ALGORITMOS Y ESTRUCTURAS DE DATOS

Trabajo Práctico no. 5 **Fecha: 20/04/23**

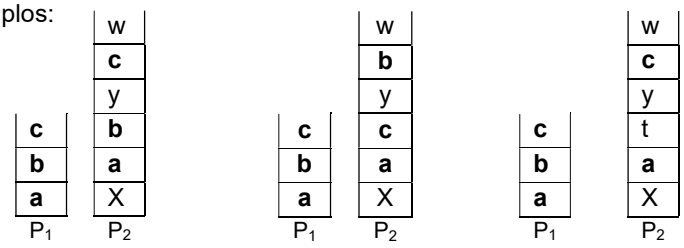
Tema: **Tipo de datos PILA**

1. Considere el ADT PILA(item) visto en clase:

   a. Agregue a la ***especificación algebraica*** del mismo las siguientes operaciones:
      i. Una selectora: FONDO, que devuelve el item que se encuentra en el fondo de la pila.
      ii. Una modificadora: POPFONDO, que elimina el ítem que se encuentra en el fondo de la pila.
      iii. Un test: ESSIMETRICA, que retorna true si la pila es simétrica, caso contrario retorna false.

   b. Como ***usuario*** del ADT PILA diseñe las siguientes funciones:
      i. **invertirLista**, que recibe una lista enlazada y, con la ayuda de una pila auxiliar, retorna la lista construida en orden inverso.
      ii. **incluida**, que dadas dos pilas retorna true si los elementos de la primera pila se encuentran contenidos en la segunda pila, respetando el orden relativo de los mismos.

      Ejemplos:



   c. Codifique en un archivo de nombre **Pila.h** una implementación en lenguaje C del ADT PILA de enteros con lista enlazada utilizando la tipificación vista en la clase práctica. Escriba un programa para probar todas las operaciones de la Pila y las funciones **invertirLista** e **incluida** del apartado b). Calcule la complejidad de las operaciones en notación O Grande

2. Teniendo en cuenta las operaciones del ADT PILA(item): PV (Pilavacia), Push, PushFondo (PushF), Pop, PopFondo (PopF), Top y Fondo indique en función de las constructoras primitivas cuál es la pila resultante en cada caso:

   a. **PushF**(**Push** (**PV**,c), **Fondo**(**Push**(**PopF**(**PushF**(**Push** (**PV**,e), f)),g)))

   b. **Push**(**Push**(**Pop**(**Push**(**PushF**(**Push**(**PV**, a), b), c)),a) , **Top**(**Push**F(**Push** (**PV**, J), k)))

3. **Como usuario del ADT PILA** escriba el algoritmo para **CONVERTIR** una expresión aritmética dada en notación infija a una expresión en notación posfija*****. El proceso de convertir acepta una expresión infija como entrada y produce una expresión posfija como salida. Considere expresiones bien formadas que tengan variables (de la 'a' a la 'z'), operadores binarios (+, -, *, /), el operador unario (~) y paréntesis terminadas por la marca final '='. La idea es utilizar una pila para almacenar los operadores a medida que son encontrados para más tarde desapilar estos operadores de acuerdo a su precedencia.

Considere los siguientes ***casos de prueba***:

forma infija	forma posfija
a+b	a b +
~a+b	a ~ b+
a+b*a	a b a * +
(a+ (~b))*c	a b ~ + c *
a*(b+c)+a/e	a b c + * a e / +
a+b-c	a b + c -

forma infija	forma posfija
(a+(~b-c))	a b ~ c - +
(a+b)-c	a b + c -
(a-(b+c))	a b c + -
(b-a)/(c+d)	b a - c d + /
a+b/(d-a)*e	a b d a - / e * +
a*b/c	ab*c/