Taller 1: Grafos

Juan Sebastian Oviedo Oviedo (2179238-2724),
Kirk Olaya Villamarín (2178638-2724), Juan Sebastian Hurtado Batioja (2179288-2724)

juan.sebastian.oviedo@correounivalle.edu.co , kikr.olaya@correounivalle.edu.co,
juan.batioja@correounivalle.edu.co

Escuela de Ingeniería de Sistemas y Computación
Facultad de Ingeniería
Fecha de entrega: 2022-12-13

Actividad 1: Grafos y su representación

1. ¿Que es un Grafo?

R. Una estructura de datos definida por un conjunto de vértices (V) y una colección de aristas o bordes (A) que representan un vínculo o una conexión entre dos o más vértices.

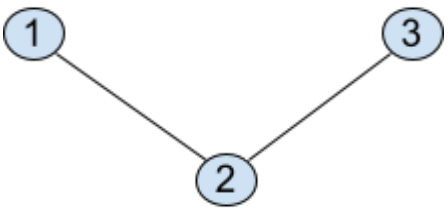
2. ¿Que es un Grafo simple?

R. Grafos no dirigidos que no contienen loops o aristas múltiples.

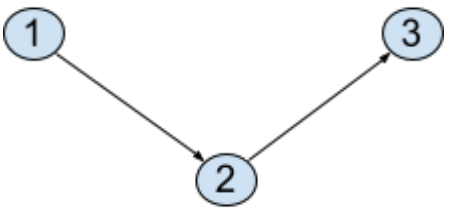
3. ¿Que es un Grafo ponderado?

R. Grafos en los que se le da un valor numérico a las aristas.

4. De un ejemplo de un grafo no dirigido y uno dirigido



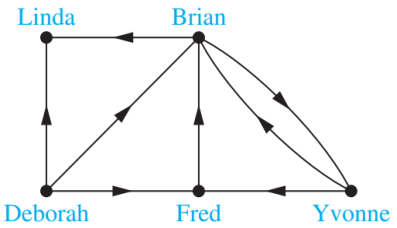
Grafo no dirigido



Grafo dirigido

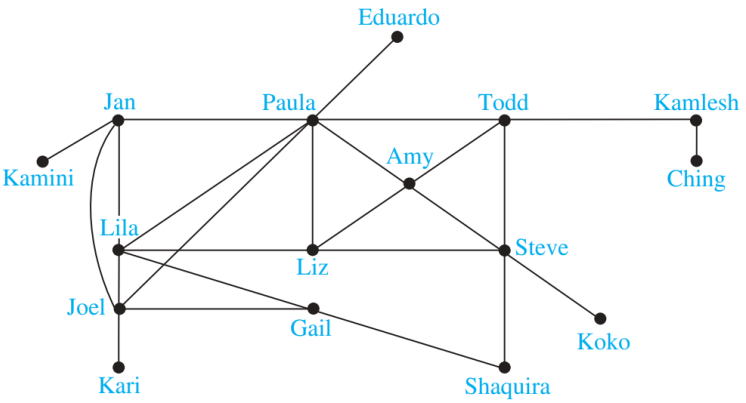
5. Dé un ejemplo de algo en la vida real que se pueda modelar por medio de un grafo dirigido.

R.



En estudios de comportamiento grupal se ha observado que algunas personas pueden influenciar el pensamiento de otras, los grafos dirigidos se pueden usar para modelar este comportamiento. Cada persona en el grupo se representa por un vértice. Habrá una arista dirigida de un vértice X a un vértice Y si la persona representada por X puede influenciar a Y.

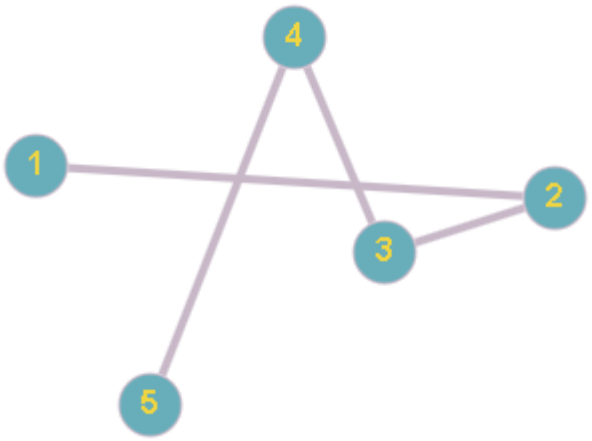
6.Dé un ejemplo de algo en la vida real que se pueda modelar por medio de un grafo no dirigido.



