOAD I, 14. ADD ONE, 15. STORE I, 16. LOAD FIVE, 17. SUBT I, and 18. SKIPCOND 000. A vertical scrollbar is located on the right side of the code area.

```
1  ORG 100
2  LOAD ZERO
3  STORE I
4
5  LOOP,  LOAD I
6  SUBT THREE
7  SKIPCOND 000
8  JUMP PRINT
9  JUMP INCREMENT
10
11 PRINT,  OUTPUT
12
13 INCREMENT, LOAD I
14 ADD ONE
15 STORE I
16 LOAD FIVE
17 SUBT I
18 SKIPCOND 000
```

```
19      JUMP LOOP
20
21  ENDLOOP, HALT
22
23  I,      DEC 0
24  ONE,    DEC 1
25  THREE,  DEC 3
26  FIVE,   DEC 5
27  ZERO,   DEC 0
28
```

OUTPUT:

OUTPUT MODE:

DEC ▾

0

1

2

3.

SOURCE CODE:

ORG 100 ; Set the origin address to 100

LOAD ZeroValue ; Load the value from address ZeroValue into the accumulator

STORE CounterVariable ; Store the value in the accumulator into the memory location CounterVariable

StartLoop, ; Label for the start of the loop

LOAD CounterVariable ; Load the value from address CounterVariable into the accumulator

SUBT ThreeValue ; Subtract the value at address ThreeValue

SKIPCOND 400 ; Skip the next instruction if the accumulator is non-positive (i.e., if result is negative or zero)

JUMP IncrementCounter ; Jump to IncrementCounter if the condition is not met

SkipPrint, ; Label for skipping the print statement

LOAD CounterVariable ; Load the value from address CounterVariable into the accumulator

OUTPUT ; Output the value in the accumulator

IncrementCounter, ; Label for the increment section

LOAD CounterVariable ; Load the value from address CounterVariable into the accumulator

ADD OneValue ; Add the value at address OneValue to the accumulator

STORE CounterVariable ; Store the updated value in the accumulator back to the memory location CounterVariable

LOAD FiveValue ; Load the value from address FiveValue into the accumulator

SUBT CounterVariable ; Subtract the value at address CounterVariable from the accumulator

SKIPCOND 400 ; Skip the next instruction if the accumulator is non-positive (i.e., if result is negative or zero)

JUMP StartLoop ; Jump back to the StartLoop label if the condition is not met

EndLoop, ; Label for the end of the loop

HALT ; Halt the program

CounterVariable, HEX 0 ; Variable to store the loop index

OneValue, DEC 1 ; Constant for the value 1

ThreeValue, DEC 3 ; Constant for the value 3
FiveValue, DEC 5 ; Constant for the value 5
ZeroValue, HEX 0 ; Constant for the value 0

```
ORG 100 ; Set the origin address to 100

LOAD ZeroValue ; Load the value from address ZeroValue into the accumulator
STORE CounterVariable ; Store the value in the accumulator into the memory

StartLoop, ; Label for the start of the loop
LOAD CounterVariable ; Load the value from address CounterVariable into the accumulator
SUBT ThreeValue ; Subtract the value at address ThreeValue
SKIPCOND 400 ; Skip the next instruction if the accumulator is non-positive
JUMP IncrementCounter ; Jump to IncrementCounter if the condition is not met

SkipPrint, ; Label for skipping the print statement
LOAD CounterVariable ; Load the value from address CounterVariable into the accumulator
OUTPUT ; Output the value in the accumulator

IncrementCounter, ; Label for the increment section
LOAD CounterVariable ; Load the value from address CounterVariable into the accumulator
```

