Ejercicio 3

Objetivo

Identificar las características de la organización y arquitectura del microprocesador de una computadora de propósito general analizando sus recursos de hardware y software para conocer capacidades y limitaciones de forma organizada y responsable.

Desarrollo

Haga uso del simulador MARIE para implementar los siguientes programas.

1. Escriba un código que solicite al usuario tres números y despliegue su suma.
2. Escriba un código que solicite al usuario dos números y despliegue el mayor.
3. Escriba un código que solicite al usuario dos números. Si el primero es menor a 5, sumar ambos números, de lo contrario restarlos. Desplegar el resultado.
4. Describa con sus propias palabras de qué manera se realiza la multiplicación de dos números signados en el código ejemplo del **Listado 1**, tomado de https://marie-js.github.io/MARIE.js/. Describa cada línea de código.

Listado 1. Multiplicación de dos números con signo

```
/ Multiplication Calculator
/ by the MARIE.js Team
/ Copyright (C) 2016. Licensed under the MIT License

/ Prompt user to type in integers
Clear
Store result
Input
Store X
Input
Store Y

/ check if Y is negative, if -ve negate Y and set negative flag
Load Y
Skipcond 000
Jump nonneg

Subt Y
Subt Y
Store Y
Clear
Add one
Store negflag
Clear
Jump loop

nonneg,  Clear
         Store negflag
         / check if Y is zero, if it is, then we jump to halt
         Load Y
```

```
         Skipcond 400
         Jump loop / false
         Jump halt / true

/ Loop for performing iterative addition
loop,    Load result
         Add X
         Store result

         Load Y
         Subt one
         Store Y

         Skipcond 400 / have we completed the multiplication?
         Jump loop / no; repeat loop
         / yes, so exit the loop

/ check for negative flag, if it is set, negate the result
Load negflag
Skipcond 800
Jump halt

/ negate result
Load result
Subt result
Subt result
Store result
/ run the next three instructions, which halts the program

/ Output result to user then halt program
halt,    Load result
         Output
         Halt

X, DEC 0
Y, DEC 0
one, DEC 1
negflag, DEC 0
result, DEC 0
```