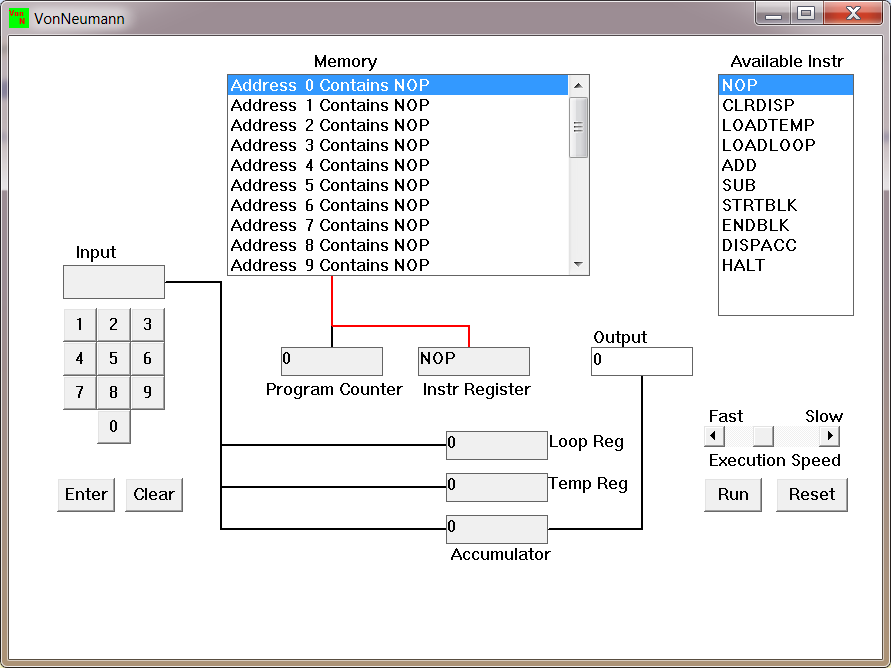
# 4CS015 Fundamentals of Computing – Task-3

