ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL
FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

ESTRUCTURAS DE DATOS
PRIMERA EVALUACIÓN - I PAO 2025

Nombre: _____ **Calificación:**
No. de matrícula: _____ **Paralelo:** __

COMPROMISO DE HONOR: Yo, en mi calidad de estudiante de la ESPOL, declaro que he sido informado y conozco las normas que rigen a la ESPOL, en particular el Código de Ética y el Reglamento de Disciplina.
Acepto con pleno conocimiento este compromiso de honor por el cual reconozco y estoy consciente de que la presente evaluación está diseñada para ser resuelta de forma individual; que sólo puedo comunicarme con la persona responsable de la recepción de la evaluación; que no haré consultas indebidas o no autorizadas por el evaluador; ni usaré dispositivos electrónicos.
Acepto el presente compromiso, como constancia de haber leído y aceptar la declaración anterior y me comprometo a seguir fielmente las instrucciones que se indican para la realización de la presente evaluación.

_____
Firma del estudiante

Tema 1. (35 puntos)

En un aplicativo web de gestión de conversaciones que se generan dentro de la cuenta de un usuario, de las cuales solo se conoce el nombre del contacto y el mensaje recibido. El aplicativo permite navegar entre todas las conversaciones, en ambos sentidos, hacia la conversación siguiente y a la conversación anterior. También, el usuario puede seleccionar su conversación favorita marcándola, con lo cual aparecerá la conversación marcada al inicio de todas las conversaciones, es decir hacer un pin sobre esta. Así mismo, al desmarcar la conversación que tiene el pin, esto hará que la conversación regrese a su puesto original. La limitación es que solo puede marcar (pin) una conversación como favorita.

Implemente los siguientes métodos para marcar y desmarcar las conversaciones:

- **`anclarConversacion(String nombreContacto)`:** Este método debe marcar la conversación como favorita, es decir poner la conversación al inicio de la lista, para esto considerar que:
  1. Si la conversación ya está marcada como favorita (se encuentra al inicio de la lista), no hay cambios.
  2. Si no existe la conversación, el método no debe realizar ninguna acción.
  3. Si ya existe una conversación marcada como favorita, primero debe de desmarcarla haciendo que vuelva a su lugar original dentro de la lista y luego marca la nueva conversación.
- **`desanclarConversacion(String nombreContacto)`:** Este método debe desmarcar la conversación, es decir debe devolver la conversación a su lugar original en la lista, para esto considerar que:
  1. Si la conversación no está anclada, no hay cambios.

Consideraciones:

- Implementar los métodos solicitados en la clase **GestorConversaciones.**
- Asumir que existe la clase **NodoConversacion** con los métodos básicos.
- No utilizar estructuras adicionales como listas dinámicas, arreglos o colecciones de Java.
- Para este tema tendrá solo una conversación por cada contacto.

Ejemplo:

Lista cronológica (de izquierda a derecha)						
[Ana]: [Hola, Vienes?]	⟷	[Profesor] : [Recuerda estudiar]	⟷	[Madre] : [Donde estas?]	⟷	[Amigo] : [Eres mi mejor amig@]

Anclamos al Amigo						
[Amigo] : [Eres mi mejor amig@]	⟷	[Ana] : [Hola, Vienes?]	⟷	[Profesor] : [Recuerda estudiar]	⟷	[Madre] : [Donde estas?]

Anclamos al Profesor						
[Profesor] : [Recuerda estudiar]	⟷	[Ana] : [Hola, Vienes?]	⟷	[Madre] : [Donde estas?]	⟷	[Amigo] : [Eres mi mejor amig@]

Desanclar conversación de Profesor						
[Ana] : [Hola, Vienes?]	⟷	[Profesor] : [Recuerda estudiar]	⟷	[Madre] : [Donde estas?]	⟷	[Amigo] : [Eres mi mejor amig@]

Nota: No hace falta que escriba getters y setters, incluya comentarios que considere oportunos.

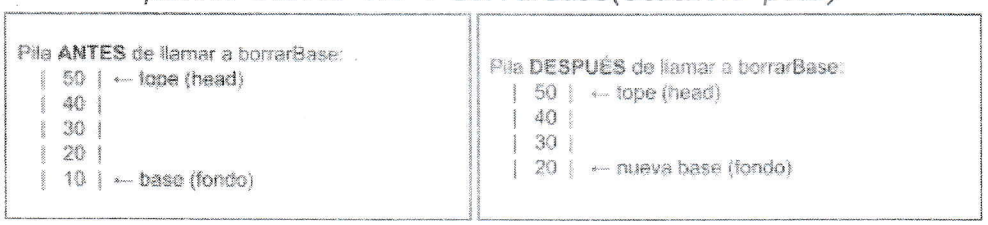
Tema 2. (25 puntos)

a. Analice el siguiente código y responda: ¿Cuál es la complejidad temporal del método **duplicarPila**, considerando que la pila original contiene "**n**" elementos?. Justifiqué su respuesta.

```
public static <T> void duplicarPila(Stack<T> pila) {
      if (pila.isEmpty()) return;
      T tope = pila.pop();
      duplicarPila(pila);
      pila.push(tope);
      pila.push(tope);        }
```

b. Implemente el método **borrarBase**, que elimina el elemento situado en la base (o fondo) de la pila y lo devuelve como resultado. Además, la pila debe quedar con los elementos originales sin la base. Este método deberá implementarse de forma RECURSIVA.

```
public static <T> T borrarBase(Stack<T> pila)
```



c. Diseñe un método genérico que reciba una pila "stack" y la invierta, es decir que el elemento que estaba en la base (fondo) pasará al tope (head) y viceversa.

```
public static <T> void invertirPila(Stack<T> pila)
```

Ejemplo:

```
Pila original: [1, 2, 3, 4, 5]
Pila invertida: [5, 4, 3, 2, 1]
```

Tema 3. (40 puntos)

En el aeropuerto internacional de Berlín, se gestiona una lista de despegue para vuelos de diferentes aerolíneas europeas. Cada vuelo tiene asignada una **prioridad de despegue** representada por un número entero mayor a cero, es decir que a mayor número representa mayor prioridad de salida. El sistema de control de vuelos desea implementar una función que permita **eliminar el enésimo vuelo más prioritario**, según el número de prioridad. Si dos o más vuelos tienen la misma prioridad, **se remueve el que apareció primero** en la listsa de despegue. A continuación se muestra un ejemplo de lista de vuelos (de izquierda a derecha, el primero llegó antes):

Vuelo	Prioridad
AF123	45
LH345	20
BA876	80
KL234	60
AF124	80
LH346	30
BA877	45

Implemente un método **`eliminarEnesimoMasPrioritario`** dentro de una clase Vuelos, que elimina el enésimo vuelo con mayor prioridad. El método debe mantener el orden de llegada de los vuelos restantes. Puede asumir que la clase Vuelos tiene acceso únicamente a los siguientes métodos:

```
void agregar(Vuelo v): Añade un vuelo al final de la lista.
Vuelo sacar(): Elimina y retorna el vuelo al frente de la lista.
Vuelo frente(): Retorna el vuelo al frente sin eliminarlo.
boolean estaVacia(): Indica si la fila está vacía.
int tamano(): Retorna el número de vuelos en la lista.
```

Ejemplo: Se remueve n = 2 (el segundo vuelo más prioritario). Dado el listado anterior, se ha eliminado KL234 porque tenía la segunda mayor prioridad (60), como se muestra en esta tabla.

Vuelo	Prioridad
AF123	45
LH345	20
BA876	80
AF124	80
LH346	30
BA877	45