Curso de Programación en C++.
TPL0-2017. Trabajo Práctico de Laboratorio. [2017-09-11]

PASSWD PARA EL ZIP: YK9L KL5G SKBT

Ejercicios

[Ej. 1] **[separate]** Escribir una función
`void separate(vector<int> &v,vector<int> &even,vector<int> &odd);`
que toma un vector de enteros `v` y separa los pares y los impares, preservando el orden en el que estaban. Por ejemplo:

```
v=[0,1,2,3,4,5,6,7,8], => even=[0,2,4,6,8], odd=[1,3,5,7]
v=[5,9,3,9,16,14,11,11,2,4,1] => even=[16,14,2,4], odd=[5,9,3,9,11,11,1]
v=[0,7,17,5,15,5,16] => even=[0,16], odd=[7,17,5,15,5]
```

[Ej. 2] **[is-greater-than]** Escribir una función
`bool is_greater_than(vector<int> &v1,vector<int> &v2);` Que retorna `true` si la secuencia `v1` es **mayor** que `v2`, en el siguiente sentido:

- Si `v1` y `v2` tienen igual longitud entonces `v1>v2` si para todo `j`: `v1[j]>=v2[j]` y al menos para algún `j` `v1[j]>v2[j]`. Por ejemplo:

  ```
  v1=[1,2,3], v2=[0,1,2] -> true
  v1=[0,1,3], v2=[0,1,2] -> true
  v1=[0,1,3], v2=[0,1,3] -> false
  v1=[3,2,1], v2=[0,1,3] -> false
  ```

- Si los vectores no tienen la misma longitud entonces se considera que se completan con $-\infty$, por lo tanto

  ```
  v1=[0,1,2,0], v2=[0,1,2] -> true
  v1=[0,1,2], v2=[0,1,2,0] -> false
  ```

  Nota: Los elementos de `v1,v2` son todos `>=0` de manera que se puede usar `-1` en vez de $-\infty$.

Ayuda: Se sugiere el siguiente algoritmo.

- Escribir una función `int elem(vector<int> &v,int j);` que retorna el elemento correspondiente `v[j]` o bien `-1` si la posición no existe.
- Sea `n=max(n1,n2)` la máxima longitud de ambos vectores. Recorrer todas las posiciones en `0<=j<n` y chequear si `elem(v1,j)<elem(v2,j)` para algún `j`, en dicho caso retornar `false`.
- Recorrer nuevamente todos los `j`, si `elem(v1,j)>elem(v2,j)` para algún `j` retornar `true` caso contrario retornar `false`.

[Ej. 3] **[max-sum]** Escribir una función
`int max_sum(vector<vector<int» &vv);` Que toma un vector de vector de enteros `vv` y retorna el índice `j` en `vv` tal que la suma de los elementos de `vv[j]` es mayor o igual que la de todos los otros. Si hay varios que tienen la misma suma máxima debe retornar el primero. Por ejemplo:

```
vv=[[0,1,2],[4],[3,2]] => 2
vv=[[0,1,2],[5],[3,2]] => 1
```

Instrucciones generales

- El examen consiste en que escriban las funciones descriptas más abajo; impleméntandolas en C++ de tal forma que el código que escriban **compile y corra correctamente**, es decir, no se aceptará un código que de algún error de compilación o que tire alguna excepción/señal de interrupción en runtime. 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.
- Se incluye un template llamado `program.cpp`. En principio sólo tienen que escribir el cuerpo de las funciones pedidas.
- Hay una función de evaluación, por ejemplo si `f` es la función a evaluar tenemos

  ```
  ev.eval1(f,vrbs);
  ```

  `ev.eval1(f,vrbs);` toma una serie de casos de prueba de entrada, le aplica la función del usuario `f` y compara la salida del usuario (`user`) con respecto a la esperada (`ref`). Si la verbosidad (el argumento `vrbs`) se pone en uno, entonces la función evaluadora reporta por consola los datos de entrada, la salida de la función de usuario y la salida esperada

mstorti@galileo/texstuff-1.4.0-69-gb3543f11/Tue Jun 22 10:41:21 2021 -0300