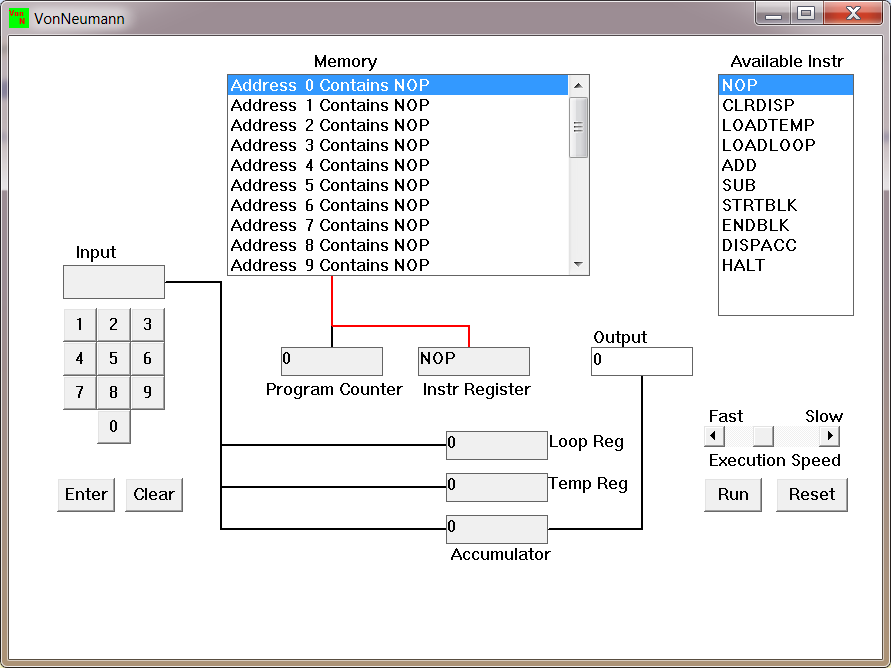
# 4CS015 Fundamentals of Computing – Workshop-6

**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.
  2. 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 than the simulator stops running the program when it executes the instruction “HALT”.
  3. 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)  
     Using the instruction LOADTEMP, the program loads a value from memory into a temporary register. The ADD instruction then adds the temporary register value to the accumulator value and stores the result in the accumulator. The subsequent instruction LOADTEMP loads another value from memory into the temporary register after this initial addition operation is completed once. The value that was loaded in the previous instruction, LOADTEMP, is used to add the value in the temporary register to the value in the accumulator once more using the next instruction, ADD. The accumulator also stores the result of this second addition operation. The accumulator's value, which is the product of the two addition operations, is then displayed using the instruction DISPACC. The instruction HALT concludes the program's execution.Diagram

     Description automatically generated
  4. Write a program to add 3 numbers together. List your program below (10 marks)

LOADTEMP

ADD

LOADTEMP

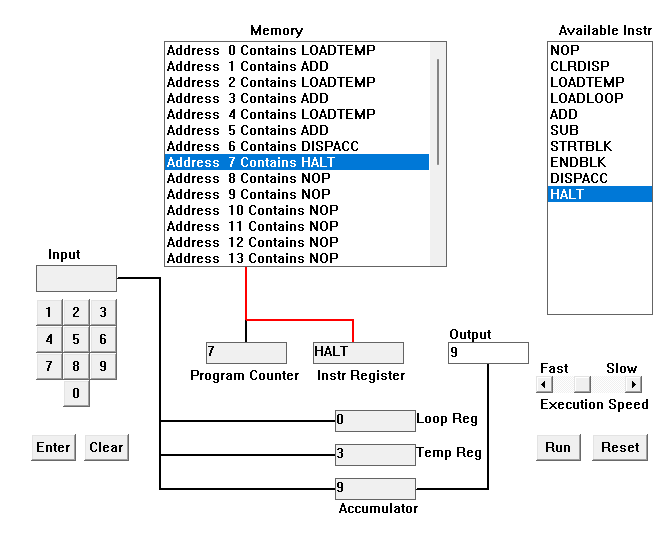
ADD

LOADTEMP

ADD

DISPACC

HALT



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

LOADTEMP

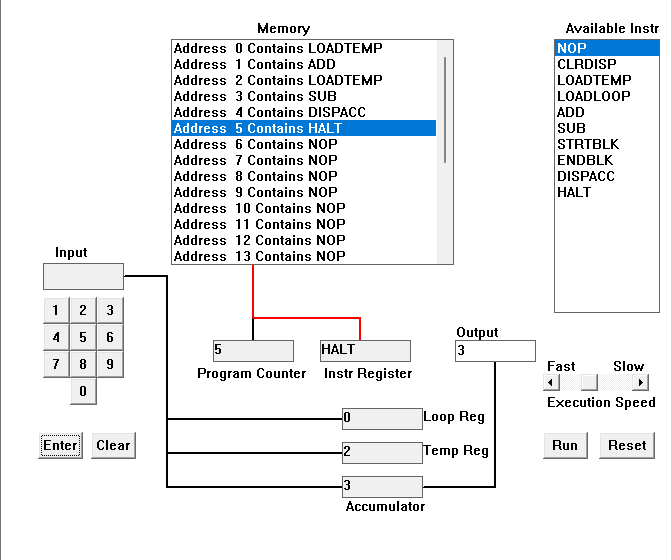
ADD

LOADTEMP

SUB

DISPACC

HALT



* 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)

