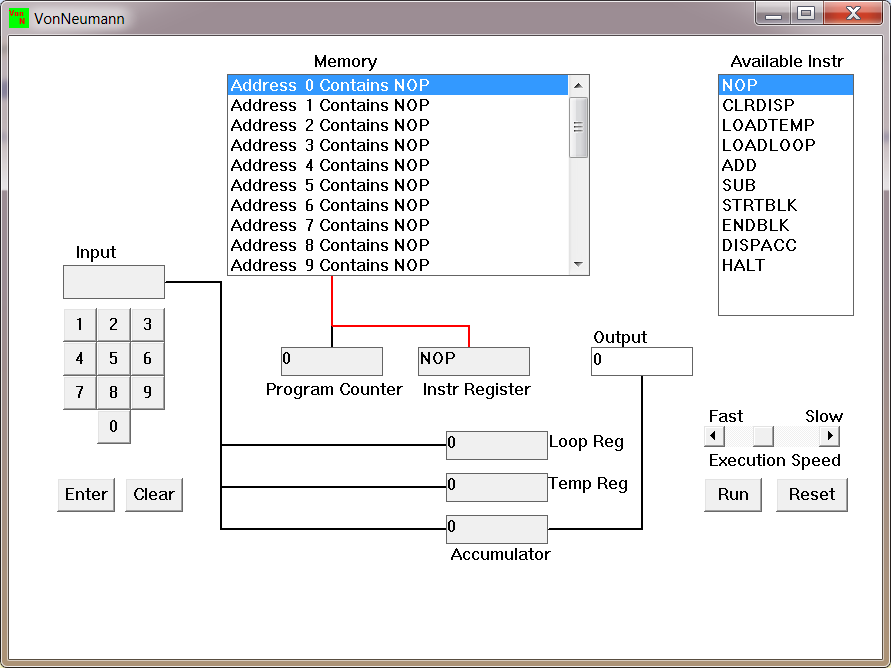
# 4CS015 Fundamentals of Computing – Workshop #6 (Task #3)

This is a marked workshop. It forms the second part of your portfolio. You will need to complete the workshop and then submit a copy of this document with a title that follows the following format (“DENNETT\_1234567\_wsp6.docx”), via CANVAS, by the deadline. The Von Neumann Simulator can be found on canvas

**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”.

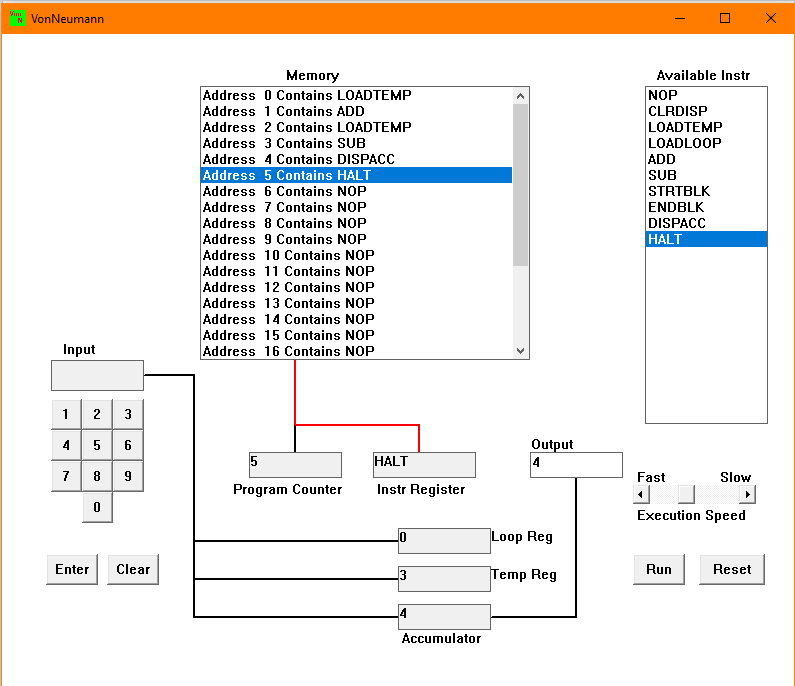
**NOTE: Remember to Write down your code along with the screenshots of your code**

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

LOADTEMP🡪7  
ADD  
LOADTEMP🡪3  
SUB  
DISPACC🡪4  
HALT

LOADTEMP gets the number put in the Input and loads it into the Temporary Register. ADD then adds the number of the Input of Temporary Register to the zero in the Accumulator. LOADTEMP again takes another number from the Input and loads it into the Temporary Register replacing the earlier input. SUB then subtracts the number in the Temporary Register with the number in the Accumulator. DISPACC then displays the Accumulator value in the Output. And with HALT the program stops.

