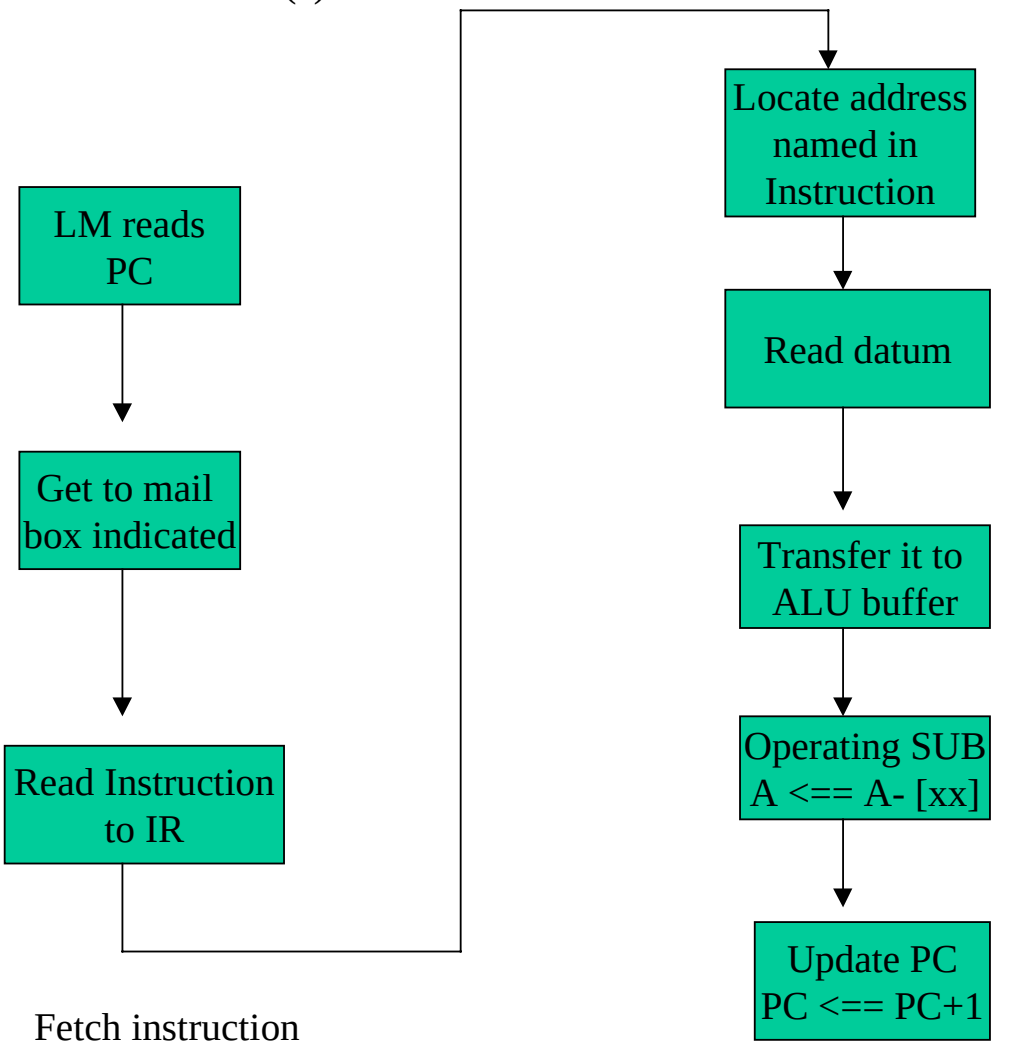
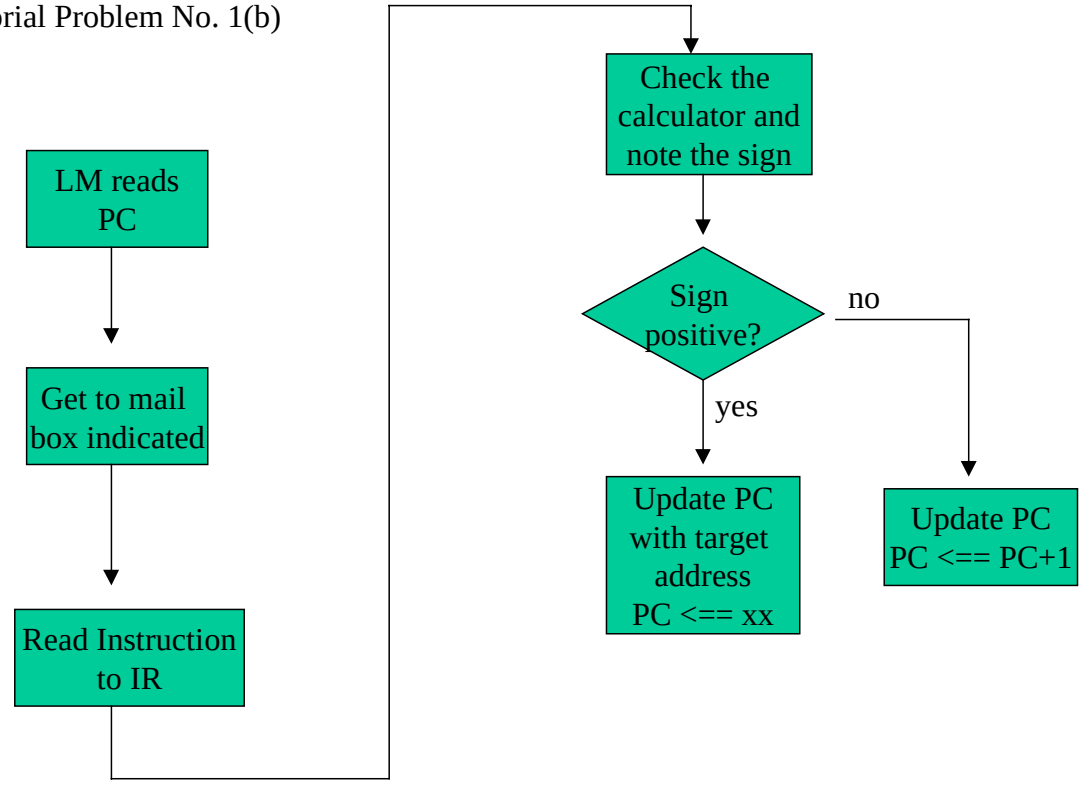
Answers to Tutorial Problems - LMC

