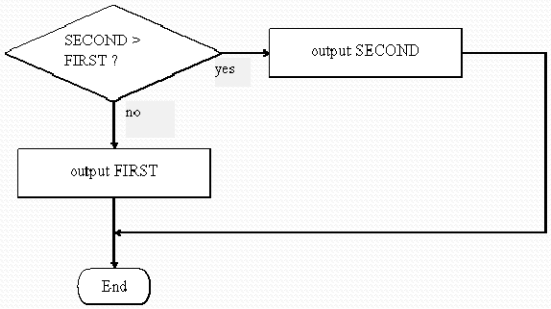
A452

LMC:larger of two numbers

flowchart for a program that will output the larger of two input values.



The Algorithm

- Input first number
- Store this as FIRST
- Input second number
- Store this as SECOND
- Subtract FIRST from SECOND
- If the result is positive SECOND is bigger so go to subsection *secondbigger*
- If not then First is bigger so load to accumulator and output
- Stop
- Subsection *secondbigger:*
- Load SECOND to accumulator and output
- Stop

LMC Code

```
INP
STA FIRST
INP
STA SECOND
SUB FIRST
BRP SECONDBIG
LDA FIRST
OUT
BRA PROGRAMEND
SECONDBIG LDA
SECOND
OUT
PROGRAMEND HLT
FIRST DAT
SECOND DAT
```

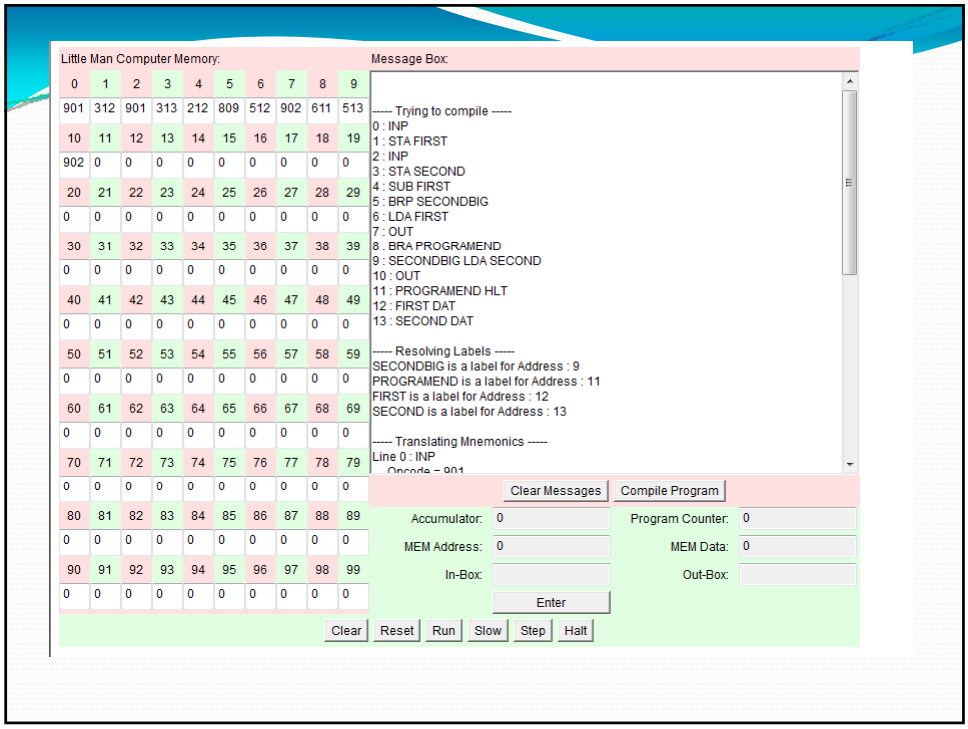
Input a number and store at labelled location FIRST.

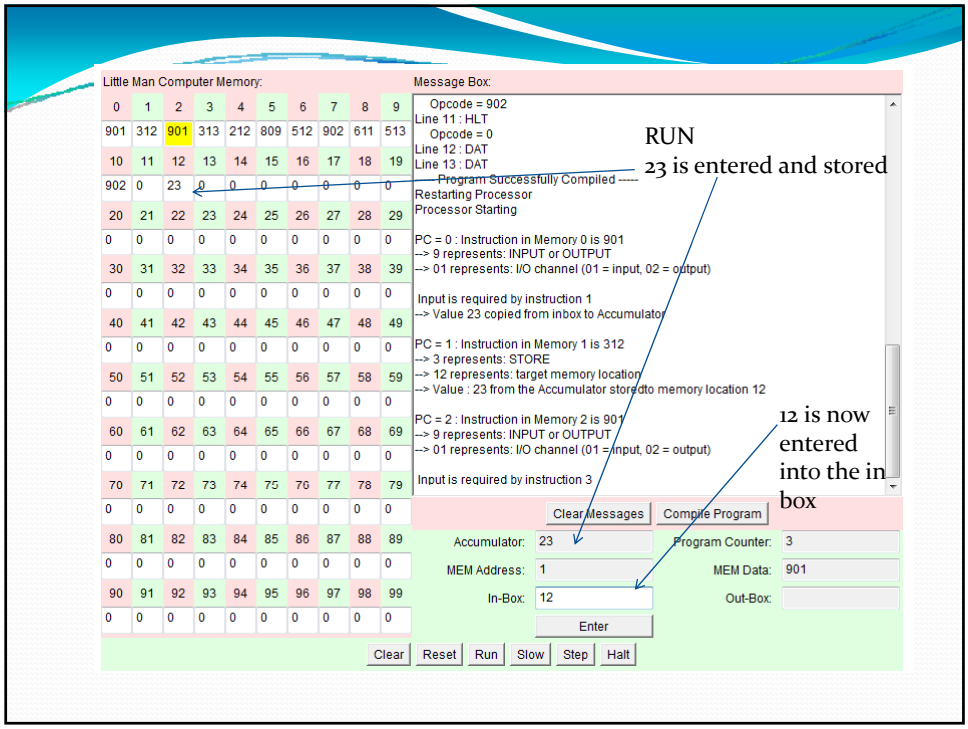
Input second number and store at location SECOND