Diagram

Description automatically generated  
Using the instruction LOADTEMP, the program loads a value from memory into a temporary register. The ADD instruction then adds the temporary register value to the accumulator value and stores the result in the accumulator. After that, the LOADLOOP instruction loads a value into a counter register that will be used to regulate the number of times the loop iterates. The beginning of a code block that will be run multiple times is indicated by the instruction STARTBLK. The instruction ADD in this block adds the value in the accumulator to the value in the temporary register and stores the result in the accumulator. The accumulator's value is then displayed using the DISPACC instruction. The program returns to the LOADLOOP instruction to determine whether the loop should continue. The instruction ENDBLK marks the end of the block of code that will be executed repeatedly. Until the counter register reaches zero, the program will continue to execute the instruction block between STARTBLK and ENDBLK. The instruction HALT concludes the program's execution. Using a loop, this program loads values from memory into a temporary register, adds them to an accumulator, and shows the result. The counter register that was loaded at the beginning of the program with the LOADLOOP instruction controls the number of iterations.

* 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)

LOADLOOP

STARTBLK

LOADTEMP

ADD

DISPACC

ENDBLK

HALT

A value that will be used to regulate the number of iterations in the loop is loaded into a counter register at the beginning of the program. The beginning of a code block that will be run multiple times is indicated by the instruction STARTBLK. After that, the instruction LOADTEMP loads a value from memory into a temporary register, and the instruction ADD adds the temporary register value to the accumulator value and stores the result in the accumulator. The accumulator's value is then displayed using the DISPACC instruction. The program returns to the LOADLOOP instruction to determine whether the loop should continue. The instruction ENDBLK marks the end of the block of code that will be executed repeatedly. Until the counter register reaches zero, the program will continue to execute the instruction block between STARTBLK and ENDBLK. The program's final instruction, HALT, halts its execution. Using a loop, this program loads values from memory into a temporary register, adds them to an accumulator, and shows the result. The counter register that was loaded at the beginning of the program with the LOADLOOP instruction controls the number of iterations.Diagram

Description automatically generated

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

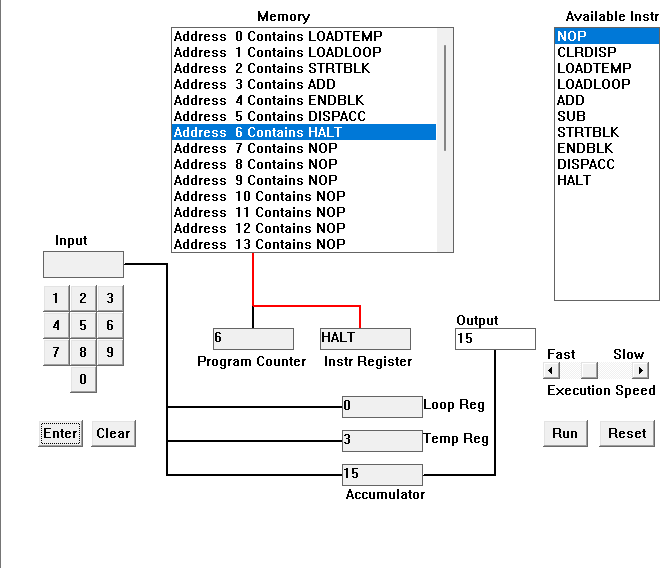
STARTBLK

ADD

ENDBLK

DISPACC

HALT

 Using the instruction LOADTEMP, the program loads a value from memory into a temporary register. Using the instruction LOADLOOP, the program loads a value into a counter register to control the number of times the loop iterates. The beginning of a code block that will be run multiple times is indicated by the instruction STARTBLK. The instruction ADD in this block adds the value in the temporary register to the value in the accumulator (until the multiplication condition is satisfied) and stores the result in the accumulator. The program returns to the LOADLOOP instruction to determine whether the loop should continue. The instruction ENDBLK marks the end of the block of code that will be executed repeatedly. Until the counter register reaches zero, the program will continue to execute the instruction block between STARTBLK and ENDBLK. The value in the accumulator, which is the result of all of the addition operations carried out by the loop, is displayed using the instruction DISPACC following the execution of the loop. The instruction HALT concludes the program's execution. Using a loop, this program loads a value from memory repeatedly, adds it to the accumulator, and displays the result after each iteration.