Tutorial Problem No. 1(a)



Tutorial Problem No. 1(b)



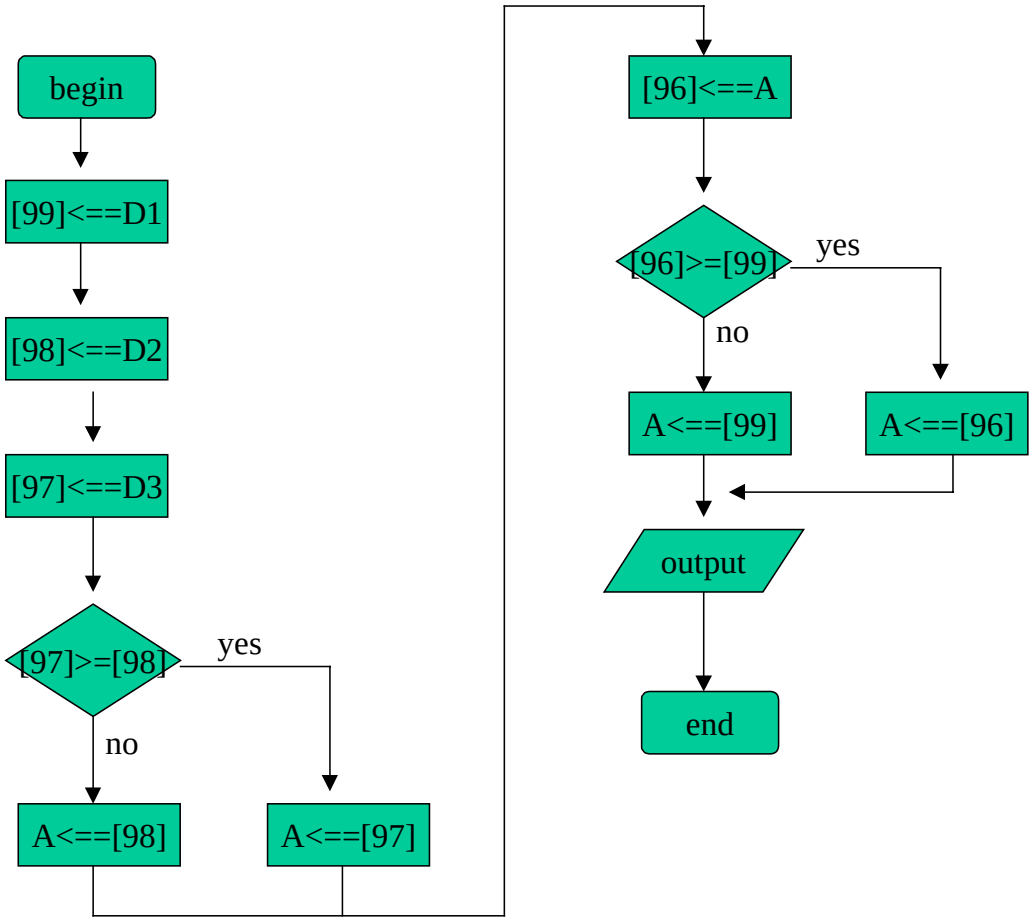
Fetch instruction

Execute Instruction

2

```
00      INPUT
01      STO     99          ;keep the first datum, D1
02      INPUT
03      STO     98          ;keep the second datum, D2
04      INPUT
05      STO     97          ;keep the third datum, D3
06      SUB     98          ;D3 - D2
07      BP      10          ;D3>=D2
08      LOAD    98          ;A <= D2
09      B       11
10      LOAD    97          ;A <= D3
11      STO     96          ;A <= max{D2,D3}
12      SUB     99
13      BP      16
14      LOAD    99
15      B       17
16      LOAD    96
17      OUTPUT
18      STOP
```

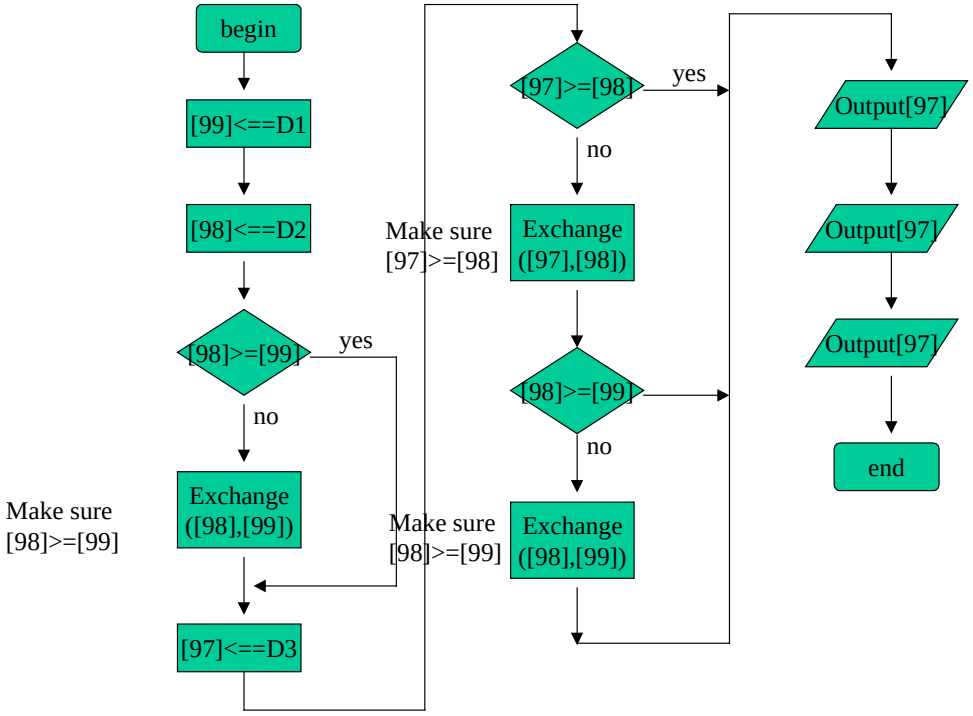
Tutorial Problem No. 2--output the max{d1, d2, d3}



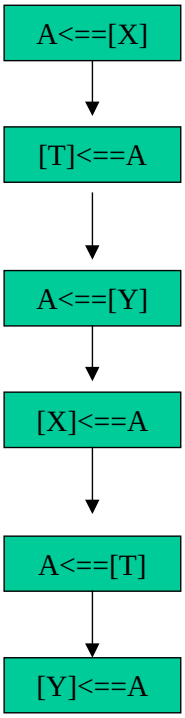
3

```
00      INPUT
01      STO     99
02      INPUT
03      STO     98
04      SUB     99
05      BP      12
06      LOAD    99          ;swap [98] and [99]
07      STO     96          ;temp
08      LOAD    98
09      STO     99
10      LOAD    96
11      STO     98
12      INPUT
13      STO     97
14      SUB     98
15      BP      30          ;number in order (97,98,99)
16      LOAD    97          ;swap [97] and [98]
17      STO     96
18      LOAD    98
19      STO     97
20      LOAD    96
21      STO     98
22      SUB     99          ;compare [98] and [99]
23      BP      30          ; number in order (98, 99)
24      LOAD    98          ;swap [98] and [99]
25      STO     96
26      LOAD    99
27      STO     98
28      LOAD    96
29      STO     99
30      LOAD    97
31      OUTPUT
32      LOAD    98
33      OUTPUT
34      LOAD    99
35      OUTPUT
36      STOP
```

