Antes de pasar a la página siguiente...
¿Has iniciado la grabación del screencast?

Mover un segmento al principio de una lista enlazada simple

Primer parcial - Estructuras de Datos - Grupo F
Facultad de Informática - UCM
Tiene un peso del 20 % en la nota final de la asignatura

Este ejercicio debe entregarse en el siguiente problema *DOMjudge*:

- URL: `http://ed.fdi.ucm.es/`
- Concurso: `ED-F-EV`
- Identificador de problema: `F10`

Debes entregar:

1. El fichero `.cpp` con el código fuente de tu solución. Utiliza la plantilla que se proporciona con este enunciado. El código fuente debe entregarse en *DOMjudge* antes de **1 hora** desde el comienzo del examen. Se evaluará la última entrega realizada antes de la hora límite.
2. Un breve video explicativo (2-5 minutos) de la solución realizada. Este video se entrega a través de la carpeta de *Google Drive* que se ha compartido contigo. Para ello dispones de **30 minutos** una vez finalizado el plazo de entrega del código fuente.
3. El vídeo con el *screencast* de la realización de la prueba. Este video se entrega a través de la carpeta de *Google Drive* que se ha compartido contigo. Para ello dispones de **1 hora** una vez finalizado el plazo de entrega del código fuente.

Recuerda que **no se permite copiar** en la entrega **código fuente externo**, salvo que provenga de la plantilla proporcionada.

Enlace a las instrucciones:
`https://drive.google.com/open?id=1y50P6GtroTeru8-f3TDjn6d8fasIW_9sHNLvMaI5aKs`.

Este ejercicio consiste en extender (mediante herencia) la clase `queue<T>`, la cual implementa el TAD de las colas mediante una lista enlazada simple. Para ello añadimos un nuevo método público llamado `move_segment_left()`.

```
template<typename T>
class queue_move_segment : public queue<T> {
  ...

  private:
    void move_segment_left(int pos_ini, int pos_fin);
};
```

El método `move_segment_left` recibe dos posiciones en la cola, `pos_ini` y `pos_fin`, tales que $0 \leq$ `pos_ini` $\leq$ `pos_fin` $\leq N$, donde N es el número de elementos de la cola. Este método mueve al principio de la cola todos los elementos contenidos entre ambas posiciones, incluyendo el elemento que está en `pos_ini`, pero excluyendo el que está en `pos_fin`. Las posiciones se numeran comenzando desde 0.

Por ejemplo, supongamos la lista `xs = [10, 20, 30, 40, 50, 60, 70, 80]`. Tras realizar la llamada `xs.move_segment_left(2, 5)`, la lista `xs` tiene el valor `[30, 40, 50, 10, 20, 60, 70, 80]`.

Importante: Para la implementación del método no pueden crearse, directa o indirectamente, nuevos

nodos mediante `new`, ni liberar nodos mediante `delete`; deben reutilizarse los nodos de la lista enlazada. Tampoco se permite copiar valores de un nodo a otro. El coste de la operación ha de ser lineal con respecto al valor de `pos_fin`.

A modo de recordatorio, mostramos un fragmento con los atributos de la clase `queue<T>`.

```
template <class T>
class queue {
  protected:

    struct Nodo {
      T elem;
      Nodo *sig;
    };

    Nodo *prim;
    Nodo *ult;
    int nelems;

  ...
};
```

Entrada

La entrada comienza con un número que indica el número de casos de prueba que vienen a continuación. Cada caso de prueba consiste en dos líneas. La primera línea contiene tres números N, i, j, tales que $0 \leq i \leq j \leq N$, donde N es la longitud de la cola, e i y j son, respectivamente las posiciones inicial y final que se pasarán al método a implementar. La segunda línea contiene N números enteros que son los elementos de la cola sobre la que se aplicará dicho método, desde el primero hasta el último.

Salida

Para cada caso de prueba se imprimirá una línea con los elementos de la cola tras la llamada al método, separados por espacios.

Entrada de ejemplo

```
3
8 2 5
10 20 30 40 50 60 70 80
8 5 8
10 20 30 40 50 60 70 80
4 3 3
2 7 1 4
```

Salida de ejemplo

```
30 40 50 10 20 60 70 80
60 70 80 10 20 30 40 50
2 7 1 4
```