**Workshop tasks:**

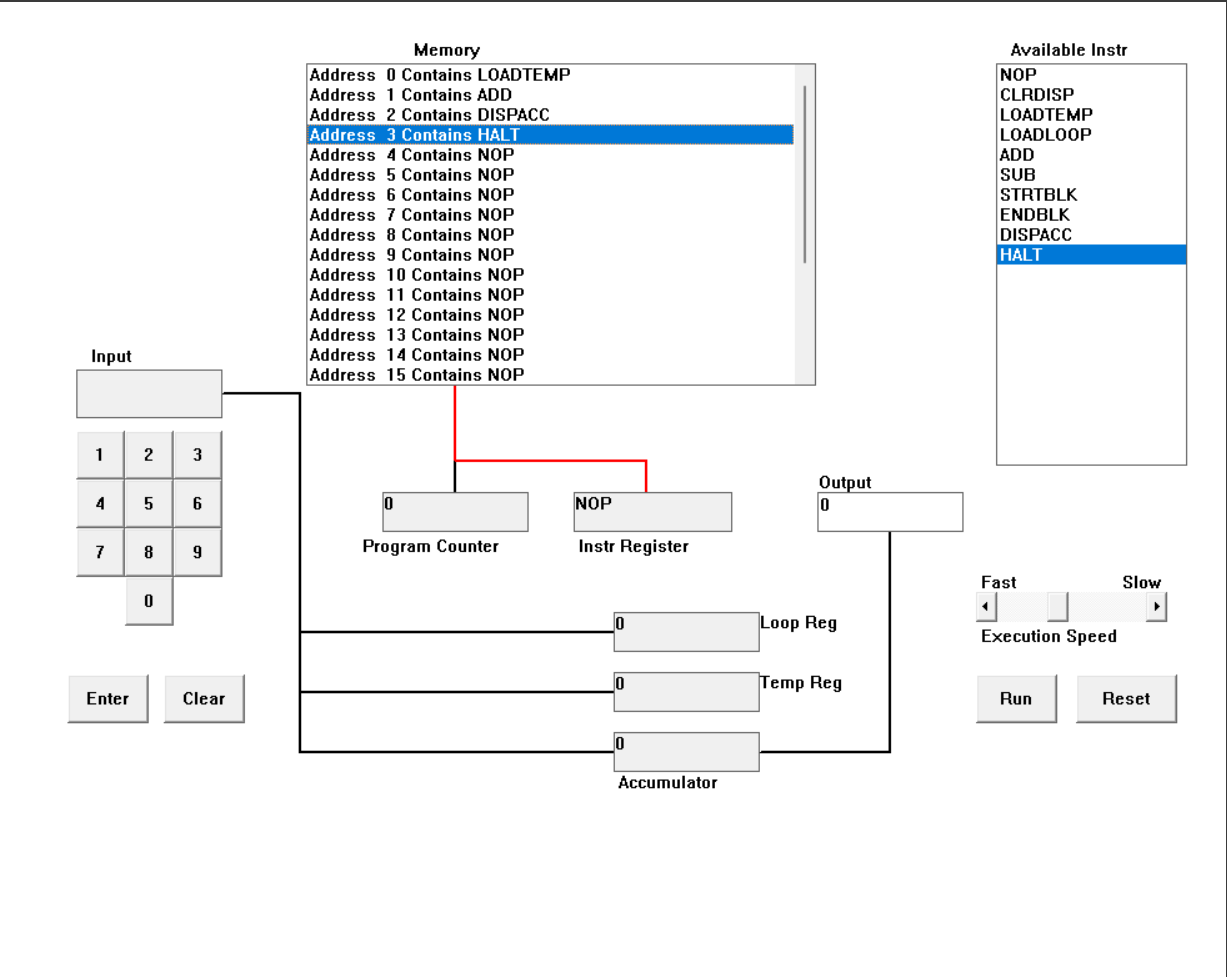
1. Von Neumann Simulator. This program simulates a very simple computer with the von Neumann architecture.
   1. Download the von Neumann Simulator (VonNeumann.exe) program from WOLF in the Week 5 folder. Save it in your Documents folder and run it. You will see a window similar to this:

  
The simulator has a small program memory area which is available for programming. To enter your program instructions simply click on the “Available” instruction on the list on the right and then click on the “Memory” location you wish to put it in.

This simulator understands only the following ten instructions:

| NOP | No Operation, i.e. do nothing. |
| --- | --- |
| LOADTEMP | Get a number from the keypad, completed by the Enter key, into the Temporary Register. |
| LOADLOOP | Get a number from the keypad, completed by the Enter key, into the Loop Register. |
| CLRDISP | Clear the Display. |
| ADD | Add the Temporary Register to the Accumulator |
| SUB | Subtract the Temporary Register from the Accumulator |
| DISPACC | Display the contents of the Accumulator |
| STRTBLK | Start of Loop Block |
| ENDBLK | End of Loop Block |
| HALT | Halt. Stop Program |

* 1. Load the following program into the memory:  
     LOADTEMP  
     ADD  
     DISPACC  
     HALT  
       
     To do this, first click on the “LOADTEMP” in the list of instructions on the right of simulator window. Then click on Memory location with “Address 0 Contains NOP”. This will then change into “Address 0 Contains LOADTEMP”. Repeat the process with “Address 1” and so on until the whole program is loaded.