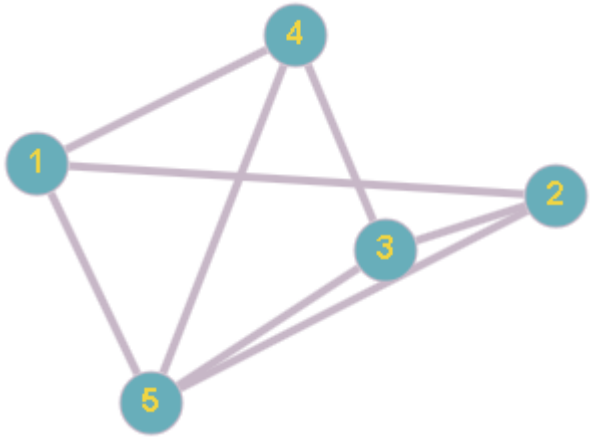
Se puede usar un grafo no dirigido para representar si dos personas se conocen. Cada persona se representa por un vértice. Una arista no dirigida se usa para conectar a dos personas que se conocen.

7. Realice el grafo que se representa con las siguientes matrices de adyacencia.

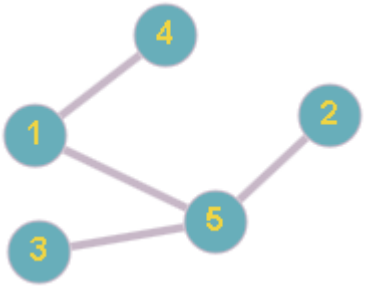
a)



b)

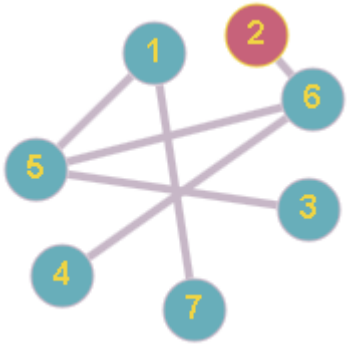


c)

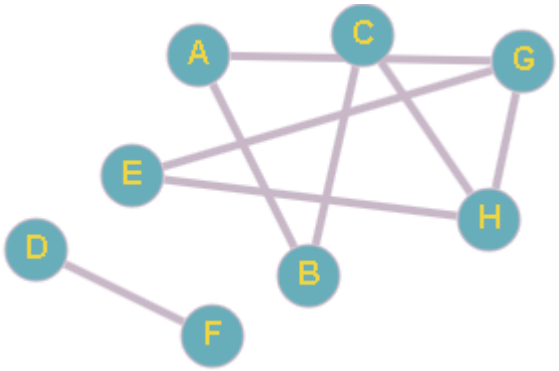


8. Realice el grafo que se representa con las siguientes listas de adyacencias.

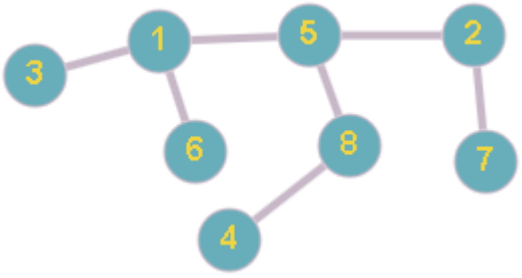
a)



b)



c)



9. Obtenga la matriz y la lista de adyacencia de los siguientes grafos.

a) Lista de adyacencia:
V: s | w | t
S: v | w
W: s | v | t
T: v | w

Matriz de adyacencia:

	V	S	W	T
V	0	1	1	1
S	1	0	1	0
W	1	1	0	1
T	1	0	1	0

b) Lista de adyacencia
1: 4 | 5 | 6
2: 4 | 5 | 6
3: 4 | 5 | 6
4: 1 | 2 | 3
5: 1 | 2 | 3
6: 1 | 2 | 3

Matriz de adyacencia

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

c) Lista de adyacencia

1: 3 | 6 | 7

2: 4

3: 4 | 1

4: 5 | 2

5: 4

6: 1

7: 1

Matriz de adyacencia

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

Actividad No. 2

A continuación se presenta el análisis del problema llevado a cabo para el desarrollo del programa planteado (tabla 1), en la figura 1 se evidencia la prueba de ejecución del programa.

