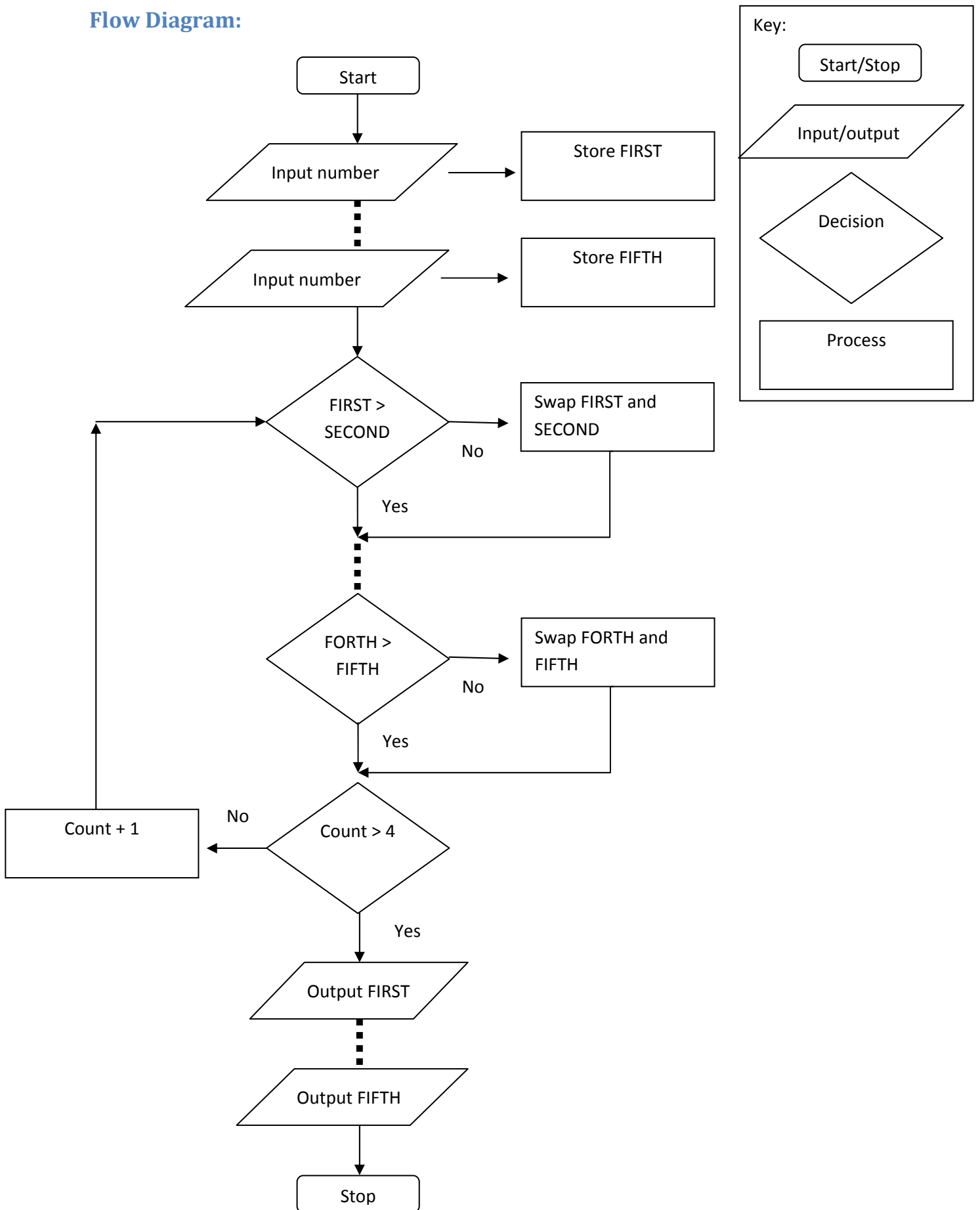


### Problem 3: Input five numbers and output in descending order

#### Flow Diagram:



## Algorithm (Pseudo Code):

- Set MAX as 4
  - Set STEP as 1
  - Input first number
  - Store as FIRST
  - Input second number
  - Store as SECOND
  - Input third number
  - Store as THIRD
  - Input fourth number
  - Store as FORTH
  - Input fifth number
  - Store as FIFTH
  - STARTLOOP
  - Load COUNT
  - Add STEP
  - Store COUNT
- Load FIRST in to accumulator
  - Subtract SECOND
  - If result is positive goto FIRSTBIGGER
  - Load FIRST
  - Store FIRST as TEMP
  - Load SECOND
  - Store as FIRST
  - Load TEMP
  - Store as SECOND