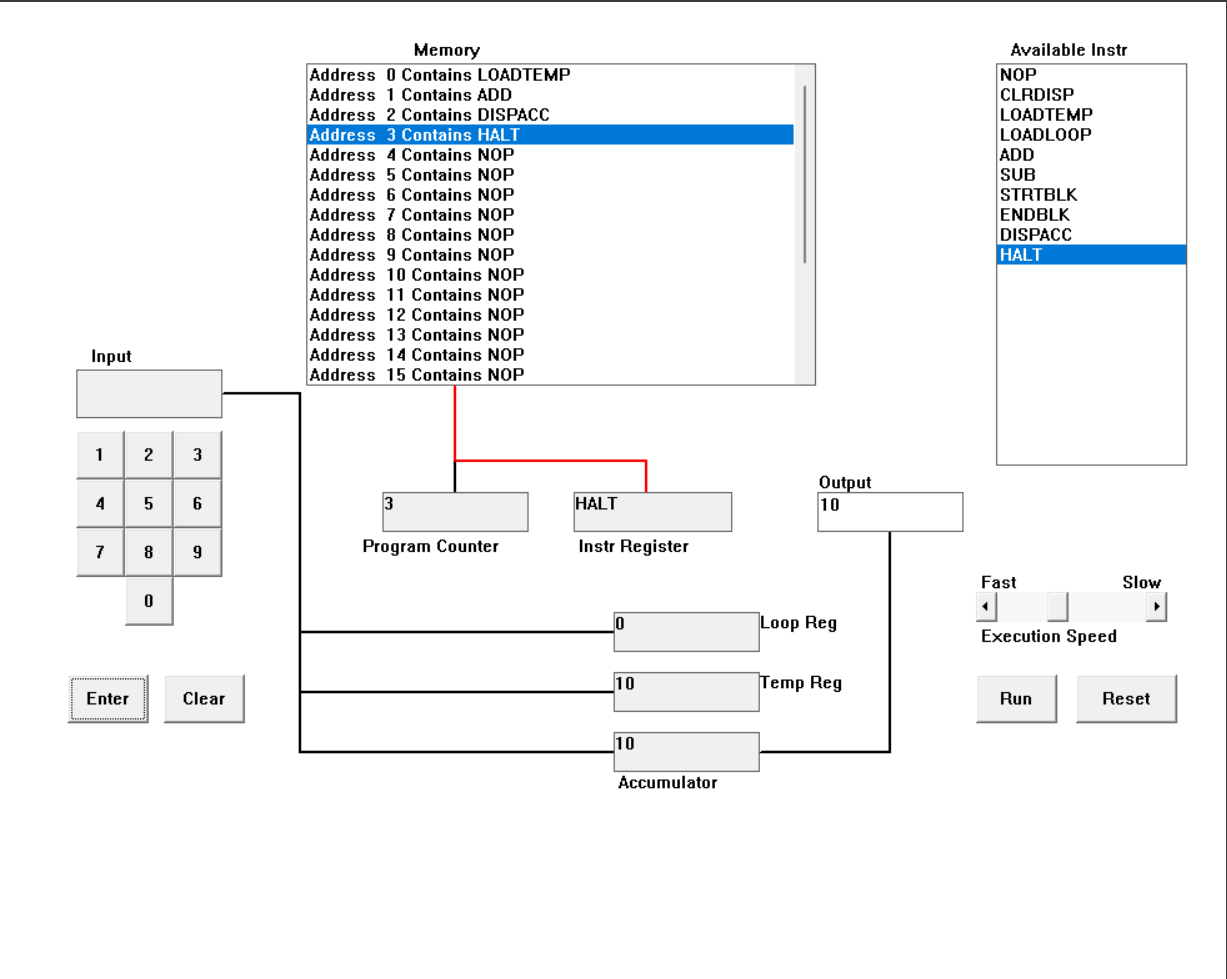


The following program displays the number entered. The program executes in the following ways.

First the LOADTEMP executes which asks to enter a number through the number pad at the left side of the program. The second instruction ADD adds the value at the accumulator to the number stored in the temporary register. Initially the number in the accumulator and in the temporary register is zero. When the user enters the number by the LOADTEMP instruction, the number is stored in the temporary register. So, the number entered by the user is added to the number in the accumulator (zero). The third instruction is DISPACC. The DISPACC instruction displays the content of the accumulator. The last instruction is HALT. This instruction stops the program.

