# Algoritmos y Estructuras de Datos.
# 1er Parcial. [2012-09-13]

## 1. Instrucciones

- El examen consiste en escriban las dos funciones 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.

El ejercicio consiste de dos partes.

### 1.1. Problema del rango de suma nula en una lista

Escribir una función `bool nullsum(list<int> &L, list<int> &L2);` que dada una lista `L`, determina si hay un **rango de iteradores** `[p,q)`) tal que su suma sea 0. En caso afirmativo debe retornar además el rango (uno de ellos, porque puede no ser único) por `L2`. En caso negativo `L2` debe quedar vacía. El algoritmo debe ser $O(n^2)$. Por ejemplo si `L=(-10 14 -2 7 -19 -3 2 17 -8 8)` entonces debe retornar `true` y `L2=(14 -2 7 -19)`, o también `L2=(-8 8)`. Si `L=(1,3,-5)` entonces debe retornar `false` y `L2=()`.

### 1.2. Filtrar con nullsum las entradas de un map

Escribir una funcion `void nullsum_filt(map<int,list<int>> &M, map<int,list<int>> &M2);` que extrae de todos los pares de asignación `(k,L)` de `M` aquellos para los cuales `L` tiene un rango de suma zero, e inserta en `M2` los pares `(k,L')` donde `L'` es el rango de `L` que tiene suma 0.
Por ejemplo si

```
M[0] -> (-7 -4 7 5 3 5 -4 2 -1 -9)
M[1] -> (-8 -3 0 9 -7 -4 -10 -4 2 6)
M[2] -> (1 -2 -3 -1 -8 0 -8 -7 -3 5)
M[3] -> (-1 -8 -8 8 -1 -3 3 6 1 -8)
M[4] -> (1 3 5 -2)
M[5] -> (3 -4 1 -10 6 3 -8 0 6 -9)
M[6] -> (-5 -5 -6 -3 6 -5 -4 -1 3 7)
M[7] -> (-6 5 -8 -5 4 -3 4 -6 -7 0)
M[8] -> (1 5 4)
M[9] -> (2 -10 6 -2 9 2 -4 -4 4 9)
```

entonces debe retornar por `M2` (recordad que como los rangos de suma nula pueden no ser únicos el valor (contradominio) de la correspondencia puede

```
M2[1] -> (0)
M2[2] -> (0)
M2[3] -> (-8 8)
M2[5] -> (3 -4 1)
M2[6] -> (-5 -4 -1 3 7)
M2[7] -> (-5 4 -3 4)
M2[9] -> (-4 4)
```

# 2. SOLUCION

Se adjunta aquí la solución al problema:

-  Decargar `nullsum.cpp`