- FIRSTBIGGER
- LOAD SECOND in to accumulator
- Subtract THIRD
- If result is positive goto SECONDBIGGER
- Load SECOND
- Store SECOND as TEMP
- Load THIRD
- Store as SECOND
- Load TEMP
- Store as THIRD

- SECONDBIGGER
- LOAD THIRD in to accumulator
- Subtract FORTH
- If result is positive goto THIRDBIGGER
- Load THIRD
- Store THIRD as TEMP
- Load FORTH
- Store as THIRD
- Load TEMP
- Store as FORTH

- THIRDBIGGER
- LOAD FOURTH in to accumulator
- Subtract FIFTH
- If result is positive goto FOURTHBIGGER
- Load FOURTH
- Store FOURTH as TEMP
- Load FIFTH
- Store as FOURTH
- Load TEMP
- Store as FIFTH

- FORTHBIGGER
- Load COUNT
- Subtract MAX
- Branch if Zero ENDLOOP
- Branch Always STARTLOOP
- **ENDLOOP**
- Output FIRST
- Output SECOND
- Output THIRD
- Output FORTH
- Output FIFTH
- Halt

## Assembly Code

<i>Assembly Code</i>	<i>Explanation</i>
MAX DAT 5	The value of MAX is set to 5 to ensure program will loop 4 times
STEP DAT 1	The initial value of STEP is step to 1
INP	First input required from user
STA FIRST	Store input as FIRST
INP	Second input required from user
STA SECOND	Store input as SECOND
INP	Third input required from user
STA THIRD	Store input as THIRD
INP	Forth input required from user
STA FORTH	Store input as FORTH
INP	Fifth input required from user
STA FIFTH	Store input as FIFTH
STARTLOOP LDA COUNT	
ADD STEP	
STA COUNT	
LDA FIRST	
SUB SECOND	
BRP 1STBIGGER	
LDA FIRST	
STA TEMP	
LDA SECOND	
STA FIRST	
LDA TEMP	
STA SECOND	
1STBIGGER LDA SECOND	
SUB THIRD	
BRP 2NDBIGGER	
LDA SECOND	
STA TEMP	
LDA THIRD	
STA SECOND	
LDA TEMP	
STA THIRD	
2NDBIGGER LDA THIRD	
SUB FORTH	
BRP 3RDBIGGER	
LDA THIRD	
STA TEMP	
LDA FORTH	
STA THIRD	
LDA TEMP	
STA FORTH	
3RDBIGGER LDA FORTH	
SUB FIFTH	
BRP 4THBIGGER	
LDA FORTH	

STA TEMP LDA FIFTH STA FORTH LDA TEMP STA FIFTH 4THBIGGER LDA COUNT SUB MAX BRZ ENDLOOP BRA STARTLOOP ENDLOOP LDA FIRST OUT LDA SECOND OUT LDA THIRD OUT LDA FORTH OUT LDA FIFTH OUT HLT FIRST DAT SECOND DAT THIRD DAT FORTH DAT FIFTH DAT TEMP DAT COUNT DAT 1	
--	--

## LMC Screenshots

**Little Man Computer Memory:**

0	1	2	3	4	5	6	7	8	9
5	1	901	366	901	367	901	368	901	369
				14	15	16	17	18	19
				372	566	267	824	566	371
20	21	22	23	24	25	26	27	28	29
567	366	571	367	567	268	833	567	371	568
30	31	32	33	34	35	36	37	38	39
367	571	368	568	269	842	568	371	569	368
40	41	42	43	44	45	46	47	48	49
571	369	569	270	851	569	371	570	369	571
50	51	52	53	54	55	56	57	58	59
370	572	200	755	612	566	902	567	902	568
60	61	62	63	64	65	66	67	68	69
902	569	902	570	902	0	0	0	0	0
70	71	72	73	74	75	76	77	78	79
0	0	1	0	0	0	0	0	0	0
80	81	82					7	88	89
0	0	0					0	0	
90	91	92	93	94	95	96	97	98	99
0	0	0	0	0	0	0	0	0	0