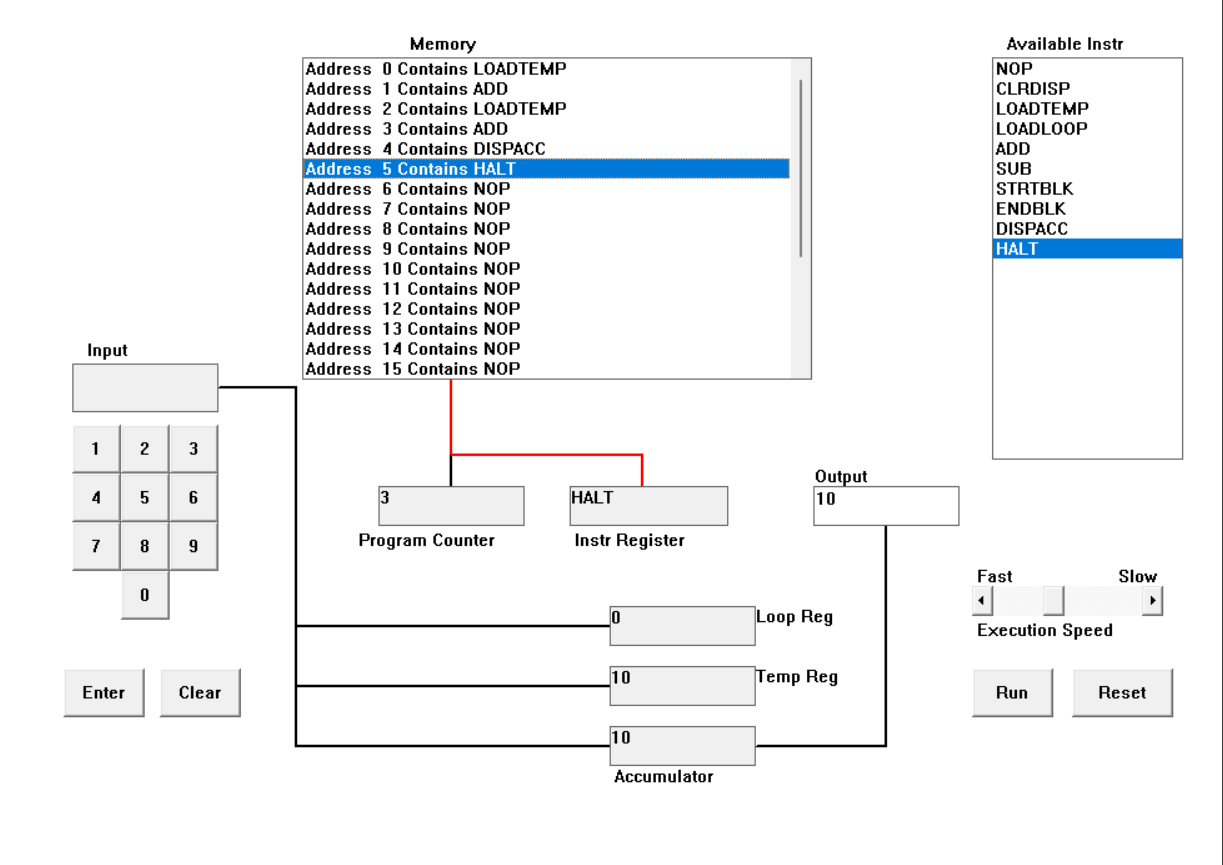
In this way, the program executes.

* 1. Run the program by clicking on the “Run” button. The simulator would highlight the Address 0 location and then pause. It is executing the instruction “LOADTEMP” which requires you to input a number into the keypad.   
       
     Click 2 or 3 numbers on the keypad and then click the “Enter” button. The simulator will then resume running the program and execute the instruction “ADD”. This adds the number that you just entered, to the zero in the accumulator.   
       
     The next instruction is “DISPACC” which stands for “Display Accumulator”, and it does exactly that. After that the simulator stops running the program when it executes the instruction “HALT”.



The following program runs exactly like the program mentioned in the first question.

In this program LOADTEMP asks for the input, in which 10 is entered. The program then continues and ADD instruction is executed ehich adds the number that is entered, in this case the number is 10, to the zero in the accumulator. The DISPACC will be executed which displays the content of the accumulator. 10 is displayed as it will be stored in the accumulator. Then HALT instruction gets executed and the program will stop.

