Sang Hwa Lee, Hieu (Sean) Tran

INST 346

December 19, 2022

Team Project Prototype Sprint

Advanced LMC program

LMC program simulator link : CPU simulator - Little Man Computer .

The LMC simulator will allow you to see what is happening during the instruction cycle. The LMC consists of a central processing unit (CPU), a memory unit, and an input/output (I/O) device. The CPU of the LMC is designed to execute a set of basic instructions, such as load, store, add, and branch. These instructions are stored in memory and are executed by the CPU in a sequential manner. We are using the LMC with simple assembly language to learn the basics of low-level programming and how computers execute instructions.

You can load programs into the LMC simulator by clicking in the left-most box under where it says, “Assembly Language Code” and hitting the “Submit” button. You can run the program by pressing the “Run” button or step through the program by hitting the “Step” button. You are also able to increase or decrease the speed of the simulation by hitting the “<<” or “>>” buttons while the program is running. Red bubbles contain addresses. Blue bubbles contain values.

The LMC instruction set is included below for your reference.

Instructions

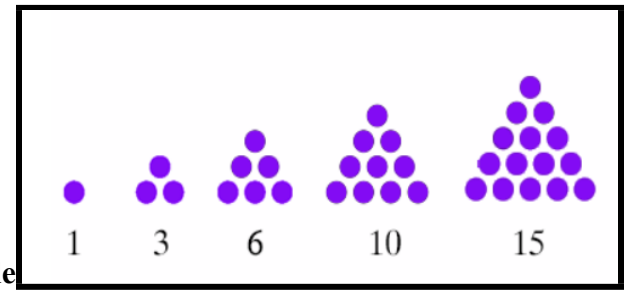
Instruction	Mnemonic	Opcode	Operand	Description
Halt	HLT	0	00	Stop running
Add	ADD	1	XX*	Add the contents of the memory address to the Accumulator
Subtract	SUB	2	XX*	Subtract the contents of the memory address from the Accumulator
Store	STA	3	XX*	Store the contents in the Accumulator in the memory address
Load	LDA	5	XX*	Load Accumulator with contents of memory address
Branch Always	BRA	6	XX*	Set the PC to the value in the operand
Branch if zero	BRZ	7	XX*	Set the PC to the operand if the Accumulator is zero
Branch if zero or positive	BRP	8	XX*	Set the PC to the operand if the contents of the Accumulator is zero or positive
Input	INP	9	01	Retrieve user input and store in the Accumulator
Output	OUT	9	02	Output the contents of the Accumulator

Data Storage	DAT			Label for a memory address; also can have a contents specified

Triangle number

In order to create an LMC program related to triangle number, first need to know what rules triangle number is completed. The number of points to form the shape of an equilateral triangle is called triangle number. It's easier to understand through examples.

