Universidad de los Andes
Escuela de Ingeniería

Estructuras de Datos y Algoritmos (II/2017)

Tarea 2

Esta tarea la pueden hacer solos o en parejas.

1 Problema

Sea G un grafo dirigido con vertices numerados desde 1 hasta n tal que cada vertice es alcanzable desde el vertice 1. Además, cada par de vértices está conectado por a lo más una arista (u,v).

Considere el algoritmo de DFS partiendo desde el vertice 1. Como cada vértice es alcanzable, cada arista (u,v) de G es clasificada en uno de cuatro grupos

1. tree edge: si v fue descubierto por primera vez cuando se recorrió (u,v)
2. back edge: si v ya estaba en el stack cuando se recorrió (u,v)
3. forward edge: si v ya se había descubierto mientras u estaba en el stack
4. cross edge: cualquier arista que no es uno de las anteriores

Para entender mejor lo anterior, considere el siguiente código:

```cpp
 // initially false
bool discovered[n];

// initially false
bool finished[n];

vector<int> g[n];

void dfs(int u) {
  // u is on the stack now
  discovered[u] = true;
  for (int v: g[u]) {
      if (finished[v]) {
          // forward edge if u was on the stack when v was discovered
          // cross edge otherwise
          continue;
      }
      if (discovered[v]) {
          // back edge
          continue;
      }
      // tree edge
      dfs(v);
  }
  finished[u] = true;
  // u is no longer on the stack
}

```

Figure 1: Algoritmo DFS

Dado 4 enteros t, b, f y c, construya cualquier grafo G que tenga exactamente:

- Una cantidad t de tree edges.
- Una cantidad b de back edges.
- Una cantidad f de forward edges.
- Una cantidad c de cross edges.

2 Formato entrada y salida

Los archivos de entrada tendrán el siguiente formato:

1. Una sola línea con cuatro enteros separados por espacios t, b, f y c.

Los archivos de salida deberán tener el siguiente formato:

1. La primera línea contiene un entero n, la cantidad de vértices en G.
2. Cada línea i contiene los siguientes enteros:
   - El primer entero es la cantidad de aristas que salen del nodo.
   - Luego el índice de cada uno de los nodos a los que está conectado el nodo i.

```
3 1 1 1

```

Figure 2: Ejemplo de Input

```
4
3 2 4 3
1 3
1 1
1 2

```

Figure 3: Ejemplo de Output

3 Evaluación

Ud. debe implementar su solución usando C++[1] y debe compilar usando el toolchain provisto por Clang (`clang++`).

Su solución *debe* compilar correctamente, sin errores, por menores que sean. Una tarea que no compila será evaluada con un 1.0.

Aquellas tareas que compilen correctamente serán evaluadas en las siguientes dimensiones:

- La corrección ante todo input
- La implementación de la solución

[1] No es necesario que su solución sea orientada a objetos, pero si le acomoda, adelante.

3.1 Corrección

Su solución debe producir resultados correctos para cada input que se le pase. Se probarán toda clase de inputs, de todo tipo de largos.

4 Entrega

Tienen tres semanas a partir de la fecha de publicación de este enunciado. El método de entrega será un repositorio git a su elección (bitbucket o github). Recuerden que esta tarea la pueden hacer en grupos de a dos personas.