* 1. Load the following program into the simulator:  
     LOADTEMP  
     ADD  
     LOADTEMP  
     ADD  
     DISPACC  
     HALT  
       
     What do you think it does? Write your answer below (10 marks)

The program adds two numbers entered by the user. The following Von Neumann program runs in the following way.

First instruction is LOADTEMP which asks the user to input the number. The next instruction is ADD which adds the first number that is entered by user, to the number which is in the accumulator (in this case is 0). The third instruction is LOADTEMP which again asks the user to input the number after that the next instruction is ADD which is added to the number stored in the temporary register (in this case is the first number). The next instruction is “DISPACC” which stands for “Display Accumulator”, and it displays the value which is stored in the Accumulator. The last instruction in the program is HALT which stops the program.

* 1. Write a program to add 3 numbers together. List your program below (10 marks)

The program to add 3 numbers together is given below.

LOADTEMP

ADD

LOADTEMP

ADD

LOADTEMP

ADD

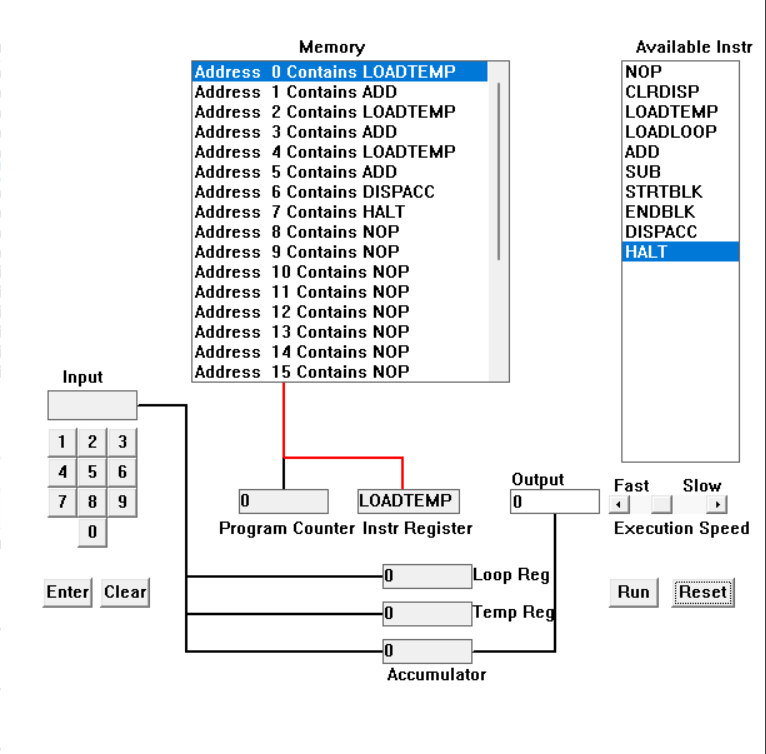
DISPACC

HALT

The program given above adds three numbers together. The program runs in the way as mentioned below.

First instruction is LOADTEMP which asks the user to enter a number. The next instruction is ADD which adds the first number that is entered by the user, to the number which is in the accumulator (initially the number is 0). The third instruction is LOADTEMP (second number) which again asks the user to input the number. After that the next instruction is ADD which is added to the number stored in the temporary register (the first number). The next instruction is again the LOADTEMP instruction. This instruction again asks the user to enter a number (third number) and followed by LOADTEMP, the next instruction is ADD instruction which adds the second number to the third number. The next instruction is “DISPACC” which stands for “Display Accumulator”, and it displays the value, which is stored in Accumulator, on the output block. The last instruction in the program is HALT which stops the program.

In this way the program runs and adds three numbers together.



* 1. Write a program to subtract a number from another. List your program below (10 marks)  
     The program to subtract a number from another is given below.