Tutorial Problem No. 3 -- to sort out three numbers from largest to smallest {D1, D2, D3}



Tutorial Problem No. 3 ---- exchange([X], [Y]) flow chart
exchange the content between two memory cells, we use temporary memory [T] to make it



4

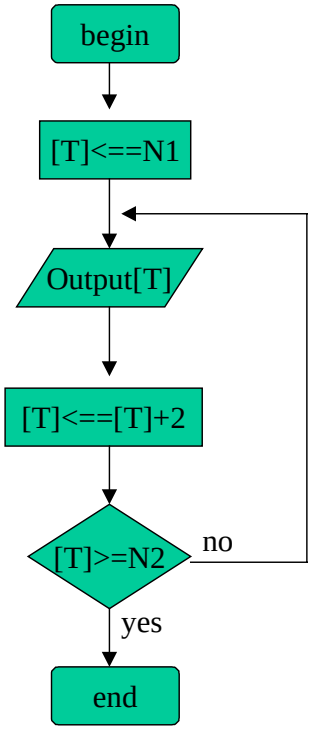
```
01      LOAD   11
02      STO    14          ;temporary memory initialisation

03      LOAD   14
04      OUTPUT
05      ADD    13
06      STO    14
07      SUB    12
08      BP     10
09      B      03
10      STOP

11      DAT    001         ;the first number
12      DAT    100         ;the upper bound condition
13      DAT    002         ;the step value
14      DAT    000         ;temporary memory
```

Tutorial Problem No. 4

N1 is the first odd number, we set it to 1 in the memory, N2 is the last number to control repetition, we set it to 100. We set the incremental step length to 2, so that the sequence of odd number can be produced accordingly. [T] is temporary memory cell for this purpose.



5,1

example
if (A>=B) then do_task_1; else do task_2
We have put A in mailbox #99, B in mailbox #98

```
        LOAD   99
        SUB    98
        BP     Task_1
Task_2: Instruction_1
        Instruction_2
        …
        B      Exit
Task_1: Instruction_1
        Instruction_2
        …
        B      Exit
Exit:   other_instruction_sequence
```