Rule



1 = 1 3 = 1+2

6 = 1+2+3 10 = 1+2+3+4

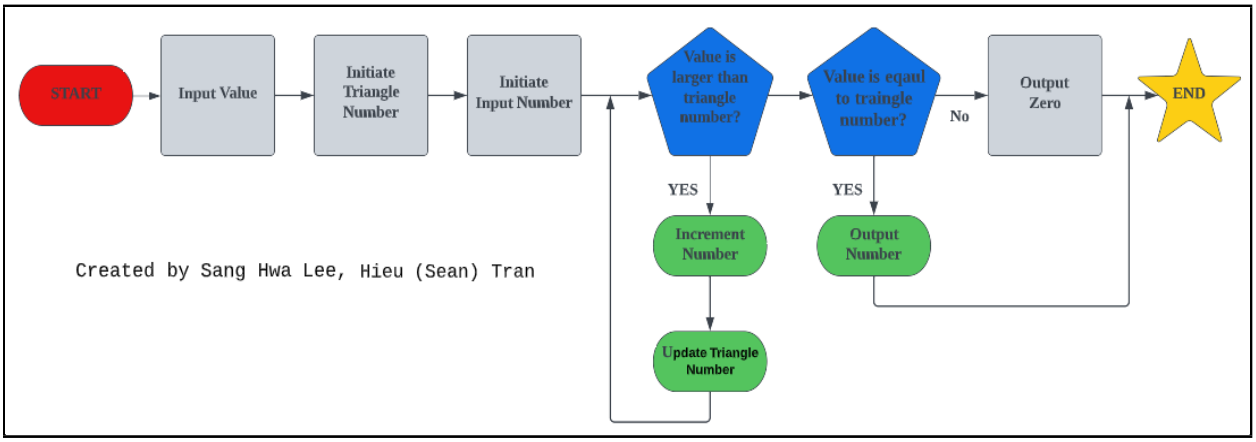
15 = 1+2+3+4+5 21 = 1+2+3+4+5+6

Triangle numbers can be represented by the formula T(n) = n(n+1)/2, where n is the nth triangle number.

Flowchart for LMC programming

A brief flowchart of how these LMC programs can be produced is needed. When the program starts first, accept the value user input. The value is then calculated until the inputted value is greater than or equal to the value of the calculated triangle. If the number is the same as the triangle, can derive the value of the triangle number. If the user's value does not match the

triangle number, a loop is used to derive 0.



Learning Objectives

After completing this lab, you should be able to:

- Use the LMC by executing a set of basic instructions, such as load, store, add, and branch by putting simple assembly language into the LMC's input device, and be able to understand and utilize the output.
- Secure knowledge on the fundamentals of low-level programming, assembly language, and how computers execute instructions.

Instructions on how to complete the lab.

Triangle Program Code	Detail	Steps
INP STA V LDA Z STA Tnum STA N	V : Value Z : Zero Tnum : Triangle number N : Number EL : End loop L : Loop = : Equal	1. Click the LMC link to start simulation 2. Input the program code 3. Click submit button 4. Click the run button

```
L       LDA Tnum
        SUB V
        BRP EL
        LDA N
        ADD One
        STA N
        ADD Tnum
        STA Tnum
        BRA L
EL      LDA V
        SUB Tnum
        BRZ =
        LDA Z
        OUT
        BRA DONE
=       LDA N
        OUT
DONE    HLT
N       DAT 000
Tnum    DAT 000
V       DAT 000
Z       DAT 000
One     DAT 001
```

5. Input user’s number. It can be a triangle number or any number is fine.
6. If the number 0 is output, the user's input number is not a triangle number.
7. If a number bigger than zero is output, it means that the user's input number is a triangle number. Also can check the nth triangle number. ex) If output is 6, it becomes the sixth triangle number. In other words, the user inputted 21.

Questions

1. If you input 666, which output will you get? Can you get the value of the nth triangle?
2. When the program is run, what number must the user enter to get the 32nd triangle number?
3. If so, describe in which mailboxes the user input(N), the current triangle number value (Tnum), the calculated triangle number, the ZERO(Z) , and One are stored in this program.
4. Loops play a very important role in this program. If so, describe the code for the loop steps used in this program.

Explain these steps

```
L       LDA Tnum
        SUB V
        BRP EL
        LDA N
        ADD One
        STA N
        ADD Tnum
        STA Tnum
        BRA L
```

