Invertir un segmento de una lista enlazada simple

Partimos de la clase `ListLinkedSingle`, que implementa el TAD de las listas de números enteros mediante listas enlazadas simples. Queremos añadir un nuevo método, llamado `reverse_segment()`:

```
class ListLinkedSingle {
private:
  struct Node {
    int value;
    Node *next;
  };
  Node *head;

public:
  ...
  void reverse_segment(int index, int length);
};
```

Esta operación da la vuelta a un segmento (elementos en posiciones consecutivas) de la lista enlazada. Por ejemplo, si la lista es `xs = [1, 2, 3, 4, 5, 6, 7, 8]`, y realizamos `xs.reverse_segment(2,4)`, invertimos el segmento que comienza en la posición 2 (las posiciones se numeran desde 0 hasta $N-1$, el número de elementos de la lista) y tiene longitud 4, obteniendo como lista resultante `[1, 2, 6, 5, 4, 3, 7, 8]` (donde se han marcado en negrita los elementos invertidos).

Se pide:

1. Implementar el método `reverse_segment()`.
2. Indicar su coste con respecto al tamaño de la lista de entrada.

Importante: Para la implementación del método no pueden crearse, directa o indirectamente, nuevos nodos mediante `new` ni borrar nodos mediante `delete`; han de reutilizarse los nodos de la lista de entrada. Tampoco se permite copiar valores de un nodo a otro.

Entrada

La entrada consta de una serie de casos de prueba. Cada caso se muestra en dos líneas. La primera contiene tres números: N, el número de elementos de la lista (un número entre 1 y 1.000); P, la posición de comienzo del segmento (un número entre 0 y $N-1$); y $L \geq 1$, la longitud (número de elementos) del segmento. Se garantiza que el segmento está incluido en la lista, es decir, $P + L - 1 < N$. En la segunda línea se muestran los N elementos de la lista, números entre 1 y 1.000.000.

Salida

Para cada caso de prueba se escribirá en una línea la lista después de invertir el segmento indicado. Puedes utilizar el método `display()` de esta clase.

Entrada de ejemplo

```
8 2 4
1 2 3 4 5 6 7 8
8 0 8
1 2 3 4 5 6 7 8
8 3 1
1 2 3 4 5 6 7 8
```

Salida de ejemplo

```
[1, 2, 6, 5, 4, 3, 7, 8]
[8, 7, 6, 5, 4, 3, 2, 1]
[1, 2, 3, 4, 5, 6, 7, 8]
```

Autor

Alberto Verdejo