

Estrategias de Programación y Estructuras de Datos

Tema 5: Pilas y Colas

Ejercicios propuestos

1. Calcular el coste de todas las operaciones públicas de `Stack<E>` y `Queue<E>`.

2. Programar tres versiones de un método:

```
StackIF<E> invierte(StackIF<E> s)
```

que invierta la pila dada por parámetro. Dicho método deberá realizarse fuera de la clase `Stack<E>`.

- a) Primera versión: utilizando iteradores.
- b) Segunda versión: sin utilizar iteradores y de forma iterativa.
- c) Tercera versión: sin utilizar iteradores y de forma recursiva.

Compare el coste asintótico temporal en el caso peor de las implementaciones.

3. Realice el ejercicio anterior sobre colas.

4. Programar un método `rotateQ` que rote una cola a derecha e izquierda:

- El método recibe un entero (sin restricciones).
- Si el entero es positivo, la cola se rotará hacia la izquierda y si es negativo, hacia la derecha. A continuación podemos ver un ejemplo:

- Cola original: primero → 1 , 2 , 3 , 4 , 5 ← último
- Cola tras `rotateQ(4)`: primero → 5 , 1 , 2 , 3 , 4 ← último
- Cola tras `rotateQ(-2)`: primero → 4 , 5 , 1 , 2 , 3 ← último

- a) Realice el método de manera externa a la clase `Queue<E>`, de forma que devuelva una cola con el resultado pedido.
- b) Añada el método a la clase `Queue<E>`, de forma que modifique la cola llamante.

Calcule el coste asintótico temporal en el caso peor de ambas versiones y compárelo. Procure que el coste real (no asintótico) sea el menor posible.

5. Programar un método `rotateS` que rote una pila hacia arriba y hacia abajo:

- El método recibe un entero (sin restricciones).
- Si el entero es positivo, la pila se rotará hacia arriba y si es negativo, hacia abajo. A continuación podemos ver un ejemplo:

- Pila original: cima → 1 , 2 , 3 , 4 , 5
- Pila tras `rotateS(3)`: cima → 4 , 5 , 1 , 2 , 3
- Pila tras `rotateS(-1)`: cima → 5 , 1 , 2 , 3 , 4

- a) Realice el método de manera externa a la clase `Stack<E>`, de forma que devuelva una pila con el resultado pedido.
- b) Añada el método a la clase `Stack<E>`, de forma que modifique la pila llamante.

Calcule el coste asintótico temporal en el caso peor de ambas versiones y compárelo. Procure que el coste real (no asintótico) sea el menor posible.

6. Implementar una clase `StackMS<E>` que extienda la clase `SequenceMS<E>` (ver ejercicio 9 del tema 4) y que implemente el interfaz `StackMSIF<E>` (ver ejercicio 4 del tema 1). Compare el coste asintótico temporal en el caso peor de todos sus métodos públicos con los correspondientes de `Stack<E>`.
7. Implementar una clase `QueueMS<E>` que extienda la clase `SequenceMS<E>` (ver ejercicio 9 del tema 4) y que implemente el interfaz `QueueMSIF<E>` (ver ejercicio 4 del tema 1). Compare el coste asintótico temporal en el caso peor de todos sus métodos públicos con los correspondientes de `Queue<E>`.
8. Enriquezca las clases `List<E>`, `Stack<E>` y `Queue<E>` con constructores por copia que permitan realizar conversiones entre esos tipos de datos. Por ejemplo, en la clase `List<E>` se debería poder construir una lista con la misma secuencia que la cola o la pila que se reciba por parámetro.