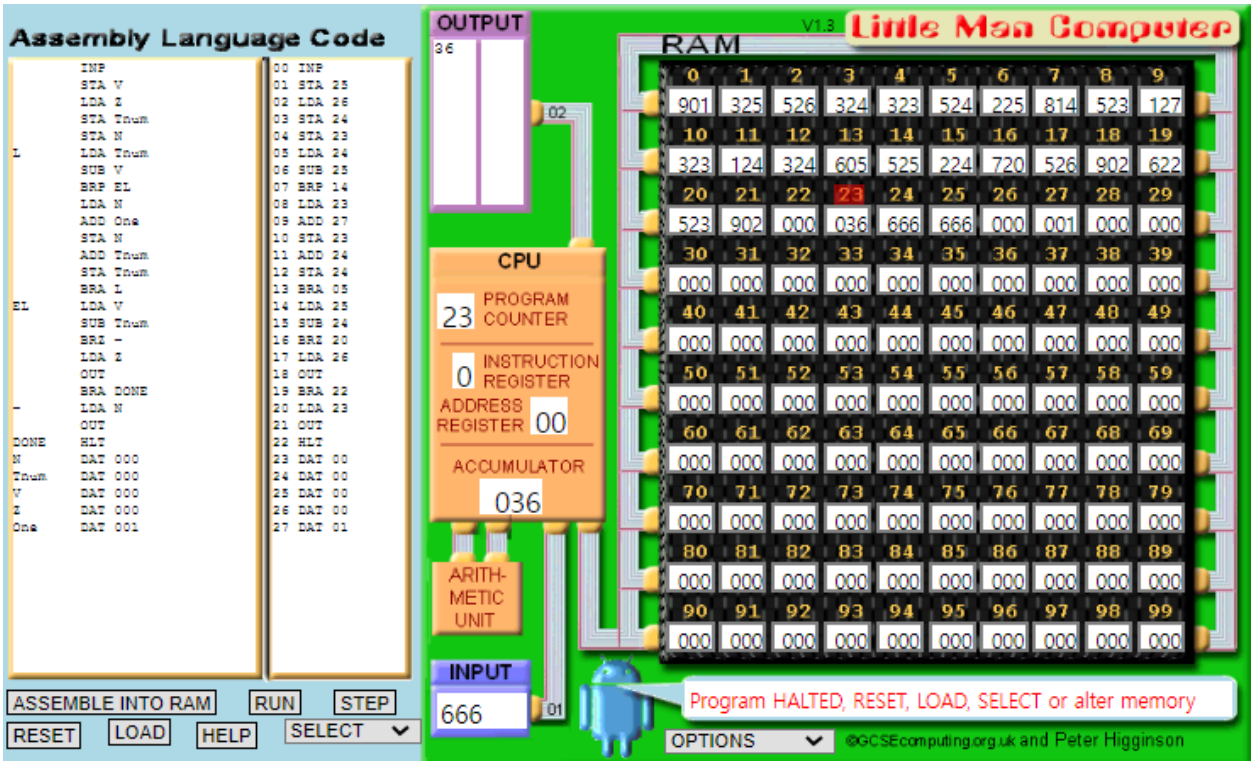
Explain Questions and answers.

https://docs.google.com/presentation/d/1rRWnZqBeUcJwQnN3SecoYjZbAegGsYV0nohAnhIx6vU/edit?usp=sharing

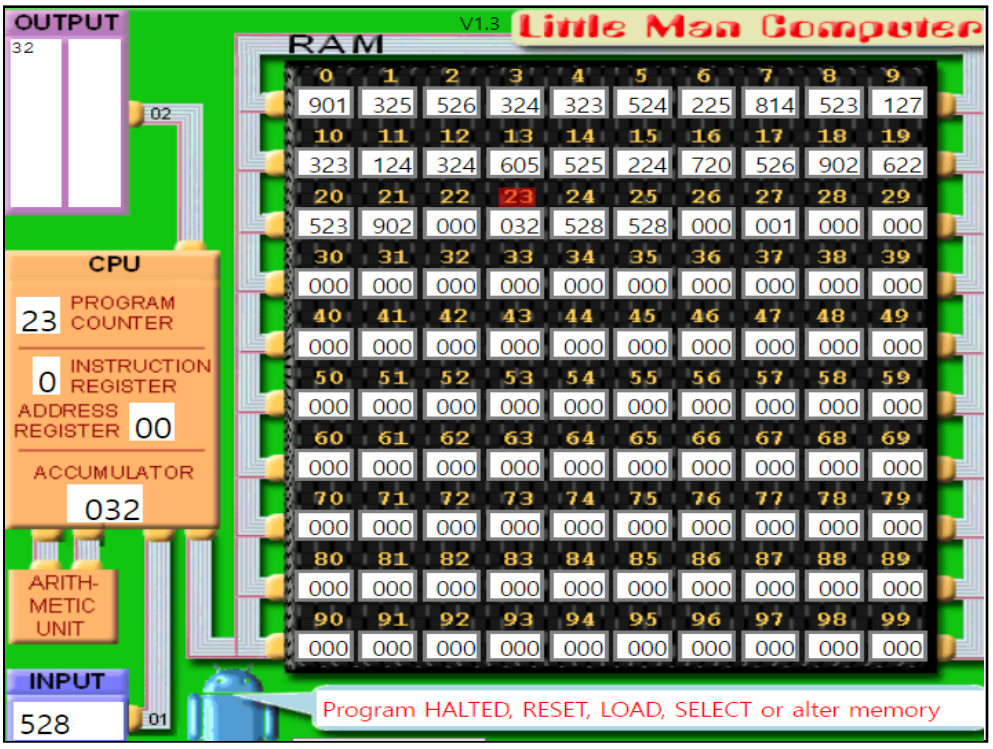
(include voice)

