Tarea N°2 – Scripts LMC

UNIVERSIDAD EL BOSQUE
FACULTAD DE INGENIERÍA
PROGRAMA – INGENIERÍA DE SISTEMAS
DOCENTE – GERMAN ENRIQUE CAMPOS HERNANDEZ

Federico Vargas Rozo

MARZO 2025

00 – Suma de Dos Números

Este código solicita dos números de entrada, los suma y muestra resultado. Solicita el primer número, lo guarda en una variable, solicita otro número, suma el valor de la variable guardada, muestra el resultado y detiene la ejecución.

Peter L. Higginson: https://peterhigginson.co.uk/lmc/

01 – Máximo entre Dos Números

Este código compara dos números ingresados por el usuario y muestra el mayor de los dos. Si *num2* es mayor o igual a *num1*, muestra *num2*, de lo contrario, muestra *num1*. Finalmente, el programa se detiene.

101Computing.net: https://www.101computing.net/lmc/

02 – Sumas y Restas con Tres Números

Este código recibe tres números como entrada, muestra la suma de los dos primeros números ingresados, después, muestra la resta del primer número al tercer número. Finalmente, el programa se detiene.

Peter L. Higginson: https://peterhigginson.co.uk/lmc/

03 – Contador

Este código realiza un bucle para contar hacia atrás a partir de un número ingresado por el usuario. El bucle muestra el valor del contador, decrementa el contador en 1, y repite el proceso mientras el contador sea positivo o cero.

101Computing.net: https://www.101computing.net/lmc/

04 – Caracteres ASCII

Este código imprime los caracteres ASCII básicos, comenzando desde el espacio (ASCII 32) hasta el carácter con código ASCII 126 (~). Utilizando la instrucción OTC (Output as Character) se puede utilizar el valor del acumulador para imprimir como un carácter, haciendo uso de la tabla ASCII. Debido a que los Caracteres ASCII Imprimibles comienzan desde el 32 (Espacio), hasta el 126 (~), este código imprimirá en orden. Los caracteres previos a 32, conocidos como Caracteres ASCII de Control, y los caracteres posteriores a 126, conocidos como Caracteres ASCII Extendido no son impresos, pues la variable que indica el máximo valor para imprimir, *max*, llega únicamente hasta 127.

Tabla ASCII: https://elcodigoascii.com.ar

Peter L. Higginson: https://peterhigginson.co.uk/lmc/

05 – Caracteres ASCII en formato de Tabla

Este código realiza fundamentalmente lo mismo que el código anterior, imprimir ciertos caracteres ASCII, sin embargo los imprime con su código ASCII antes.

Tabla ASCII: https://elcodigoascii.com.ar

Peter L. Higginson: https://peterhigginson.co.uk/lmc/

06 – Multiplicación de Dos Números

Este código realiza la multiplicación de dos números ingresados por el usuario utilizando un enfoque de suma repetitiva, pues en el LMC no existe el operador de multiplicación. Realiza una serie de sumas que almacenan el resultado y utiliza la variable del segundo número ingresado como un contador, para seguir realizando sumas hasta que se cumpla que el contador es negativo, posterior a esto se resta únicamente una vez el primer número ingresado para quitar la última suma realizada.

101Computing.net: https://www.101computing.net/lmc/

07 – Números Triangulares

Este código realiza una secuencia de números triangulares, donde cada número es la suma acumulativa de los valores de *counter*. El bucle se detiene cuando *counter* supera 10, ya que *ten* está definido como 10.

101Computing.net: https://www.101computing.net/lmc/

08 – Factorial de un Número

Este código realiza el factorial de un número. Nuevamente, como el LMC no tiene la capacidad de multiplicar, toca “anidar” el proceso de multiplicación. El programa solicita un número, si este es 0, muestra 1, de lo contrario, calcula el factorial utilizando un enfoque de multiplicación repetitiva. Finalmente, muestra el resultado final y se detiene.

101Computing.net: https://www.101computing.net/lmc/

09 – Valores Inmediatos

Este código realiza una operación simple de suma y resta con valores inmediatos (constantes) especificadas con la etiqueta #, seguido de un número literal. Este código puede no funcionar en todos los simuladores LMC ya que algunos no soportan esta funcionalidad.

101Computing.net: https://www.101computing.net/lmc/

10 – Direccionamiento Indirecto

Este código toma un número ingresado por el usuario y lo almacena en una secuencia de posiciones de memoria, comenzando desde la dirección 99, valor inicial de la variable *address*, y decrementando hacia atrás. El uso de *STA @address* implica que *address* actúa como un puntero a la dirección de memoria donde se almacenará el valor. Esto es una extensión del LMC estándar, ya que el LMC tradicional no admite direccionamiento indirecto. Por lo tanto, este código, al igual que el anterior, posiblemente no corra en ciertos simuladores LMC.

101Computing.net: https://www.101computing.net/lmc/

11 – Patrón de 0s y 1s

Este código imprime alternadamente los valores 1 y 0 un número específico de veces, especificado por la variable *count*. En cada iteración el programa realizará la muestra del 1, la muestra del 0 y decrementar el *count*. Cuando *count* sea negativo, el ciclo se detiene.

Wellingborough School: https://wellingborough.github.io/LMC/LMC0.3.html

12 – Cuadrado de un Número

Este código realiza el cuadrado de un número ingresado por el usuario. El programa calcula la suma acumulativa de un número consigo mismo, lo que equivale a multiplicar el número por sí mismo (es decir, n x n, es decir n^2). Si ingreso 3, el programa hará 3 + 3 + 3 = 9.

Wellingborough School: https://wellingborough.github.io/LMC/LMC0.3.html

13 – División Entera de dos Números

Este código realiza la division entera de dos números ingresados por el usuario. El programa divide el primer número *dividend* por el segundo número *divisor* y muestra el cociente *answer*. El programa solicita dos números al usuario: el dividendo y el divisor. Luego, entra en bucle, donde resta el divisor al dividendo, incrementa el cociente *answer* en 1, y repite el proceso mientras el dividendo sea mayor o igual que el divisor. Cuando el dividendo es menor que el divisor, el programa muestra el cociente y se detiene. Como es la división entera, al dividir un número y el resultado no es exacto, es decir, tiene posiciones decimales, solo mostrará la parte entera. Ejemplo, 3 / 2 = 1 (pero realmente es 1.5).

Wellingborough School: https://wellingborough.github.io/LMC/LMC0.3.html