

Little Man Computer

A computer simulation for the World Book TutorVision

Written by Michael Hayes

<http://www.midnightblueinternational.com>

Date of last modification: August 25, 2021

Addendum 2: Beginner Exercises

Introduction

Once you have completed the exercises in the Instruction Manual, you are ready to try writing Little Man Computer programs on your own. This first set of additional exercises is intended to be easier than the ones that follow.

Because I released this program as a free educational product, I look forward to seeing your solutions. See if you can implement each solution within the “par” number of lines. I welcome any requests for assistance.



Beginner Exercise 1: Negation

Description

Read one value from the Input Stream and store it to memory (one of the Mailboxes).

Output the negative equivalent of this value. If it's positive, then output a negative number. If it's negative, then output a positive number.

Halt (take a Coffee Break) after output.

Par

My solution uses a total of 8 code lines:

- 6 lines of instructions
- 1 DAT line containing a variable
- 1 DAT line containing a constant

Hints

You will need to store the value to memory, even though there is only one input.

That's because you will have to perform a mathematical operation on the Accumulator.

The limited instruction set of Little Man Computer makes simple tasks a little trickier than you may think.

Look back to the Subtraction example and see if you can't tweak it somehow to work here. Hopefully you still have the program saved on your copy of Little Man Computer.

What mathematical operation would you need to perform on the Accumulator?

Scratchpad

Use a pencil & eraser to work out your solution in the space below before opening the Code Editor.

Line Number	Label	Instruction	Reference
00			
01			
02			
03			
04			
05			
06			
07			
08			
09			

Beginner Exercise 2: Double If Equal

Description

Read two values from the Input Stream and store them to memory (the Mailboxes).

If the two input values are the same, then output double the value.

If the two input values are different, then output both values instead.

Halt (take a Coffee Break) after output.

Par

My solution uses a total of 16 code lines:

- 14 lines of instructions
- 2 DAT lines containing variables

Hints

Which Branch instruction would you use to check for equality? Look back to the Countdown example.

Be sure to set Overflow behavior to “Wrap Around” in the Options Menu.

Don't worry too much about Par here. Get it to work first, and then see if you can make Par.

Line Number	Label	Instruction	Reference
00			
01			
02			
03			
04			
05			
06			
07			
08			
09			
10			
11			
12			
13			
14			
15			

Beginner Exercise 3: Difference (Absolute Value after Subtraction)

Description

Read two values from the Input Stream and store them to memory.

Subtract one value from the other to calculate the difference.

If a negative value results, change it to a positive value.

Output this value, and then Halt.

Par

My solution uses a total of 15 code lines:

- 11 lines of instructions
- 3 DAT lines containing variables
- 1 DAT line containing a constant

You can reduce this to 13 lines with a little “trickery” but try for 15 lines first.

Hints

Try combining what you know from the Subtraction and Negation examples.

Line Number	Label	Instruction	Reference
00			
01			
02			
03			
04			
05			
06			
07			
08			
09			
10			
11			
12			
13			
14			
15			

Beginner Exercise 4: Multiplication

Description

Read two values from the Input Stream and store them to memory.

Multiply these Factors together, output the Product, and then Halt.

Par

My solution uses a total of 20 code lines:

- 16 lines of instructions
- 3 DAT lines containing variables
- 1 DAT line containing a constant

Hints

Look back to the Square example and see how you could modify it for this exercise.

Line Number	Label	Instruction	Reference
00			
01			
02			
03			
04			
05			
06			
07			
08			
09			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			

Beginner Exercise 5: PIN Entry

Description

Choose any number you want in the range 0-99.

Read a value from the Input Stream.

If that value matches your number, output a 1.

If that value doesn't match your number, output a 0.

Halt after output.

Par

My solution uses a total of 13 code lines:

- 9 lines of instructions
- 1 DAT line containing a variable
- 3 DAT lines containing constants

Hints

Think about how the program should flow. There is one result if a right answer is given, and a separate result if a wrong answer is given.

Extra Credit

Allow for up to three attempts. Display the number of attempts if one of them is correct.

Scratchpad

Line Number	Label	Instruction	Reference
00			
01			
02			
03			
04			
05			
06			
07			
08			
09			
10			
11			
12			

Beginner Exercise 6: Maximum of Three

Description

Read three values from the Input Stream and store them in memory.

Figure out which value is highest.

Output only the highest value and then Halt.

Par

My solution uses a total of 26 code lines:

- 23 lines of instructions
- 3 DAT lines containing variables

Hints

Don't worry too much about Par here. Get it to work first, and then see if you can make Par.

As with the previous example, think about how the program should flow. Make sure you compare all the numbers and don't miss any.

Extra Credit

Stretch the program to compare four values instead of three. It will take a bigger rewrite than you might expect at first.

Scratchpad

Line Number	Label	Instruction	Reference
00			
01			
02			
03			
04			
05			
06			
07			
08			
09			
10			
11			
12			

Line Number	Label	Instruction	Reference
13			
14			
15			
16			
17			
18			
19			
20			
21			
22			
23			
24			
25			
26			
27			
28			
29			
30			
31			
32			
33			

Conclusion

How did you do? Were these examples too easy or too difficult for you? Did you make Par on all the examples? Did you complete the Extra Credit on the last two examples? Go to my website and send me an e-mail to let me know. I'm happy to know my implementation of Little Man Computer was of some instructional value to somebody. This has been a departure from writing video games, which I usually do.

As with any other skill, the key to proficiency at programming is through practice and lots of it. I have more programming exercises which I plan to release soon, and which will be more difficult than these. Good luck.