Initial value of Accumulator = 0  
7 + 0 = 7  
7 - 3 = 4  
Output = 4

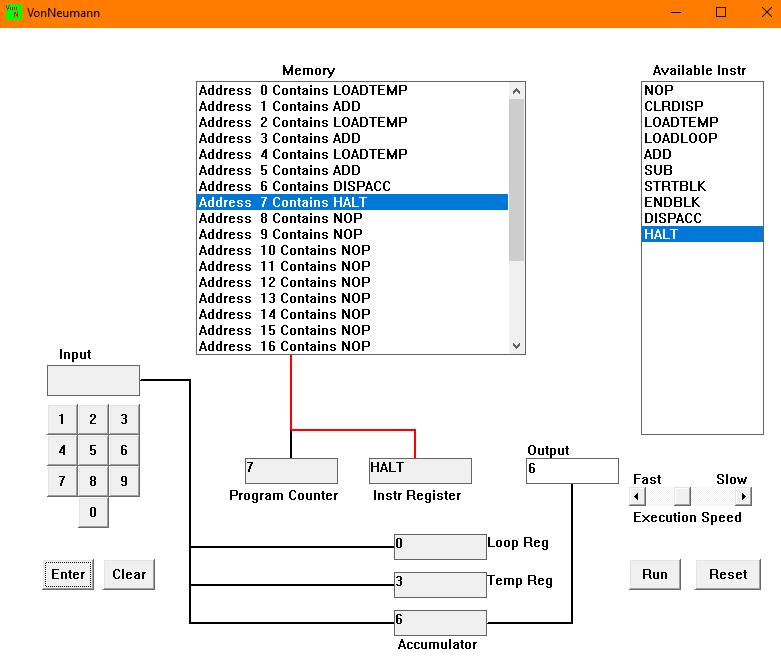


* 1. Write a program to add 3 numbers together. List your program below and explain the code (10 marks)

LOADTEMP🡪1  
ADD  
LOADTEMP🡪2  
ADD  
LOADTEMP🡪3  
ADD  
DISPACC🡪6  
HALT

LOADTEMP gets the number put in the Input and loads it into the Temporary Register. ADD then adds the number of the Input of Temporary Register to the zero in the Accumulator. LOADTEMP again takes another number from the Input and loads it into the Temporary Register replacing the earlier input. ADD then again adds the number in the Temporary Register with the number in the Accumulator. Finally, LOADTEMP again takes another number from the Input and loads it into the Temporary Register replacing the earlier input. And the final ADD adds the number in the Temporary Register with the Accumulator. DISPACC then displays the Accumulator value in the Output. And with HALT the program stops.

Initial value of Accumulator = 0  
0 + 1 = 1  
1 + 2 = 3  
3 + 3 = 6  
Output = 6

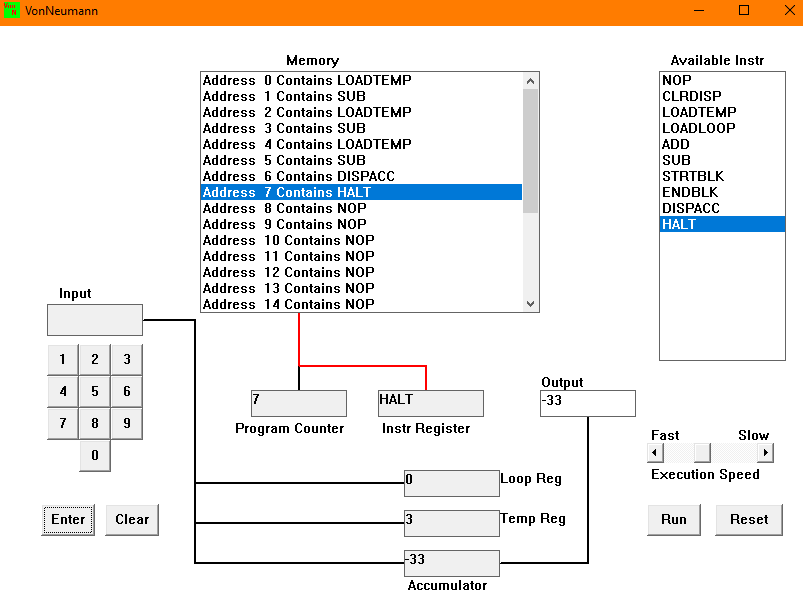


* 1. Write a program to simplify: – 25 - (5 + 3). List your program below. (10 marks)