**Message Box:**

```

Line 57: LDA
  Opcode = 5 Address = 67
Line 58: OUT
  Opcode = 902
Line 59: LDA
  Opcode = 5 Address = 68
Line 60: OUT
  Opcode = 902
Line 61: LDA
  Opcode = 5 Address = 69
Line 62: OUT
  Opcode = 902
Line 63: LDA
  Opcode = 5 Address = 70
Line 64: OUT
  Opcode = 902
Line 65: HLT
  Opcode = 0
Line 66: DAT
Line 67: DAT
Line 68: DAT
Line 69: DAT
Line 70: DAT
Line 71: DAT
Line 72: DAT
----- Program Successfully Compiled -----
    
```

Max set to 5  
Step to 1

Count set to 1

Fig1: Problem3 after assembly code has been compiled to machine code

Accumulator:

MEM Address:

In-Box:

Program Counter:

MEM Data:

Out-Box:

Little Man Computer Memory:										Message Box:
0	1	2	3	4	5	6	7	8	9	--> 68 represents: target memory location --> Value : 3 from the Accumulator stored to memory location 68
5	1	901	366	901	367	901	368	901	369	PC = 8 : Instruction in Memory 8 is 901 --> 9 represents: INPUT or OUTPUT --> 01 represents: I/O channel (01 = input, 02 = output)
10	11	12	13	14	15	16	17	18	19	Input is required by instruction 11 --> Value 5 copied from inbox to Accumulator
901	370	572	101	372	566	267	824	566	371	PC = 9 : --> 3 represents: STORE --> 69 represents: target memory location --> Value : 4 from the Accumulator stored to memory location 69
20	21	22	23	24	25	26	27	28	29	PC = 10 : Instruction in Memory 10 is 901 --> 9 represents: INPUT or OUTPUT --> 01 represents: I/O channel (01 = input, 02 = output)
567	366	571	367	567	268	833	567	371	568	Input is required by instruction 11 --> Value 5 copied from inbox to Accumulator
30	31	32	33	34	35	36	37	38	39	PC = 11 : Instruction in Memory 11 is 370 --> 3 represents: STORE --> 70 represents: target memory location --> Value : 5 from the Accumulator stored to memory location 70
367	571	368	568	269	842	568	371	569	368	
40	41	42	43	44	45	46	47	48	49	
571	369	569	270	851	569	371	570	369	571	
50	51	52	53	54	55	56	57	58	59	
370	572	200	755	612	566	902	567	902	568	
60	61	62	63	64	65	66	67	68	69	
902	569	902	570	902	0	1	2	3	4	
70	71	72	73	74	75	76	77	78	79	
5	0	1								
80	81	82								
0	0	0	0	0	0	0	0	0	0	
90	91	92	93	94	95	96	97	98	99	
0	0	0	0	0	0	0	0	0	0	

**Fig 2:** Problem 3 is running and user has entered the numbers 1,2,3,4,5

Numbers 1-5 have been entered and stored

Clear Messages    Compile Program

Accumulator:     Program Counter:

MEM Address:     MEM Data:

In-Box:     Out-Box:



The screenshot shows the Little Man Computer (LMC) simulator. The memory table is as follows:

0	1	2	3	4	5	6	7	8	9
5	1	901	366	901	367	901	368	901	369
10	11	12	13	14	15	16	17	18	19
901	370	572	101	372	566	267	824	566	371
20	21	22	23	24	25	26	27	28	29
567	366	571	367	567	268	833	567	371	568
30	31	32	33	34	35	36	37	38	39
367	571	368	568	269	842	568	371	569	368
40	41	42	43	44	45	46	47	48	49
571	369	569	270	851	569	371	570	369	571
50	51	52	53	54	55	56	57	58	59
567	902	568	67	68	69	70	71	72	73
4	5	0	0	0	0	0	0	0	0
80	81	82	83	84	85	86	87	88	89
0	0	0	0	0	0	0	0	0	0
90	91	92	93	94	95	96	97	98	99
0	0	0	0	0	0	0	0	0	0