Answers

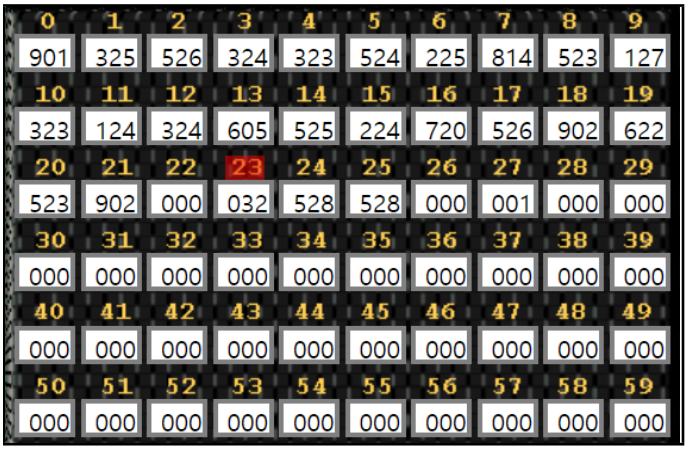
1. Yes, if user input 666, user can get 36 output. That means 666 is the value of the 36th triangle number.



2. User has to input 528.



3. The user's input value is 25, Tnum is 24, the calculated number of triangles is 23, Zero is 26 and One is 27 and stored in the mailbox.



0	1	2	3	4	5	6	7	8	9
901	325	526	324	323	524	225	814	523	127
10	11	12	13	14	15	16	17	18	19
323	124	324	605	525	224	720	526	902	622
20	21	22	23	24	25	26	27	28	29
523	902	000	032	528	528	000	001	000	000
30	31	32	33	34	35	36	37	38	39
000	000	000	000	000	000	000	000	000	000
40	41	42	43	44	45	46	47	48	49
000	000	000	000	000	000	000	000	000	000
50	51	52	53	54	55	56	57	58	59
000	000	000	000	000	000	000	000	000	000

Simulation when input number is 528.

4.

L LDA Tnum	-	Load the value of the triangle number stored in the mailbox and enter it in the accumulator.
SUB V	-	The inputted value is subtracted from the triangle number.
BRP EL	-	If the result is negative, keep trying to loop to calculate the triangle number.
LDA N	-	Load number value
ADD One	-	To calculate the next number of triangles, the number continues incremented by one.
STA N	-	The value of the new number is saved.
ADD Tnum	-	The new stored number is added to the triangle number.
STA Tnum	-	The new triangle number is saved
BRA L	-	Indicates the end of the loop. Set it to return to the beginning of the loop.

Works Cited

Maths in a minute: Triangular numbers. Plus Maths. (1970, November 24). Retrieved December 19, 2022, from https://plus.maths.org/content/maths-minute-triangular-numbers

Wikimedia Foundation. (2020, January 27). *Little man computer*. Wikipedia. Retrieved December 19, 2022, from https://en.wikipedia.org/w/index.php?title=Little_man_computer&oldid=937810035

Littel Man Computer Program Simulator. Little man computer - CPU simulator. (n.d.). Retrieved December 19, 2022, from http://peterhigginson.co.uk/LMC/

Part of the description of Professor Heidenblad's INST346 Lab03

(we thought that it explained the LMC elements well so there was no point in changing it)