Tarea 3
Teoría de Grafos

Ejercicio 1 — 10 Pt

(a) [5 Pt] Pruebe que si G es un grafo simple de 11 vértices, entonces G o su complemento $\overline{G}$ son no-planares.[1]

(b) [5 Pt] Pruebe que el resultado anterior se extiende a cualquier grafo simple de más de 11 vértices.

Ejercicio 2 — 10 Pt

Un grafo es *agradable* si eliminar cualquiera de sus vértices (y aristas incidentes) reduce su número cromático. Probar que si G es un grafo agradable de número cromático k, entonces

(a) [5 Pt] G es conexo;

(b) [5 Pt] todo vértice de G tiene grado al menos $k-1$.

Ejercicio 3 — 40 Pt

Frederick Olmue es un ávido cinéfilo. Le gusta hacer maratones de películas los domingos y sus actores favoritos son Kevin Bacon y Maryl Streep. Las maratónes de Fred son muy peculiares. Todos los domingos por la mañana Fred escoge un actor A (que no es Bacon ni Streep), luego escoge las películas de la maratón de manera que cumplan las siguientes condiciones:

1) A debe actuar en la PRIMER película.
2) Maryl Streep debe actuar en ALGUNA película de la maratón.
3) Kevin Bacon debe actuar en la ÚLTIMA película de la maratón.
4) Cada película debe compartir AL MENOS un actor con la película anterior (excepto la primera).

Si bien Frederick (o Fred, para los amigos) disfruta mucho sus maratones, le aburre escoger las películas de la maratón cada mañana, por lo que te pide ayuda a ti, un EXPERTO en grafos, para que lo ayudes con este problema. Fred quiere que programes tres funciones: `NroMinPeliculas`, `UnaMaratonMinimal`, `NroMaratonesMinimales`. Las tres funciones reciben como entrada el nombre del actor A, la colección de películas de la que Fred dispone y la lista de actores de cada película. Cada una debe retornar, respectivamente:

(a) [16 Pt] `NroMinPeliculas`: la cantidad minima de películas necesarias para hacer tal maratón (para que Fred sepa si alcanza a ver la maratón en un día o no).

(b) [12 Pt] `UnaMaratonMinimal`: una lista con nombres de películas que califiquen como maratón y donde la cantidad de películas sea mínima. (Fred es un hombre ocupado, no puede ver películas en otros días).

[1] El complemento de un grafo simple $G = (V, E)$ es el grafo $\overline{G} = (V, \overline{E})$ donde $e \in \overline{E}$ si y sólo si $e \notin E$.

(c) [12 Pt] `NroMaratonesMinimales`: la cantidad de maratones distintas posibles donde la cantidad de películas sea mínima.

Para estas funciones el nombre del actor A se entregara como un string, la colección de Fred se entregara como una lista de strings, y los actores de cada película serán una lista de listas de strings. A continuación se muestra el tipo de cada entrada para cada lenguaje:

Python

```
A="Peibl Towers"
coleccion= ["Eficiente y furioso 3: Dijkstra Drift","Mi papa es un grafo!","Discretas 4, una
 ↪  nueva esperanza"]
actores=[["Peibl Towers","Vim Diesel"],["Vim Diesel","Maryl Streep", "Andrea
 ↪  Lalancha"],["Andrea Lalancha","Kevin Bacon"]]
print(NroMinPeliculas(A,coleccion,actores)) ##imprime 3
print(UnaMaratonMinimal(A,coleccion,actores)) ##imprime ["Eficiente y furioso 3:
 ↪  Dijkstra Drift","Mi papa es un grafo!","Discretas 4, una nueva esperanza"] :D
print(NroMaratonesMinimales(A,coleccion,actores)) ##imprime 1
```

C++

```
string A="Peibl Towers";
vector<string> coleccion= {"Eficiente y furioso 3: Dijkstra Drift","Mi papa es un
 ↪  grafo!","Discretas 4, una nueva esperanza"};
vector<vector<string> > actores={{"Peibl Towers","Vim Diesel"},{"Vim Diesel","Maryl
 ↪  Streep", "Andrea Lalancha"},{"Andrea Lalancha","Kevin Bacon"}};
cout<<NroMinPeliculas(A,coleccion,actores); //imprime 3
vector <string> res=UnaMaratonMinimal(A,coleccion,actores);
for (int i=0;i<res.size();i++)
   cout<<'\"' << res[i] << "\" ";// imprime "Eficiente y furioso 3: Dijkstra Drift" "Mi papa
 ↪  es un grafo!" "Discretas 4, una nueva esperanza" :D
cout<<NroMaratonesMinimales(A,coleccion,actores); //imprime 1
```

Java

```
String A="Peibl Towers";
String[] coleccion= {"Eficiente y furioso 3: Dijkstra Drift","Mi papa es un
 ↪  grafo!","Discretas 4, una nueva esperanza"};
String[][] actores={{"Peibl Towers","Vim Diesel"},{"Vim Diesel","Maryl Streep", "Andrea
 ↪  Lalancha"},{"Andrea Lalancha","Kevin Bacon"}};
System.out.print(NroMinPeliculas(A,coleccion,actores)); //imprime 3
String[] res=UnaMaratonMinimal(A,coleccion,actores);
for(int i=0;i<res.length ;i++)
      System.out.print("\""+res[i]+"\" ");// imprime "Eficiente y furioso 3: Dijkstra
 ↪  Drift" "Mi papa es un grafo!" "Discretas 4, una nueva esperanza" :D
System.out.print(NroMaratonesMinimales(A,coleccion,actores)); //imprime 1
```

Importante: Asuma que A, Maryl Streep y Kevin Bacon nunca estarán en la misma película. Para construir los grafos en Python puede usar la librería *networkx*. Se permite el uso de la librería sólo para construir o explorar los grafos (está prohibido el uso de algoritmos, por ejemplo Dijkstra o DFS).

Bonus (10pt extra)

Importante: *Esta parte sólo se corregirá si ha obtenido nota* 4.0 *o superior en esta pregunta.*

(d) [5 Pt bonus] Usando la libreria de Python *networkx* haga una función `ImprimirUnaMaratonMinimal` que tome los mismos tres argumentos que las funciones ya definidas y muestre gráficamente el grafo donde los nodos sean las películas (el nodo debe mostrarse con el nombre de la película que representa), las aristas conecten pares de nodos que representan películas con al menos un actor en común, y donde se destaquen los nodos y aristas que representan la maratón devuelta por `UnaMaratonMinimal`.

(e) [5 Pt bonus] Una función `ImprimirTodasLasMaratonesMinimales` similar a la anterior, pero que destaque en distintos colores todos los caminos que representan maratones de largo mínimo.