*Fill in the following tables. You should be placing an answer in any non-blacked-out box!*

**Addition Program:**This program should load a variable X, and then add the number 5 to it, placing the result into a variable Y.

|  |  |  |  |
| --- | --- | --- | --- |
| **Line** | **Instruction** | **Value** | **Comment** |
| 0 | LOD |  | We want to load the value of variable X into the ALU. |
| 1 |  | #5 | We want to add the number 5 to the current value in the ALU, placing the result in the accumulator. |
| 2 |  |  | We want to store the result into the variable Y. |
| 3 | HLT |  |  |

Now modify the program so it SUBTRACTS 5 instead of adding it without using a negative number.

|  |  |  |
| --- | --- | --- |
| **Line** | **Instruction** | **Value** |
| 0 |  |  |
| 1 |  |  |
| 2 |  |  |
| 3 | HLT |  |

Now modify the program so it adds the variable Z to X, instead of a predefined number.

|  |  |  |
| --- | --- | --- |
| **Line** | **Instruction** | **Value** |
| 0 |  |  |
| 1 |  |  |
| 2 |  |  |
| 3 | HLT |  |

(Extra lines if you need them)

|  |  |  |  |
| --- | --- | --- | --- |
| **Line** | **Instruction** | **Value** | **Comment** |
| 0 |  |  |  |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |
| 5 |  |  |  |
| 6 |  |  |  |
| 7 |  |  |  |
| 8 |  |  |  |
| 9 |  |  |  |
| 10 |  |  |  |
| 11 |  |  |  |
| 12 |  |  |  |
| 13 |  |  |  |
| 14 |  |  |  |

**Looping Program**

This program should repeatedly add the variable Y to the variable X, and then place that result in variable X. For example if X = 1 and Y = 2, then on each **loop** of the program the value of X would be:

Loop 1: X = 1, Loop 2: X = 3, Loop 3: X = 5

|  |  |  |  |
| --- | --- | --- | --- |
| **Line** | **Instruction** | **Value** | **Comment** |
| 0 | LOD |  | We want to load the value of variable X into the ALU. |
| 1 |  |  | We want to add the variable Y to the current value in the ALU, placing the result in the accumulator. |
| 2 |  |  |  |
| 3 |  | 0 | We want to jump back to line 0. |

Now modify the program so that it records the number of times the program looped (i.e how many times the JMP instruction was run) (hint: store the number of times the program looped into a new variable, perhaps Z?):

|  |  |  |  |
| --- | --- | --- | --- |
| **Line** | **Instruction** | **Value** | **Comment** |
| 0 |  |  |  |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |
| 5 |  |  |  |
| 6 |  |  |  |
| 7 |  |  |  |
| 8 |  |  |  |
| 9 |  |  |  |
| 10 |  |  |  |
| 11 |  |  |  |

**Branching Program:**

Write a program which will place a 1 into variable Z if the variables X and Y are equal, otherwise place a 0 in Z. This is essentially building an “equals(x, y) -> 1 | 0” function. (Hint – you can check if X and Y are equal by checking if X – Y = 0)

|  |  |  |  |
| --- | --- | --- | --- |
| **Line** | **Instruction** | **Value** | **Comment** |
| 0 | LOD | X |  |
| 1 |  |  |  |
| 2 | JMZ |  |  |
| 3 | LOD | #0 |  |
| 4 |  |  |  |
| 5 |  |  |  |
| 6 |  | #1 |  |
| 7 |  | Z |  |
| 8 | HLT |  |  |

(Additional lines if you need them)

|  |  |  |  |
| --- | --- | --- | --- |
| **Line** | **Instruction** | **Value** | **Comment** |
| 0 |  |  |  |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |
| 5 |  |  |  |
| 6 |  |  |  |
| 7 |  |  |  |
| 8 |  |  |  |
| 9 |  |  |  |
| 10 |  |  |  |
| 11 |  |  |  |
| 12 |  |  |  |
| 13 |  |  |  |
| 14 |  |  |  |
| 15 |  |  |  |
| 16 |  |  |  |
| 17 |  |  |  |
| 18 |  |  |  |

**Multiplication Program**

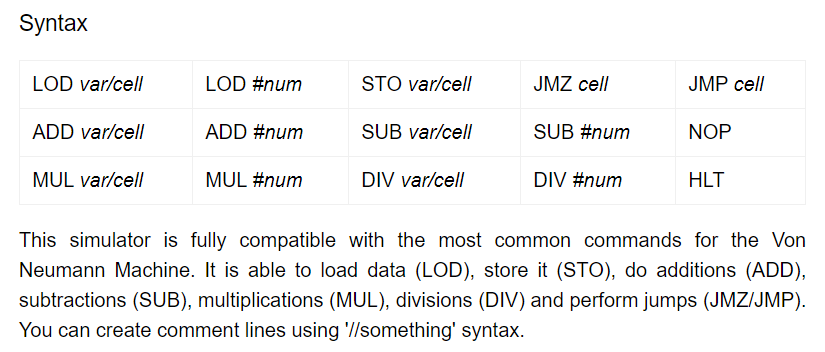
Without using the MUL instruction, create a program which multiplies the values in variables X and Y together, placing the result in variable Z.

|  |  |  |  |
| --- | --- | --- | --- |
| **Line** | **Instruction** | **Value** | **Comment** |
| 0 | LOD |  | Z = Z + X |
| 1 |  |  |
| 2 | STO |  |
| 3 |  |  | Y = Y - 1 |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  | IF Y == 0 THEN EXIT, OTHERWISE JUMP TO 0 |
| 7 |  |  |
| 8 |  |  |
| 9 | HLT |  |  |

(Additional lines if you need them)

|  |  |  |  |
| --- | --- | --- | --- |
| **Line** | **Instruction** | **Value** | **Comment** |
| 0 |  |  |  |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |
| 5 |  |  |  |
| 6 |  |  |  |
| 7 |  |  |  |
| 8 |  |  |  |
| 9 |  |  |  |

**Language Reference**

****

|  |  |  |
| --- | --- | --- |
| **LOAD** | <variable> | Load data into ALU |
| **STO** | <variable> | Store the value in the accumulator into the given variable. |
| **ADD** | <variable> OR #<number> | Add a variable or number to the current value in the ALU and place it in the accumulator. |
| **SUB** | <variable> OR #<number> | Subtract a variable or number to the current value in the ALU and place it in the accumulator. |
| **JMP** | <line> | Jump to the given line number. |
| **JMZ** | <line> | If the accumulator contains the value 0, THEN AND ONLY THEN jump to the given line. If it does not contain 0 then continue onto the next line (default behaviour) |
| **HLT** |  | Stop the program. |