UNIDAD TEMÁTICA 4 – GRAFOS DIRIGIDOS– Trabajo de Aplicación 6

Ejercicio 1 (total 30 minutos)

1. Desarrollar en lenguaje natural y seudocódigo de alto nivel un algoritmo sencillo que permita determinar si un grafo contiene ciclos (**15** minutos)
2. Dado el código de TDA GrafoDirigido ya implementado y las clases auxiliares distribuidas, crear un método sencillo para determinar si el grafo tiene ciclos o no. (implementar tanto el método de Grafo como el de Vértice), respetando las firmas (20 minutos):
    - Boolean TGrafoDirigido.tieneCiclo(); // indica si el grafo tiene ciclo y si es así, imprime las etiquetas de los vértices involucrados en el ciclo encontrado.
    - Boolean TVertice.tieneCiclo (TCamino unCamino);

```
Precondiciones:
     El camino está creado y vacío.

Postcondiciones:
     El camino contendrá todas las etiquetas del ciclo.
```

3. Dados los archivos “conexiones_2.txt” y “aeropuertos_2.txt”, ejecutar el programa e indicar si el grafo resultante contiene ciclos o no. (5 minutos)
   Entregable: seudocódigo en la tarea TAREA UT4_TA6, código fuente en el GIT hasta la hora 21:15

Ejercicio 2 – Camino crítico

La ordenación topológica o sort topológico es ampliamente utilizado en planificación y gestión de proyectos o procesos que tienen una gran cantidad de tareas. Las tareas suelen tener recursos asignados (tiempo, horas hombre, materia prima necesaria, etc.). Las tareas pueden verse como un cambio de estado de un cierto elemento. Existen naturalmente precedencias entre las tareas (ejemplo: antes de pintar la pared es necesario revocarla, y antes construirla, etc etc).

De todos los posibles "caminos" que representan las diferentes secuencias de tareas, existe al menos uno que, si cualquiera de las tareas que lo constituyen, se atrasa, entonces todo el proyecto se atrasará. Este es el llamado "camino crítico" del proyecto (una herramienta muy utilizada en planificación es el PERT, "diagrama de camino crítico").

Las demás secuencias (que no son camino crítico), podrán tener ciertas demoras en algunas de sus tareas, hasta alcanzar la magnitud del camino crítico. Por ejemplo, si una secuencia de tareas que es camino crítico tiene un costo total de 23 días (suma de los costos de las tareas o arcos), y otra secuencia tiene un costo o duración total de 15 días, esto significará que existe una "holgura" de 8 días, es decir, algunas de las tareas que componen esta secuencia podrán empezar y terminar en diferentes fechas, siempre que no excedan la final. Esto nos permite, al planificar los trabajos, buscar formas de distribuir los recursos existentes de manera más homogénea, evitar picos de necesidad de recursos o espacios ociosos.

Se desea desarrollar entonces un algoritmo que, dado un grafo (que debería se acíclico) permita

- calcular el costo total del camino crítico e indicar cómo está compuesto este camino, y
- listar todas las demás secuencias de tareas posibles, sus costos totales y la holgura existente
- deberán establecerse las estructuras apropiadas con las modificaciones que sean necesarias.