INF-134 Estructuras de Datos, 2016-1

Tarea 4

Profesor: Diego Arroyuelo B.
darroyue@inf.utfsm.cl

Ayudantes:
Francisco Olivares francisco.olivars.14@sansano.usm.cl,
Camilo Saldias camilo.saldias.12@sansano.usm.cl,
Cristian Vallejos cristian.vallejos.14@sansano.usm.cl.

Fecha de entrega: 8 de Julio, 2016
Plazo máximo de entrega: 5 horas.

1. Reglas del Juego

La presente tarea debe hacerse en grupos de 3 personas. Toda excepción a esta regla debe ser conversada con los ayudantes ANTES de comenzar la tarea. No se permiten de ninguna manera grupos de más de 3 personas. Las tareas deben compilar en los computadores que se encuentran en el laboratorio LDS – B-032. Deben usarse los lenguajes de programación C o C++. Se recomienda compilar en el terminal usando `gcc archivo.c -o output -Wall` (en el caso de lenguaje C) o `g++ archivo.cpp -o output -Wall` (en el caso de C++). Recordar que a lo más una tarea en el semestre puede tener nota menor a 30. El no cumplimiento de esta regla implica reprobar el curso. Además, si todas las tareas anteriores tienen nota ≥ 30, la nota de la presente tarea reemplaza la peor nota de las tareas anteriores (en caso de que ese cambio convenga).

2. Objetivos

Entender y familiarizarse con el uso de grafos.

3. Ciudades Inalcanzables

Para poder brindar un mejor servicio a sus usuarios, una aerolínea está tratando de determinar qué ciudades no son alcanzables mediante vuelos de la compañia desde una ciudad de origen dada. Diseñar un algoritmo eficiente, que haga uso de grafos dirigidos para modelar la situación y resolver el problema.

4. Entrada de Datos

La entrada de datos será a través de la entrada standard, y consiste de un único caso de prueba por vez. La entrada comienza con una línea conteniendo un único valor `N` tal que $1 \leq$ `N` $\leq$ 1,000,000. Éste indica el número de ciudades con las que se quiere trabajar. Cada ciudad será representada por un entero en el rango 0 a `N` $- 1$. Luego, le sigue una línea conteniendo un único valor `M` tal que $0 \leq$ `M` $\leq$ 50,000,000, indicando la cantidad de vuelos entre pares de ciudades que tiene la compañía. Luego le siguen `M` líneas, cada una con dos valores `O D`, separados por un único espacio, indicando que existe un viaje desde la ciudad origen `O` hasta

la ciudad destino `D`. Luego, sigue un único valor `Q` tal que $1 \leq$ `Q` $\leq 10{,}000$, indicando el número de consultas que se harán sobre el grafo. A continuación le siguen `Q` líneas, cada una conteniendo un vértice `V` que será usado como consulta.

Un ejemplo de entrada válida es:

```
7
7
0 1
6 2
2 3
4 3
5 4
0 6
2 5
3
2
3
0
```

Para facilitar la prueba, se recomienda crear un archivo (por ejemplo, `input-edd-t4.txt`), el cual contenga los casos de prueba con el formato pedido. Luego, en la línea de comando (al momento de ejecutar el programa, suponiendo que el ejecutable se llama `tarea`) se redirecciona la entrada standard de la siguiente manera:

```
./tarea < input-edd-t4.txt
```

Aquí, "`<`" indica al sistema que los datos deben ser leídos desde el archivo indicado, y no desde el teclado como es usual. De esta manera, no hay que reingresar los casos de prueba cada vez que se usa el programa.

5. Salida de Datos

La salida de datos se realizará a través de la salida standard. La primera línea de la entrada consiste del valor `Q`, indicando la cantidad de consultas realizadas (es el mismo valor `Q` de la entrada). Luego le siguen `Q` líneas, cada línea conteniendo los vértices que son inalcanzables desde cada vértice `V` de la entrada. Cada una de estas líneas comienza con un valor `R` indicando la cantidad de vértices inalcanzables. Luego, le siguen `R` valores en la misma línea, cada uno separado por un único espacio. Los vértices en cada línea deben estar ordenados de forma creciente.

La salida correspondiente a la entrada de ejemplo de la sección anterior es:

```
3
3 0 1 6
6 0 1 2 4 5 6
0
```

6. Entrega de la Tarea

La entrega de la tarea debe realizarse enviando un archivo comprimido llamado

`tarea4-apellido1-apellido2-apellido3.tar.gz`

(reemplazando sus apellidos según corresponda) en el moodle del curso, a más tardar el día 8 de julio de 2016, a las 23:55:00 hs (Chile Continental), el cual contenga:

- Los archivos con los códigos fuentes necesarios para el funcionamiento de la tarea. Los archivos deben compilar!
- **nombres.txt**, Nombre, ROL, Paralelo y qué programó cada integrante del grupo.
- **README.txt**, Instrucciones de compilación en caso de ser necesarias.

7. Restricciones y Consideraciones

- Por cada hora de atraso en la entrega de la tarea se descontarán 10 puntos en la nota.
- El plazo máximo de entrega es 5 horas después de la fecha original de entrega.
- Pueden programar la tarea en C o C++ según ustedes consideren conveniente. Al programar en C++ queda prohibido utilizar la librería STL.
- Las tareas deben compilar en los computadores que se encuentran en los laboratorios LDS (B-032). **Las tareas que no compilen no serán revisadas y serán calificadas con nota 0**.
- Por cada *Warning* en la compilación se descontarán 5 puntos.
- Si se detecta **COPIA** la nota automáticamente sera 0 (CERO), para todos los grupos involucrados. El incidente será reportado al jefe de carrera.
- La prolijidad, orden y legibilidad del código fuente es obligatoria. Habrá descuentos si alguno de estos items no se cumple.

8. Consejos de Programación

El código fuente del programa debe estar estructurado adecuadamente en archivos (separados de ser necesario). Si el código fuente está desordenado, se pueden descontar hasta 20 puntos de la nota.

Cada función programada debe tener comentarios de la siguiente forma:

```
/*****
*   TipoFunción NombreFunción
******
*   Resumen Función
******
*   Input:
*       tipoParámetro NombreParámetro : Descripción Parámetro
*        .......
******
*   Returns:
*       TipoRetorno, Descripción retorno
*****/
```

Por cada comentario faltante, se restarán 5 puntos.

Por último, la indentación (1 TAB o 4 espacios), es muy importante. Por **cada bloque mal indentado, se quitarán 10 puntos**.