The message box shows the following execution log:

```

--> Value 3 copied from Accumulator to outbox
PC = 61 : Instruction in Memory 61 is 569
--> 5 represents: LOAD
--> 69 represents: source memory location
--> Value : 2 from memory location 69 transferred to the Accumulator
PC = 62 : Instruction in Memory 62 is 902
--> 9 represents: INPUT or OUTPUT
--> 02 represents: I/O channel (01 = input, 02 = output)
--> Value 2 copied from Accumulator to outbox
PC = 63 : Instruction in Memory 63 is 570
--> 5 represents: LOAD
--> 70 represents: source memory location
--> Value : 1 from memory location 70 transferred to the Accumulator
PC = 64 : Instruction in Memory 64 is 902
--> 9 represents: INPUT or OUTPUT
--> 02 represents: I/O channel (01 = input, 02 = output)
--> Value 1 copied from Accumulator to outbox
PC = 65 : Instruction in Memory 65 is 0
--> 0 represents: HALT
--> Execution Stopped
Processor Stopped
  
```

Callout boxes in the image provide additional context:

- "The inputted numbers have been sorted into ascending order and outputted" (referring to memory locations 70-79).
- "Fig 3: Program 3 has completed run and number have been outputted" (referring to the message box).
- "COUNT is equal to MAX" (referring to the value 5 in memory location 1).

## Output:

LMC Output	Explanation
PC = 0 : Instruction in Memory 0 is 5 Invalid Instruction	Set the value of memory location 0 to 5 this called MAX
PC = 1 : Instruction in Memory 1 is 1 Invalid Instruction	Set the value of memory location 1 to 1
PC = 2 : Instruction in Memory 2 is 901 --> 9 represents: INPUT or OUTPUT --> 01 represents: I/O channel (01 = input, 02 = output)	Requests input from user
Input is required by instruction 3 --> Value 1 copied from inbox to Accumulator	User input is 1
PC = 3 : Instruction in Memory 3 is 366 --> 3 represents: STORE --> 66 represents: target memory location --> Value : 1 from the Accumulator stored to memory location 66	User input is stored in memory location 66 (FIRST)

PC = 4 : Instruction in Memory 4 is 901  
--> 9 represents: INPUT or OUTPUT  
--> 01 represents: I/O channel (01 = input, 02 = output)

Input is required by instruction 5  
--> Value 2 copied from inbox to Accumulator

PC = 5 : Instruction in Memory 5 is 367  
--> 3 represents: STORE  
--> 67 represents: target memory location  
--> Value : 2 from the Accumulator stored to memory location 67

PC = 6 : Instruction in Memory 6 is 901  
--> 9 represents: INPUT or OUTPUT  
--> 01 represents: I/O channel (01 = input, 02 = output)

Input is required by instruction 7  
--> Value 3 copied from inbox to Accumulator

PC = 7 : Instruction in Memory 7 is 368  
--> 3 represents: STORE  
--> 68 represents: target memory location  
--> Value : 3 from the Accumulator stored to memory location 68

PC = 8 : Instruction in Memory 8 is 901  
--> 9 represents: INPUT or OUTPUT  
--> 01 represents: I/O channel (01 = input, 02 = output)

Input is required by instruction 9  
--> Value 4 copied from inbox to Accumulator

PC = 9 : Instruction in Memory 9 is 369  
--> 3 represents: STORE  
--> 69 represents: target memory location  
--> Value : 4 from the Accumulator stored to memory location 69

PC = 10 : Instruction in Memory 10 is 901  
--> 9 represents: INPUT or OUTPUT  
--> 01 represents: I/O channel (01 = input, 02 = output)

Input is required by instruction 11  
--> Value 5 copied from inbox to Accumulator

Requests input from user

User input is 2

User input is stored in memory location 67 (SECOND)

Requests input from user

User input is 3

User input is stored in memory location 68 (THIRD)

Requests input from user

User input is 4

User input is stored in memory location 69 (FORTH)

Requests input from user

User input is 5

PC = 11 : Instruction in Memory 11 is 370  
--> 3 represents: STORE  
--> 70 represents: target memory location  
--> Value : 5 from the Accumulator stored to memory location 70

PC = 12 : Instruction in Memory 12 is 572  
--> 5 represents: LOAD  
--> 72 represents: source memory location  
--> Value : 1 from memory location 72 transferred to the Accumulator