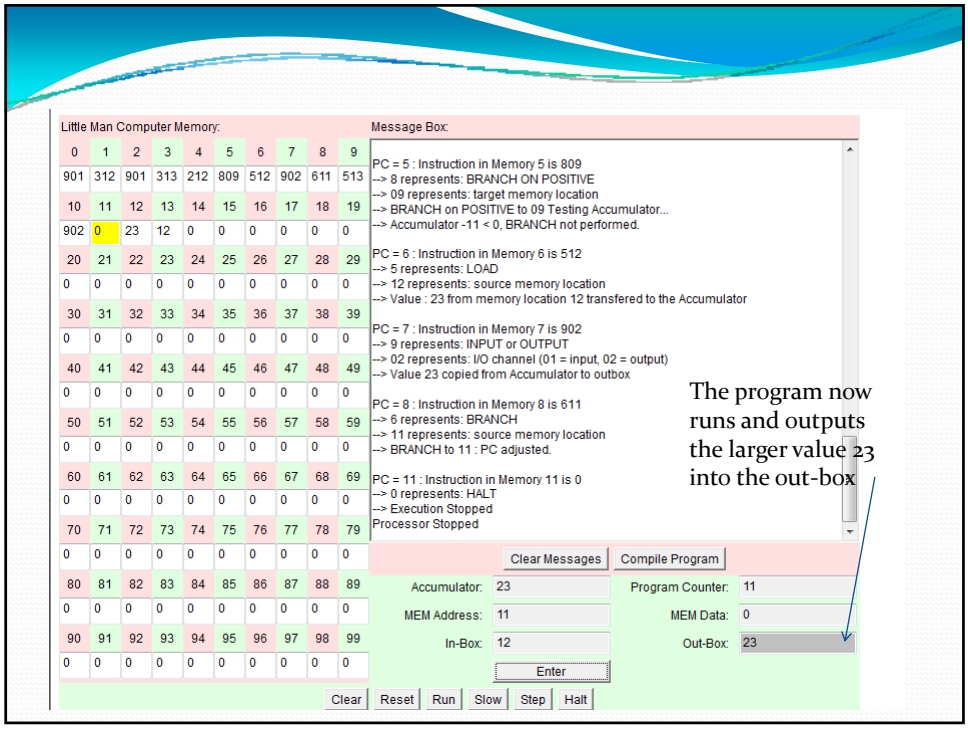
SECOND is still in the accumulator so subtract FIRST
If the result is positive SECOND is bigger so branch to the label SECONDBIG
IF not FIRST is bigger so Load it to the accumulator and output the value.
Done so branch to the end of the program

This is the label SECONDBIG so load SECOND to the accumulator and output it
At the label PROGRAMEND halt the program

These are the storage labels for FIRST and SECOND


Little Man Computer Memory:
Message Box:
----- Trying to compile -----
0 : INP
1 : STA FIRST
2 : INP
3 : STA SECOND
4 : SUB FIRST
5 : BRP SECONDBIG
6 : LDA FIRST
7 : OUT
8 . BRA PROGRAMEND
9 : SECONDBIG LDA SECOND
10 : OUT
11 : PROGRAMEND HLT
12 : FIRST DAT
13 : SECOND DAT
----- Resolving Labels -----
SECONDBIG is a label for Address : 9
PROGRAMEND is a label for Address : 11
FIRST is a label for Address : 12
SECOND is a label for Address : 13
----- Translating Mnemonics -----
Line 0 : INP
Clear Messages
Compile Program
Accumulator: 0
Program Counter: 0
MEM Address: 0
MEM Data: 0
In-Box:
Out-Box:
Enter
Clear
Reset
Run
Slow
Step
Halt