LOADTEMP

ADD

LOADTEMP

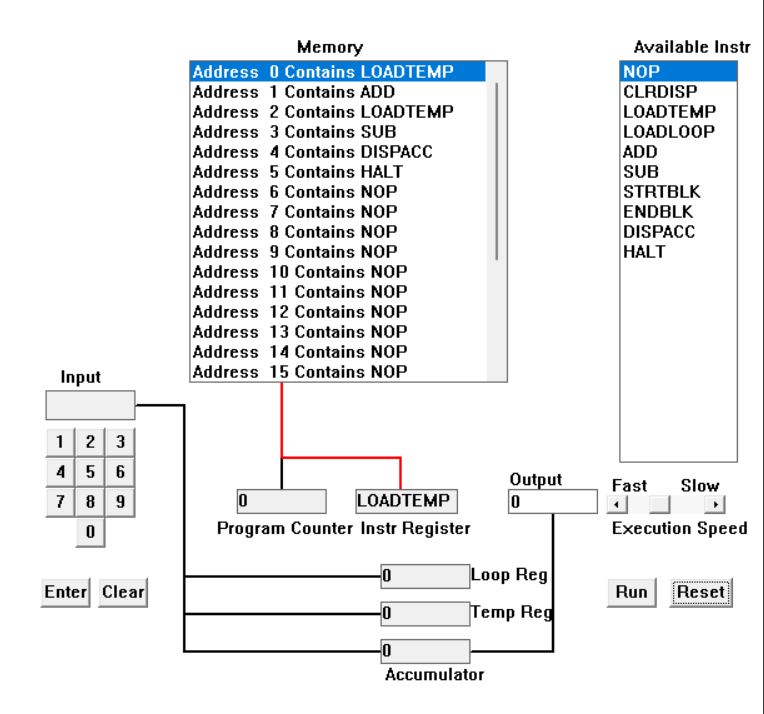
SUB

DISPACC

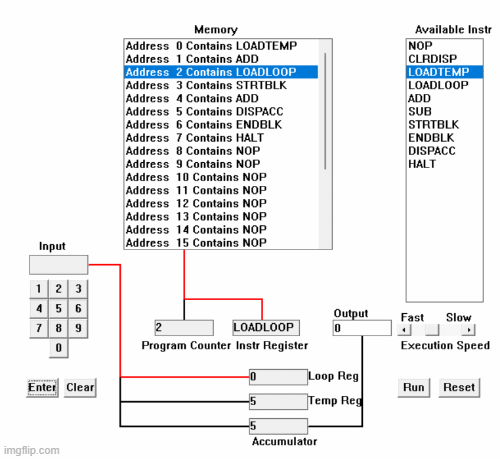
HALT

The program executes in the following ways.

The first instruction of the program is LOADTEMP. The LOADTEMP instruction asks the user to enter a number, which will be stored in the temporary register. The second instruction is ADD which adds the number stored in the temporary register (initially 0). The third instruction in LOADTEMP which asks the user to enter a number. The fourth instruction is SUB instruction. The SUB instruction subtracts the number stored in the accumulator to the number stored inthe temporary register. The fifth instruction is DISPACC. It displays the contents of the accumulator. In this case, the accumulator displays the difference of the second number and first number.



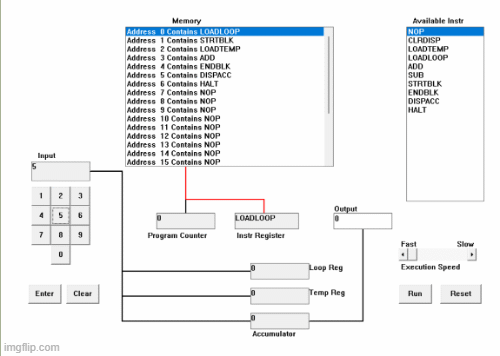
* 1. Load the following program into the simulator:

LOADTEMP  
ADD  
LOADLOOP  
STRTBLK  
ADD  
DISPACC  
ENDBLK  
HALT  
  
Run it and when it reach the LOADTEMP instruction, enter 5 on the keypad and click the “Enter” button. When it reaches the LOADLOOP instruction, enter 6. What do you think the program does? Write your answer below in the form of an equation (10 marks)  


I think the program will add 5 six times as the loop is instructed to run six times. The program, I think, will run in the following ways.

