UNIVERSIDAD DE CARABOBO
Facultad Experimental de Ciencias y Tecnología
Departamento de Computación
U.A. de Algoritmos, Programación y Lenguajes
CAO403: Programación II

Taller #1
Período Lectivo 1-2015
(Valor: 30%)

1.INSTRUCCIONES

- Este taller de programación es una evaluación estrictamente individual; por lo tanto, durante la realización del mismo, no está permitido prestar o recibir ayuda de otro(s) estudiante(s).
- ESTA permitido el uso de códigos fuentes en digital hechos por usted para el desarrollo del taller.
- Pueden traer material bibliográfico (libros, guías, manuales, código impreso, etc.) para el desarrollo del taller de programación.
- El estudiante que infrinja alguna de las instrucciones anteriores será retirado del taller y se le considerará aplazado con la nota mínima (Artículo 34 de las Normas de Evaluación de los Aprendizajes).
- **El taller tiene una duración estimada de tres (03) horas académicas.**
- El ejercicio propuesto requiere la lectura de datos de la entrada estándar (standard input) y la escritura de resultados en la salida estándar (standard output).
- Para el desarrollo del taller debe utilizar el lenguaje de programación C++, junto con sus librerías estándar. Para la compilación de sus códigos fuentes, debe realizarla por medio de un archivo makefile, el cual deberá entregar junto con sus códigos fuentes.
- Debe proveer un archivo **makefile** que permita compilar su taller, la falta del mismo conllevará a la no revisión de su taller, **adicionalmente dicho makefile debe generar un ejecutable con el nombre de** "***sublista",*** con el fin de facilitar las labores de corrección de su taller, el no hacerlo generara puntos menos en su nota final del taller.

2.ACTIVIDADES A REALIZAR

Una lista de enteros ***A*** es ***sublista*** de ***B***, si todos los elementos de ***A*** se encuentran en ***B*** en el mismo orden y de manera consecutiva. Utilizando el paradigma de programación orientado a objetos, extienda la ***clase Lista*** simplemente enlazada vista en el aula de clases, e implementada mediante el uso de plantillas, implementando un método que reciba una lista y determine ***cuantas veces*** dicha lista es ***sublista***.

2.1. Formato de entrada

Cada caso de prueba constara de cuatro (4) líneas, en la primera línea se indicara un numero entero ***m***, que representara el numero de elementos de la lista ***A***, la siguiente línea contendrá ***m*** números enteros separados por un espacio en blanco, los cuales serán cargados en dicha lista, la siguiente línea contendrá un numero ***p*** que representara el numero de elementos de la lista ***B***, la siguiente línea contendrá ***p*** números enteros separados por un espacio en blanco, los cuales serán cargados en la lista ***B.***

2.2. Formato de salida

La salida constara de ***n*** líneas sin líneas vacías de por medio, que representaran la salida para los ***n*** casos de prueba, que contendrá el numero de veces que ***A*** es sublista de ***B*** para cada caso de prueba.

Ejemplo:

Entrada

```
5
30 10 40 20 50
11
30 30 30 10 40 20 50 60 50 20 400
3
2 2 2
6
2 2 2 2 2 2
```

Salida

```
1
4
```

3.OBSERVACIONES

- No debe mostrar por pantalla ningún tipo de mensaje al usuario para su interacción con el programa, ninguna interfaz gráfica desarrollada por usted, será corregida positivamente o le aportara ningún punto extra para su calificación final
- El formato de salida debe ser respetado como se indica en el enunciado, de lo contrario acarreará con puntos menos en su nota final.