LOADTEMP🡪25  
SUB   
LOADTEMP🡪5  
SUB  
LOADTEMP🡪3  
SUB  
DISPACC🡪-33  
HALT

LOADTEMP gets the number put in the Input and loads it into the Temporary Register. SUB then subtracts the number of the Input of Temporary Register with that to the zero in the Accumulator. LOADTEMP again takes another number from the Input and loads it into the Temporary Register replacing the earlier input. SUB then again subtracts the number in the Temporary Register with the number in the Accumulator. Finally, LOADTEMP again takes another number from the Input and loads it into the Temporary Register replacing the earlier input. And the final SUB subtracts the number in the Temporary Register with the Accumulator. DISPACC then displays the Accumulator value in the Output. And with HALT the program stops.

Initial value of Accumulator = 0  
0 - 25 = -25  
-25 - 5 = -30  
-30 - 3 = -33  
Output = -33



* 1. Load the following program into the simulator:  
     LOADTEMP  
     ADD  
     LOADLOOP  
     STRTBLK  
     ADD  
     DISPACC  
     ENDBLK  
     HALT  
     Run it and when it reaches 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)

LOADTEMP🡪5  
ADD  
LOADLOOP🡪6  
STRTBLK  
ADD  
DISPACC🡪35  
ENDBLK  
HALT

LOADTEMP gets the number put in the Input and loads it into the Temporary Register. ADD then adds the number of the Input of Temporary Register to the zero in the Accumulator. LOADLOOP gets the number put in the Input and loads it into the Loop Register. STRTBLK starts the loop. ADD instruction inside of the loop keeps on adding the initial Input with the Accumulator until the Loop Register value drops to zero. ENDBLCK then ends the loop block once the Loop Register is zero. DISPACC then displays the Accumulator value in the Output. And with HALT the program stops.

Initial value of Accumulator = 0

0 + 5 = 5

Loop Starts (6 times):

5 + 5 = 10

10 + 5 = 15

15 + 5 = 20

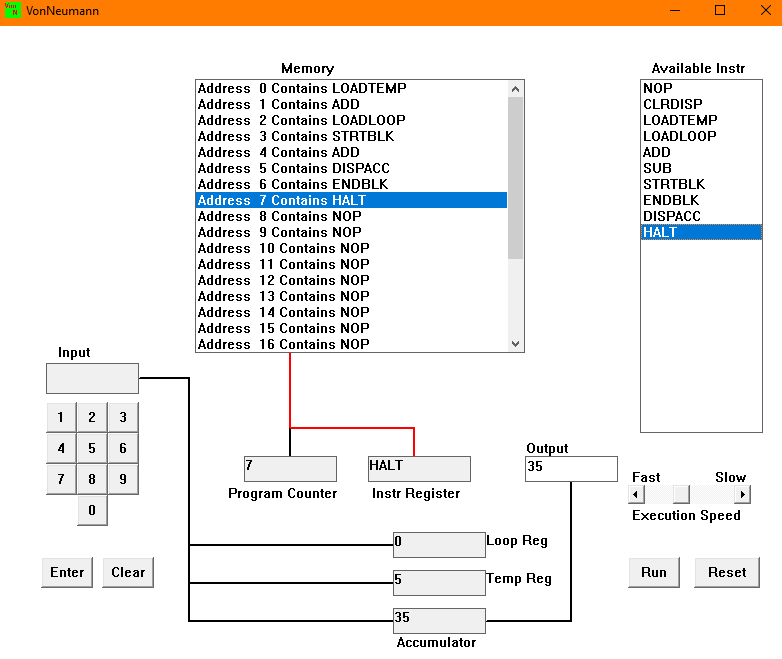
20 + 5 = 25

25 + 5 = 30

30 + 5 = 35

Loop Ends!