Little Man Computer Memory:
Message Box:
Opcode = 902
Line 11 : HLT
Opcode = 0
Line 12 : DAT
Line 13 : DAT
----- Program Successfully Compiled -----
Restarting Processor
Processor Starting
PC = 0 : Instruction in Memory 0 is 901
--> 9 represents: INPUT or OUTPUT
--> 01 represents: I/O channel (01 = input, 02 = output)
Input is required by instruction 1
--> Value 23 copied from inbox to Accumulator
PC = 1 : Instruction in Memory 1 is 312
--> 3 represents: STORE
--> 12 represents: target memory location
--> Value : 23 from the Accumulator stored to memory location 12
PC = 2 : Instruction in Memory 2 is 901
--> 9 represents: INPUT or OUTPUT
--> 01 represents: I/O channel (01 = input, 02 = output)
Input is required by instruction 3
RUN
23 is entered and stored
12 is now entered into the in box
Clear Messages
Compile Program
Accumulator: 23
Program Counter: 3
MEM Address: 1
MEM Data: 901
In-Box: 12
Out-Box:
Enter
Clear
Reset
Run
Slow
Step
Halt



PC = 0 : Instruction in Memory 0 is 901
--> 9 represents: INPUT or OUTPUT
--> 01 represents: I/O channel (01 = input, 02 = output)

Input is required by instruction 1
--> Value 23 copied from inbox to Accumulator

PC = 1 : Instruction in Memory 1 is 312
--> 3 represents: STORE
--> 12 represents: target memory location
--> Value : 23 from the Accumulator storedto memory location 12

PC = 2 : Instruction in Memory 2 is 901
--> 9 represents: INPUT or OUTPUT
--> 01 represents: I/O channel (01 = input, 02 = output)

Input is required by instruction 3
--> Value 12 copied from inbox to Accumulator

PC = 3 : Instruction in Memory 3 is 313
--> 3 represents: STORE
--> 13 represents: target memory location
--> Value : 12 from the Accumulator storedto memory location 13

PC = 4 : Instruction in Memory 4 is 212
--> 2 represents: SUBTRACT
--> 12 represents: source memory location
--> Value : 23 from memory location 12 subtracted from the Accumulator

PC = 5 : Instruction in Memory 5 is 809
--> 8 represents: BRANCH ON POSITIVE
--> 09 represents: target memory location
--> BRANCH on POSITIVE to 09 Testing Accumulator...
--> Accumulator -11 < 0, BRANCH not performed.

PC = 6 : Instruction in Memory 6 is 512
--> 5 represents: LOAD
--> 12 represents: source memory location
--> Value : 23 from memory location 12 transfered to the Accumulator

PC = 7 : Instruction in Memory 7 is 902
--> 9 represents: INPUT or OUTPUT
--> 02 represents: I/O channel (01 = input, 02 = output)
--> Value 23 copied from Accumulator to outbox

PC = 8 : Instruction in Memory 8 is 611
--> 6 represents: BRANCH
--> 11 represents: source memory location
--> BRANCH to 11 : PC adjusted.

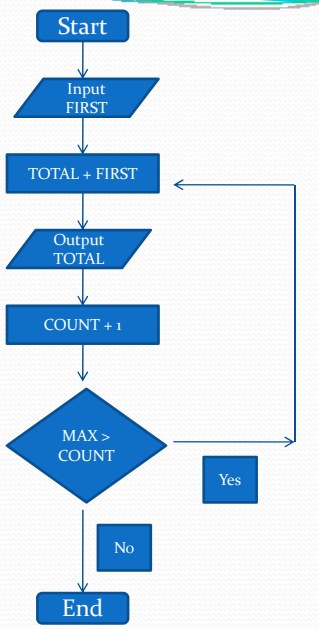
PC = 11 : Instruction in Memory 11 is 0
--> 0 represents: HALT
--> Execution Stopped
Processor Stopped

Candidates should now explain this output from LMC.

A452

Produce a multiplication table from 1 to 10 for any number input by the user

Flowchart



The Algorithm

- Store MAX as 10
- Store STEP as 1
- Input first number
- Store as FIRST
- Subsection *maxbigger:*
- Load TOTAL to accumulator
- Add FIRST to accumulator
- Store as TOTAL
- Output accumulator
- Load COUNT to accumulator
- Add STEP to accumulator
- Store as COUNT
- Subtract COUNT from MAX
- If the result is positive MAX is bigger so go to subsection *maxbigger*
- Stop

LMC Code

```
INP
STA FIRST
MAXBIGGER LDA TOTAL
ADD FIRST
STA TOTAL
OUT
LDA COUNT
ADD STEP
STA COUNT
LDA MAX
SUB COUNT
BRP MAXBIGGER
HLT
FIRST DAT
TOTAL DAT
COUNT DAT 1
MAX  DAT 10
STEP DAT 1
```

Input a number and store at labelled location FIRST.

Input second number and store at location SECOND

SECOND is still in the accumulator so subtract FIRST
If the result is positive SECOND is bigger so branch to the label SECONDBIG
IF not FIRST is bigger so Load it to the accumulator and output the value.
Done so branch to the end of the program

This is the label SECONDBIG so load SECOND to the accumulator and output it
At the label PROGRAMEND halt the program

These are the storage labels for each variable