Algoritmos y Estructuras de Datos.
RECUPARATORIO LAB [2012-10-17]

Instrucciones

- El examen consiste en escriban la función descriptas más abajo; impleméntandolas en C++ de tal forma que el código que escriban **debe compilar y correr correctamente**, es decir, no se aceptará un código que de algún error de compilación o que tire alguna excepción/señal de interrupción en runtime. Junto tendrán que programar algunos casos de prueba, que les daremos como ejemplo, y la salida debe corresponder a la indicada. Básicamente se hace una evaluación de caja negra, aunque le daremos un rápido vistazo al código.
- Pueden utilizar todas las funciones y utilidades del estándar de C++ que por supuesto contiene a la librería STL.

Consigna

Escribir una función `void make_map_odd_even(list<int> &L,map_t &M);` que dada una lista de enteros `L`, construye la correspondencia `M` que mapea cada elemento **impar** de la lista al mayor rango de elementos contigous **pares** que sigue al elemento impar, por ejemplo si `L=(9,10,6,7,6,8,6,10,2,7)`, entonces debe ser `M=[9->(10,6),7->(6,8,6,10,2)]`.

Nota 1: Notar que si bien 7 aparece dos veces en la lista, el rango correspondiente para la última instancia es el rango vacío, por lo tanto queda el primer rango `(6,8,6,10,2)`.

Nota 2: Si hay varios rangos con la longitud mínima, entonces puede queadar cualquiera de ellos.

Restricción: El algoritmo debe ser $O(n)$, donde n es la longitud de la lista.

Ayuda: Se sugiere el siguiente algoritmo, para cada posición `p`, si el elemento es impar recorrer los siguientes elementos pares poniéndolos en una lista `tmp`. Cuando se llega al final o a un elemento impar se asigna la lista `tmp` all valor almacenado en `p` a condición de que no tenga ya un valor asignado, o si su longitud es menor que la de `tmp`.

mstorti@galileo/aed-2.0.4-135-ge732569/Mon Oct 15 15:22:45 2012 -0300