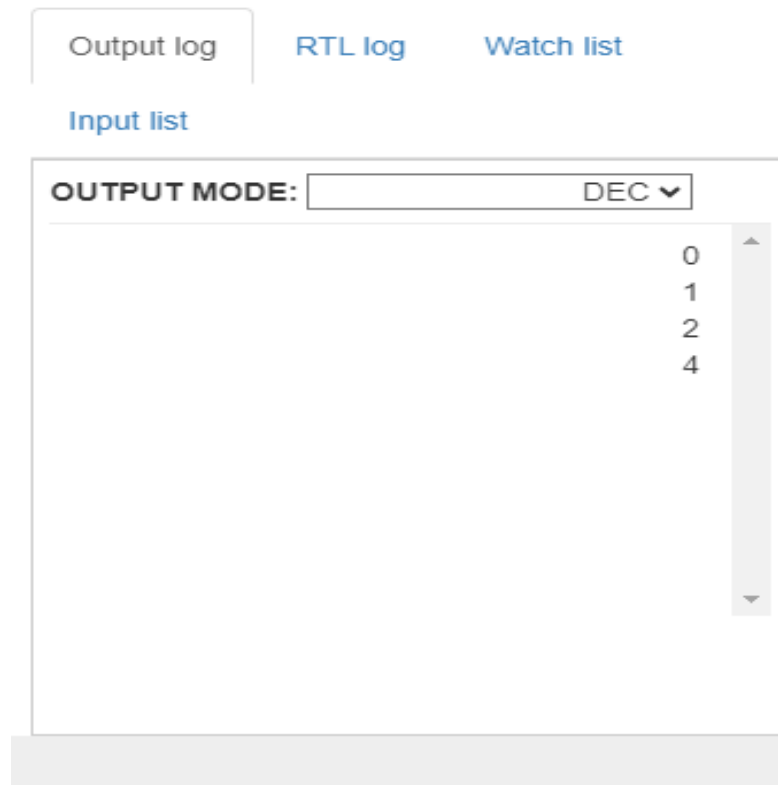
```
ADD OneValue ; Add the value at address OneValue to the accumulator
STORE CounterVariable ; Store the updated value in the accumulator back to memory

LOAD FiveValue ; Load the value from address FiveValue into the accumulator
SUBT CounterVariable ; Subtract the value at address CounterVariable from the accumulator
SKIPCOND 400 ; Skip the next instruction if the accumulator is non-positive
JUMP StartLoop ; Jump back to the StartLoop label if the condition is not met

EndLoop, ; Label for the end of the loop
HALT ; Halt the program

CounterVariable, HEX 0 ; Variable to store the loop index
OneValue, DEC 1 ; Constant for the value 1
ThreeValue, DEC 3 ; Constant for the value 3
FiveValue, DEC 5 ; Constant for the value 5
ZeroValue, HEX 0 ; Constant for the value 0
```

OUTPUT:



4.

SOURCE CODE:

ORG 100

LOAD 0 // Load 0 (ZERO)

STORE RESULT

LOAD ITERATIONS

STORE 100 // Use memory location 100 to store the counter

ADD_LOOP, LOAD RESULT

ADD NUMBER

STORE RESULT

LOAD 100 // Load the current counter from memory location 100

SUBT ONE

STORE 100 // Store the new counter value

SKIPCOND 400

JUMP ADD_LOOP

LOAD RESULT

OUTPUT

HALT

NUMBER, DEC 4

ITERATIONS, DEC 3

RESULT, DEC 0

ONE, DEC 1

Assembly code:

Autosaved file

```
1  ORG 100
2  LOAD 0      // Load 0 (ZERO)
3  STORE RESULT
4
5  LOAD ITERATIONS
6  STORE 100   // Use memory location 100 to store the counter
7
8  ADD_LOOP, LOAD RESULT
9  ADD NUMBER
10 STORE RESULT
11 LOAD 100    // Load the current counter from memory location 100
12 SUBT ONE
13 STORE 100   // Store the new counter value
14
15 SKIPCOND 400
16 JUMP ADD_LOOP
17
18 LOAD RESULT
```



```

7
8  LOAD RESULT
9  OUTPUT
0  HALT
1
2  NUMBER, DEC 4
3  ITERATIONS, DEC 3
4  RESULT, DEC 0
5  ONE, DEC 1
6

```

Machine halted normally.

| | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +A | +B | +C |
|----|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 00 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |
| 10 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 | 0000 |

Assemble Step Microstep Step Back Halted Restart

OUTPUT:

AC 000C Output log RTL log Watch list
 Input list

IR 7000 OUTPUT MODE: DEC
 MAR 10E 12
 MBR 7000
 PC 10F
 IN 0000
 OUT 000C

| | +D | +E | +F |
|---|------|------|------|
| 0 | 0000 | 0000 | 0000 |
| 1 | 0000 | 0000 | 0000 |