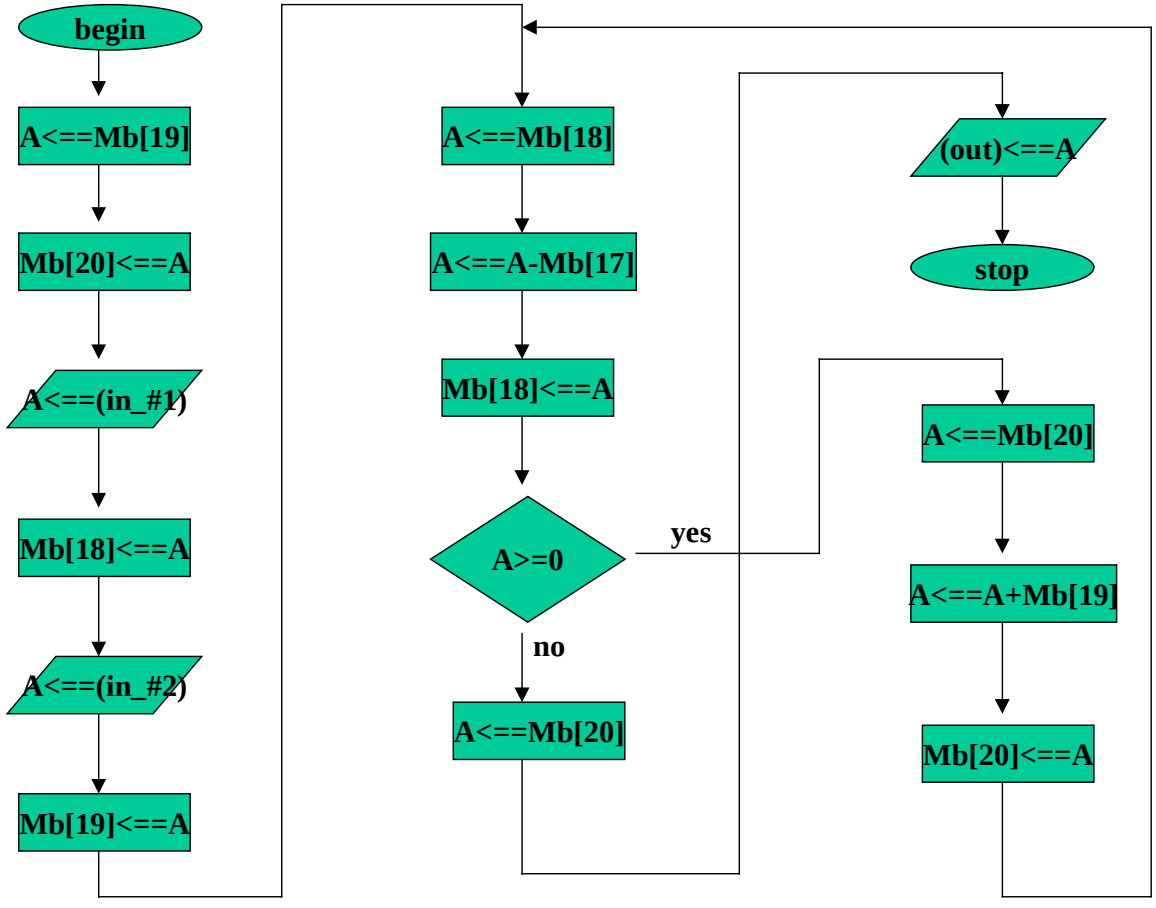
5,2
example
repeat the task_1, until the condition (A < B) is met
We put A in mailbox #99, B in mailbox #98, mailbox 97 is the 002, the decrement step is of 2

```
Task_1: Instruction_1
        Instruction_2
        …
        LOAD   99
        SUB    97
        STO    99          ;change the condition to make sure not trap in deadlock

        LOAD   99          ;this instruction can be omitted, but it's no harm to put it here
        SUB    98
        BP     Task_1
```

7

00	519	LOAD	19	A<= MBox[19];	A=000
01	320	STO	20	MBox[20]<=A;	MBox[20]=000
02	901	INPUT		A <= (input_1);	
03	318	STO	18	MBox[18] <= A;	MBox[18]=(input_1)
04	901	INPUT		A <= (input_2);	
05	319	STO	19	MBox[19] <= A;	MBox[19]=(input_2)
06	518	LOAD	18	A <= MBox[18];	A=(input_1)
07	217	SUB	17	A <= A - MBox[17];	A=(input_1 - 1)
08	318	STO	18	MBox[18] <= A;	MBox[18]=(input_1 - 1)
09	813	BP	13	if (A>=0) then PC <= 13; else PC<=09+1;	
10	520	LOAD	20	A <= MBox[20];	
11	902	OUTPUT		print the content in A	
12	000	STOP			
13	520	LOAD	20	A <= MBox[20];	
14	119	ADD	19	A <= A + MBox[19]	
15	320	STO	20	MBox[20] <= A;	
16	606	B	06	PC <= 06	
17	001	---			
18	000	---			
19	000	---			
20	000	---			



Tutorial Problem No.7

After each of the instruction is executed, the content data in Accumulator and Memory are shown below: This can be a part of your draft to keep to your own when you are working out with this problem.

Addr.	A	[17]	[18]	[19]	[20]	
Init.	**---**	**001**	**000**	**000**	**000**	
00	**000**					
01					**000**	
02	**002**					
03			**002**			
04	**004**					
05				**004**		
06	**002**					
07	**001**					
08			**001**			
09						**PC ← 13**
13	**000**					
14	**004**					
15					**004**	
16						**PC ← 06**
06	**001**					
07	**000**					
08			**000**			
09						**PC ← 13**
13	**004**					
14	**008**					
15					**008**	
16						**PC ← 06**
06	**000**					
07	**-001**					
08			**001**			
09						**PC ← PC+1**
10	**008**					
11	**output(008)**					
12	**stop**					

The output is 8 for the given inputs. (the first input datum is 2, the second one is 4)

The program does a multiplication. (You can change the inputs to different values, the output is the product of the inputs)