PC = 13 : Instruction in Memory 13 is 101  
--> 1 represents: ADD  
--> 01 represents: source memory location  
--> Value : 1 from memory location 01 added to the Accumulator

PC = 14 : Instruction in Memory 14 is 372  
--> 3 represents: STORE  
--> 72 represents: target memory location  
--> Value : 2 from the Accumulator stored to memory location 72

PC = 15 : Instruction in Memory 15 is 566  
--> 5 represents: LOAD  
--> 66 represents: source memory location  
--> Value : 1 from memory location 66 transferred to the Accumulator

PC = 16 : Instruction in Memory 16 is 267  
--> 2 represents: SUBTRACT  
--> 67 represents: source memory location  
--> Value : 2 from memory location 67 subtracted from the Accumulator

PC = 17 : Instruction in Memory 17 is 824  
--> 8 represents: BRANCH ON POSITIVE  
--> 24 represents: target memory location  
--> BRANCH on POSITIVE to 24 Testing Accumulator...  
--> Accumulator  $-1 < 0$ , BRANCH not performed.

PC = 18 : Instruction in Memory 18 is 566  
--> 5 represents: LOAD  
--> 66 represents: source memory location  
--> Value : 1 from memory location 66 transferred to the Accumulator

User input is stored in memory location 70 (FIFTH)

The Variable COUNT is loaded from location 72 to the accumulator

The Variable STEP is added to the accumulator

The result replaces COUNT at location 72

FIRST is loaded to the accumulator

SECOND is subtracted from FIRST

IF the result is +ve program jumps to Location 24 (1STBIGGER)

In this case the result is -ve so the program doesn't jump

FIRST is loaded to the accumulator

PC = 19 : Instruction in Memory 19 is 371  
--> 3 represents: STORE  
--> 71 represents: target memory location  
--> Value : 1 from the Accumulator stored to memory location 71

FIRST is stored as TEMP (Location 71)

PC = 20 : Instruction in Memory 20 is 567  
--> 5 represents: LOAD  
--> 67 represents: source memory location  
--> Value : 2 from memory location 67 transferred to the Accumulator

SECOND is loaded to accumulator

PC = 21 : Instruction in Memory 21 is 366  
--> 3 represents: STORE  
--> 66 represents: target memory location  
--> Value : 2 from the Accumulator stored to memory location 66

SECOND is stored as FIRST

PC = 22 : Instruction in Memory 22 is 571  
--> 5 represents: LOAD  
--> 71 represents: source memory location  
--> Value : 1 from memory location 71 transferred to the Accumulator

TEMP is loaded to the accumulator

PC = 23 : Instruction in Memory 23 is 367  
--> 3 represents: STORE  
--> 67 represents: target memory location  
--> Value : 1 from the Accumulator stored to memory location 67

TEMP is stored as SECOND

PC = 24 : Instruction in Memory 24 is 567  
--> 5 represents: LOAD  
--> 67 represents: source memory location  
--> Value : 1 from memory location 67 transferred to the Accumulator

SECOND is loaded to the accumulator

PC = 25 : Instruction in Memory 25 is 268  
--> 2 represents: SUBTRACT  
--> 68 represents: source memory location  
--> Value : 3 from memory location 68 subtracted from the Accumulator

THIRD is subtracted from SECOND

PC = 26 : Instruction in Memory 26 is 833  
--> 8 represents: BRANCH ON POSITIVE  
--> 33 represents: target memory location  
--> BRANCH on POSITIVE to 33 Testing Accumulator...  
--> Accumulator  $-2 < 0$ , BRANCH not performed.

IF the result is +ve program jumps to Location 33 (2<sup>nd</sup> BIGGER)

In this case the result is -ve so the program doesn't jump

PC = 27 : Instruction in Memory 27 is 567  
--> 5 represents: LOAD  
--> 67 represents: source memory location  
--> Value : 1 from memory location 67 transferred to the Accumulator

SECOND is loaded to the accumulator

PC = 28 : Instruction in Memory 28 is 371  
--> 3 represents: STORE  
--> 71 represents: target memory location  
--> Value : 1 from the Accumulator stored to memory location 71

SECOND is stored as TEMP (Location 71)

