# UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

## FACULTAD DE INGENIERÍA


**ESTRUCTURA DE DATOS Y ALGORITMOS I (1227)**

*Profesor:* ***M.I. Marco Antonio Martínez Quintana***

*Semestre 2021-2*

**Actividad Asíncrona #03 Lunes 21 de Junio**

**Nombre del alumno**: Cadena Luna Iván Adrián

**Grupo:** 15

**Fecha:** (28/06/2021)

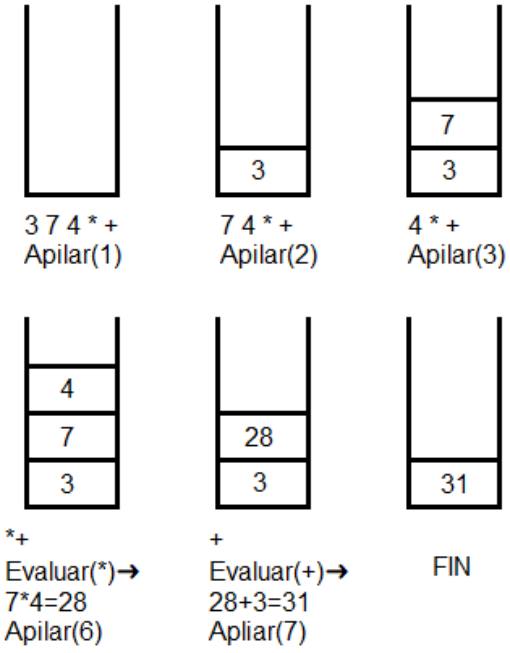
- [**AVANCE**] Desarrollar y probar un algoritmo que pueda resolver operaciones introducidas en la terminal como una cadena continua.

Para implementar la calculadora de números enteros en una cadena continua, el usuario dispondrá de una pila donde almacenar los operandos y los resultados de las operaciones parciales, conforme estas se realizan.

Además, debemos permitir que el usuario disponga de una serie de memorias de los resultados, donde podrá almacenar, en cada momento, el contenido de la cabeza de la pila de operandos (una cola). El número de memorias almacenadas podrá ir variando a lo largo de la ejecución de la calculadora.

**3+7*4**

Para cada símbolo que se analiza se muestra la acción a realizar como el estado de la pila. La evaluación procede de la siguiente manera: el 3, el 7 y el4 son apilados, en ese orden en la pila. Para procesar el *, se desapilan los dos elementos superiores de la pila: esto es, el 4 y después el 7. El primer elemento desapilado se convierte en el operando derecho del operador, y el segundo elemento desapilado en el operando izquierdo: por tanto, los parámetros se obtienen de la pila en orden inverso al natural. Para la multiplicación, esto no importa, pero para la resta y la división, desde luego que sí. El resultado de la multiplicación es 28, que es apilado en la pila. En este momento, la cima de la pila es un 28, y debajo hay un 3. Para procesar el +, se desapilan el 28 y el 3, y su suma, 31, se apila. En este punto, la expresión se ha leído completamente y la pila solo tiene un elemento. Por tanto, la respuesta final a la evaluación de la expresión es 31. Como hay 3 operandos y 2 operadores, habrá 5 pasos y 5 apilamientos para evaluar la expresión.



**Algoritmo de la función “push”:**

Creamos un nodo para el valor que colocaremos en la pila.

Hacemos que nodo->siguiente apunte a Pila.

Hacemos que Pila apunte a nodo.

**Algoritmo de la función “pop”:**

Hacemos que nodo apunte al primer elemento de la pila, es decir a Pila.

Asignamos a Pila la dirección del segundo nodo de la pila: Pila->siguiente.

Guardamos el contenido del nodo para devolverlo como retorno (puesto que la operación pop equivale sólo a leer y borrar).

Liberamos la memoria asignada al primer nodo, el que queremos eliminar.