Tabla 1. *Análisis del problema.*

Definición del problema	Desarrollar un programa que implemente los grafos como una estructura de datos. Este debe leer archivos de texto y representar de manera gráfica los grafos contenidos en los archivos
Entradas	Archivo: contiene información de un grafo (formato JAPeTo)
Salidas	grafo, vértices, aristas, tipo_grafo
Proceso	● Definir la clase "LectorGrafos", encargada de la lectura de archivos. ○ lec = LectorGrafos() -> Se encarga de leer los archivos de texto con los grafos y almacena la información correspondiente para luego ser utilizada. ○ Se inicializa el tipo y la representación del grafo contenido en el archivo. ○ self.datos = None - self.datos_procesados = None - self.tipo_grafo = "" #Directed (D), Non-Directed (N) or Weighted (P) - self.representacion_grafo = "" #adjacency matrix (MA), adjacency list (LA) or incidence matrix (MI) ● Definir la clase "Grafo", encargada de implementar la estructura de datos para un grafo. -> grafo = Grafo(lectorGraf.datos_procesados, lectorGraf.tipo_grafo, lectorGraf.representacion_grafo) ○ Definir clase auxiliar "Vertice" - > nuevo_vertice = Vertice(id_vertice, x, y, etiqueta) ○ Definir clase auxiliar "Arista" -> nueva_arista = Arista(cola, cabeza) ○ Dado los datos de un grafo, inicializar atributos del grafo (tipo de grafo, representación, vértices y aristas). ○ Dibujar grafo -> grafo.dibujar_grafo() ● Definir la clase "GUI", encargada de presentar el grafo leído y las características del mismo de manera gráfica. -> app = App()

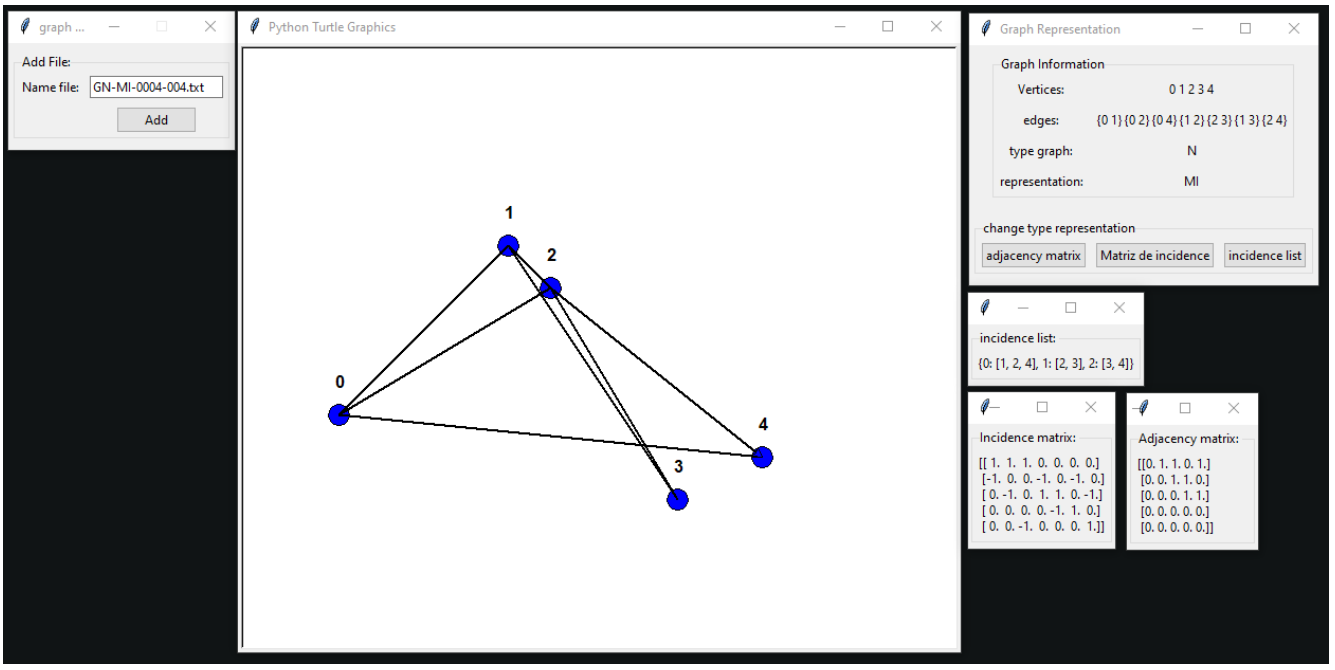


Figura 1. Prueba de ejecución.