PC = 29 : Instruction in Memory 29 is 568  
--> 5 represents: LOAD  
--> 68 represents: source memory location  
--> Value : 3 from memory location 68 transferred to the Accumulator

THIRD is loaded to accumulator

PC = 30 : Instruction in Memory 30 is 367  
--> 3 represents: STORE  
--> 67 represents: target memory location  
--> Value : 3 from the Accumulator stored to memory location 67

THIRD is stored as SECOND

PC = 31 : Instruction in Memory 31 is 571  
--> 5 represents: LOAD  
--> 71 represents: source memory location  
--> Value : 1 from memory location 71 transferred to the Accumulator

TEMP is loaded to the accumulator

PC = 32 : Instruction in Memory 32 is 368  
--> 3 represents: STORE  
--> 68 represents: target memory location  
--> Value : 1 from the Accumulator stored to memory location 68

TEMP is stored as THIRD

PC = 33 : Instruction in Memory 33 is 568  
--> 5 represents: LOAD  
--> 68 represents: source memory location  
--> Value : 1 from memory location 68 transferred to the Accumulator

THIRD is loaded to the accumulator

PC = 34 : Instruction in Memory 34 is 269  
--> 2 represents: SUBTRACT  
--> 69 represents: source memory location  
--> Value : 4 from memory location 69 subtracted from the Accumulator

FORTH is subtracted from THIRD

PC = 35 : Instruction in Memory 35 is 842  
--> 8 represents: BRANCH ON POSITIVE  
--> 42 represents: target memory location  
--> BRANCH on POSITIVE to 42 Testing  
Accumulator...  
--> Accumulator  $-3 < 0$ , BRANCH not performed.

IF the result is +ve program jumps to Location 42 (3rdBIGGER)

In this case the result is -ve so the program doesn't jump

PC = 36 : Instruction in Memory 36 is 568  
--> 5 represents: LOAD  
--> 68 represents: source memory location  
--> Value : 1 from memory location 68 transfered to the Accumulator

THIRD is loaded to the accumulator

PC = 37 : Instruction in Memory 37 is 371  
--> 3 represents: STORE  
--> 71 represents: target memory location  
--> Value : 1 from the Accumulator stored to memory location 71

THIRD is stored as TEMP (Location 71)

PC = 38 : Instruction in Memory 38 is 569  
--> 5 represents: LOAD  
--> 69 represents: source memory location  
--> Value : 4 from memory location 69 transfered to the Accumulator

FORTH is loaded to accumulator

PC = 39 : Instruction in Memory 39 is 368  
--> 3 represents: STORE  
--> 68 represents: target memory location  
--> Value : 4 from the Accumulator stored to memory location 68

FORTH is stored as THIRD

PC = 40 : Instruction in Memory 40 is 571  
--> 5 represents: LOAD  
--> 71 represents: source memory location  
--> Value : 1 from memory location 71 transfered to the Accumulator

TEMP is loaded to the accumulator

PC = 41 : Instruction in Memory 41 is 369  
--> 3 represents: STORE  
--> 69 represents: target memory location  
--> Value : 1 from the Accumulator stored to memory location 69

TEMP is stored as FORTH

PC = 42 : Instruction in Memory 42 is 569  
--> 5 represents: LOAD  
--> 69 represents: source memory location  
--> Value : 1 from memory location 69 transfered to the Accumulator

FORTH is loaded to the accumulator

PC = 43 : Instruction in Memory 43 is 270  
--> 2 represents: SUBTRACT  
--> 70 represents: source memory location  
--> Value : 5 from memory location 70  
subtracted from the Accumulator

FIFTH is subtracted from FORTH

PC = 44 : Instruction in Memory 44 is 851  
--> 8 represents: BRANCH ON POSITIVE  
--> 51 represents: target memory location  
--> BRANCH on POSITIVE to 51 Testing  
Accumulator...  
--> Accumulator  $-4 < 0$ , BRANCH not performed.