At first, the LOADTEMP instruction asks the user to enter a number and as mentioned in the question above, 5 is entered and the program stores the value in the temporary register and then the ADD instruction adds 0 (zero) to five as zero is first stored in the temporary register. Then LOADLOOP will be executed and it asks the user to enter a number to run the loop desired amount of time, in this case the loop is set to run six times. After that STRTBLK will be executed which denotes the loop has been started and all the operations that should happen are entered inside STRTBLK and ENDBLK. The next instruction is ADD which is inside the loop. The next instruction inside the loop is DISPACC which is the short form of “Display Accumulator” and it does exactly that. It displays the value that is in the accumulator. The next instruction is ENDBLK, this instruction ends the loop when it is supposed to end. The last instruction is HALT which terminates the program.

The program’s result will be 35 as 5 is added 6 times.

* 1. Write a program that will let you add 5, or 10 or 20 numbers together. List your program below and explain how it works (25 marks)  
     

The program is given below.

LOADLOOP

STRTBLK

LOADTEMP

ADD

ENDBLK

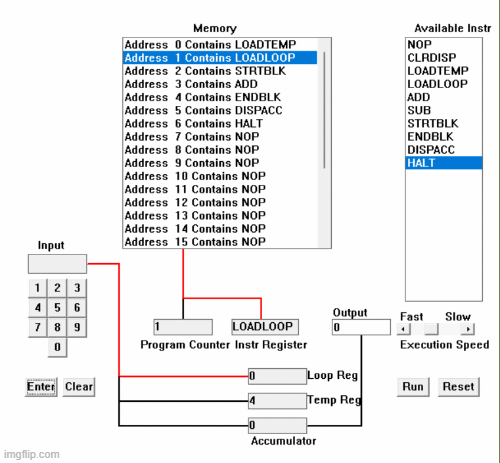
DISPACC

HALT

The following program can add numbers together. In the example, the program adds 5 numbers together.

The program runs in the following way.

First the program executes the LOADLOOP instruction. The LOADLOOP instruction asks the user to enter a number for which how many time the loop should run. In this case, we have to add 5 or 10 or 20 numbers together. So, we can enter 5 to add 5 numbers together. Or we can enter 10 to add 10 numbers together or do the same with 20. Then comes the STRTBLK which denotes the starting of the loop. After that LOADTEMP instruction executes. It asks the user to enter a number. The next instruction that executes inside the loop is ADD instruction. The ADD instruction adds the entered number to the number which is stored in temporary register. After that, ENDBLK instruction executes which denotes the end of the loop block. The next instruction is DISPACC which displays the number stored in the accumulator. The last instruction of the program is HALT. This instruction ends the program.

* 1. Write a program that will let you multiply 2 numbers together. List your program below and explain how it works (35 marks)  
       
     

The program to multiply two numbers is given below.

LOADTEMP

LOADLOOP

STRTBLK

ADD

ENDBLK

DISPACC

HALT

The program runs in the following way.

As there is no multiplication instruction available in the simulator, the only way we can multiply the number together is through loop. First, the LOADTEMP instruction gets executed which asks to enter a number. The second instruction is LOADLOOP which asks the user to enter a number and the loop will run the entered number of times. The next instruction is STRTBLK which indicates the start of the loop. The fifth instruction is ADD which adds the entered number to the zero of the accumulator. The next instruction is ENDBLK which denotes the end of the loop block. After that, the DISPACC will be executed which displays the content of the accumulator. And at last HALT will be executed which stops the program.

Example:

At first, 4 is entered to the temporary register when the LOADTEMP asks for the number. After that, the loop is set to run 5 times as 5 is entered in LOADLOOP instruction. So, the loop will run 5 times. The next instruction ADD adds the entered number 4 to the 0 of the accumulator. Now, the result of 4+0 is stored in the accumulator. Then the loop is set to run for the second time. So 4 is again added to the content of the accumulator, which is 4 (result of 4+0). Now, 8 is stored in the accumulator. The loop now runs for the third time so, 4 is added to 8 (content of the accumulator). Now 12 (result of 4+12) is stored in the accumulator. The program runs like this for the next two times. And the content in the accumulator will be 20 as 4 is added 5 times.

So, the program runs in this way.