Output = 35

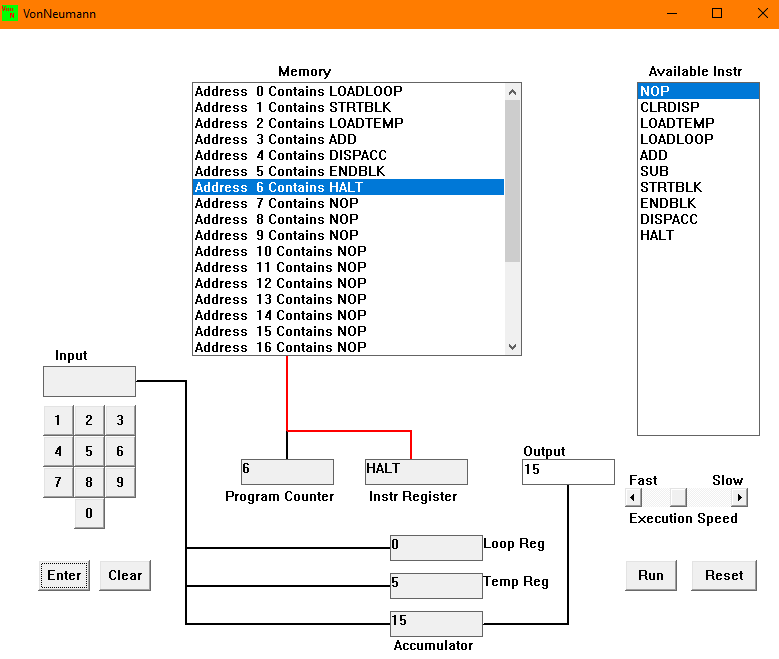


* 1. Write a program that will let you add 5, or 10 or 20 numbers together. List your program below and explain how it works (25 marks)

LOADLOOP🡪5  
STRTBLK  
LOADTEMP🡪1, 2, 3, 4, 5  
ADD  
DISPACC🡪15  
ENDBLK  
HALT

LOADLOOP gets the number put in the Input and loads it into the Loop Register. STRTBLK starts the loop. LOADTEMP gets the number put in the Input and loads it into the Temporary Register. ADD then adds the number of the Input of Temporary Register to the zero in the Accumulator. DISPACC then displays the Accumulator value in the Output. ENDBLCK then ends the loop block. The input in the Loop Register decreases by one and the loop re-iterates till the Loop Register’s value becomes zero. ENDBLCK then ends the loop once the Loop Register is zero. And with HALT the program stops.

Number of times to loop = 5  
Initial value of Accumulator = 0  
Loop Starts:  
0 + 1 = 1  
1 + 2 = 3  
3 + 3 = 6  
6 + 4 = 10  
10 + 5 = 15   
Loop Ends!  
Output = 15



* 1. Write a program that will let you multiply 3 numbers together. List your program below and explain how it works. (35 marks)

**E.g.: (4×6×7)**  
LOADLOOP🡪4  
LOADTEMP🡪6  
STRTBLK  
ADD  
ENDBLK  
DISPACC  
LOADTEMP🡪24 (As 4 \* 6 = 24)  
LOADLOOP🡪7  
STRTBLK  
ADD  
ENDBLK  
SUB  
DISPACC🡪168  
HALT

LOADLOOP gets the number put in the Input and loads it into the Loop Register. LOADTEMP gets the number put in the Input and loads it into the Temporary Register. STRTBLK starts the loop. ADD then adds the number of the Input of Temporary Register to the zero in the Accumulator. ADD instruction inside of the loop keeps on adding the initial Input with the Accumulator until the Loop Register value drops to zero. ENDBLCK then ends the loop block once the Loop Register is zero. ENDBLCK then ends the loop block. DISPACC then displays the Accumulator value in the Output. This Output is the multiplication of two Inputs. Now, LOADTEMP gets another number put in the Input and loads it into the Temporary Register. Again, LOADLOOP gets the number put in the Input and loads it into the Loop Register and the ADD instruction keeps on adding the initial Input with the Accumulator until the Loop Register value drops to zero then once it reaches zero ENDBLK instruction is run and the loop is ended. SUB then subtracts the third input from the Accumulator. Finally, DISPACC displays the Output and HALT ends the program.

Number of times to loop = 4  
Initial value of Accumulator = 0  
Input = 6  
Loop Starts:  
0 + 6 = 6  
6 + 6 = 12  
12 + 6 = 18  
18 + 6 = 24   
Loop Ends!  
Output = 24 (This is the multiplication of first two inputs)

Input = 24 (As 4 \* 6 = 24; Because the Accumulator already has 24 stored, we need to subtract it from the final output)  
Number of times to loop = 7  
Value of Accumulator = 24  
Loop Starts:  
24 + 24 = 48  
48 + 24 = 72  
72 + 24 = 96  
96 + 24 = 120   
120 + 24 = 144  
144 + 24 = 168  
168 + 24 = 192  
Loop Ends!  
SUB = 192 – 24 = 168  
Output = 168   
  