IF the result is +ve program jumps to Location 51  
(4THBIGGER)

In this case the result is -ve so the program  
doesn't jump

PC = 45 : Instruction in Memory 45 is 569  
--> 5 represents: LOAD  
--> 69 represents: source memory location  
--> Value : 1 from memory location 69 transfered  
to the Accumulator

FORTH is loaded to the accumulator

PC = 46 : Instruction in Memory 46 is 371  
--> 3 represents: STORE  
--> 71 represents: target memory location  
--> Value : 1 from the Accumulator storedto  
memory location 71

FORTH is stored as TEMP (Location 71)

PC = 47 : Instruction in Memory 47 is 570  
--> 5 represents: LOAD  
--> 70 represents: source memory location  
--> Value : 5 from memory location 70 transfered  
to the Accumulator

FIFTH is loaded to accumulator

PC = 48 : Instruction in Memory 48 is 369  
--> 3 represents: STORE  
--> 69 represents: target memory location  
--> Value : 5 from the Accumulator storedto  
memory location 69

FIFTH is stored as FORTH

PC = 49 : Instruction in Memory 49 is 571  
--> 5 represents: LOAD  
--> 71 represents: source memory location  
--> Value : 1 from memory location 71 transfered  
to the Accumulator

TEMP is loaded to the accumulator

PC = 50 : Instruction in Memory 50 is 370  
--> 3 represents: STORE  
--> 70 represents: target memory location  
--> Value : 1 from the Accumulator storedto  
memory location 70

TEMP is stored as FIFTH

PC = 51 : Instruction in Memory 51 is 572  
--> 5 represents: LOAD  
--> 72 represents: source memory location  
--> Value : 2 from memory location 72 transferred to the Accumulator

PC = 52 : Instruction in Memory 52 is 200  
--> 2 represents: SUBTRACT  
--> 00 represents: source memory location  
--> Value : 5 from memory location 00 subtracted from the Accumulator

PC = 53 : Instruction in Memory 53 is 755  
--> 7 represents: BRANCH ON ZERO  
--> 55 represents: target memory location  
--> BRANCH on ZERO to 55 : Testing Accumulator...  
--> Accumulator -3 not equal to 0, Branch NOT performed.

PC = 54 : Instruction in Memory 54 is 612  
--> 6 represents: BRANCH  
--> 12 represents: source memory location  
--> BRANCH to 12 : PC adjusted.

PC = 55 : Instruction in Memory 55 is 566  
--> 5 represents: LOAD  
--> 66 represents: source memory location  
--> Value : 5 from memory location 66 transferred to the Accumulator

PC = 56 : Instruction in Memory 56 is 902  
--> 9 represents: INPUT or OUTPUT  
--> 02 represents: I/O channel (01 = input, 02 = output)  
--> Value 5 copied from Accumulator to outbox

PC = 57 : Instruction in Memory 57 is 567  
--> 5 represents: LOAD  
--> 67 represents: source memory location  
--> Value : 4 from memory location 67 transferred to the Accumulator

PC = 58 : Instruction in Memory 58 is 902  
--> 9 represents: INPUT or OUTPUT  
--> 02 represents: I/O channel (01 = input, 02 = output)  
--> Value 4 copied from Accumulator to outbox



PC = 59 : Instruction in Memory 59 is 568  
--> 5 represents: LOAD  
--> 68 represents: source memory location  
--> Value : 3 from memory location 68 transferred to the Accumulator

PC = 60 : Instruction in Memory 60 is 902  
--> 9 represents: INPUT or OUTPUT  
--> 02 represents: I/O channel (01 = input, 02 = output)  
--> Value 3 copied from Accumulator to outbox

PC = 61 : Instruction in Memory 61 is 569  
--> 5 represents: LOAD  
--> 69 represents: source memory location  
--> Value : 2 from memory location 69 transferred to the Accumulator

PC = 62 : Instruction in Memory 62 is 902  
--> 9 represents: INPUT or OUTPUT  
--> 02 represents: I/O channel (01 = input, 02 = output)  
--> Value 2 copied from Accumulator to outbox

PC = 63 : Instruction in Memory 63 is 570  
--> 5 represents: LOAD  
--> 70 represents: source memory location  
--> Value : 1 from memory location 70 transferred to the Accumulator

PC = 64 : Instruction in Memory 64 is 902  
--> 9 represents: INPUT or OUTPUT  
--> 02 represents: I/O channel (01 = input, 02 = output)  
--> Value 1 copied from Accumulator to outbox

PC = 65 : Instruction in Memory 65 is 0  
--> 0 represents: HALT  
--> Execution Stopped  
Processor Stopped