Pontificia Universidad Javeriana

Taller de Grafos - Parte teórica

Gabriel Jaramillo Cuberos
Salomon Avila

Asignatura: Estructuras de datos

Profesor: John Jairo Corredor Franco

20 de mayo de 2025

1.- Responda las siguientes preguntas teniendo en cuenta el grafo representado por la siguiente matriz de adyacencia:

$$\mathbf{G_1} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{G_2} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{G_3} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad \mathbf{G_4} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Imagen 1: Matrices de adyacencia

Grafos:

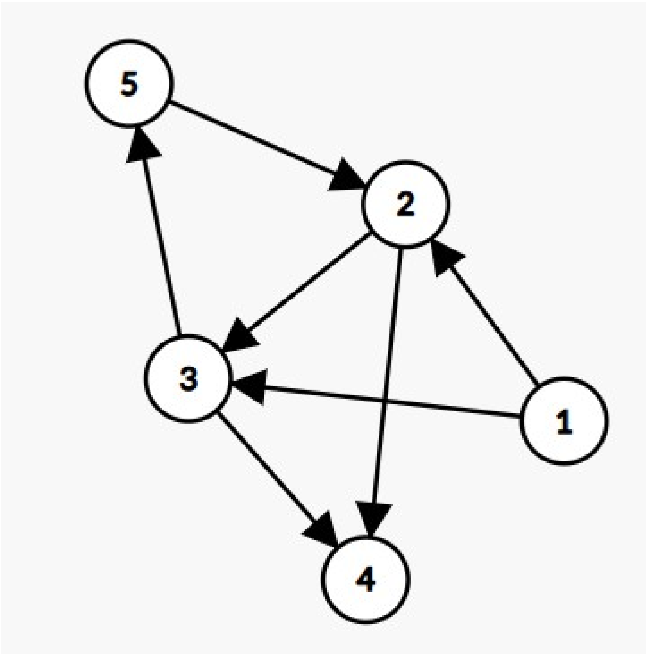


Imagen 2: Grafo número 1

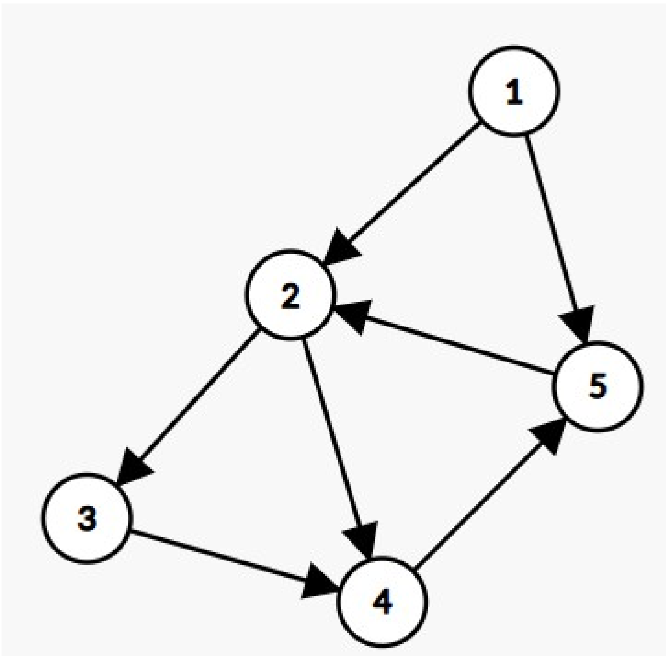


Imagen 3: Grafo número 2

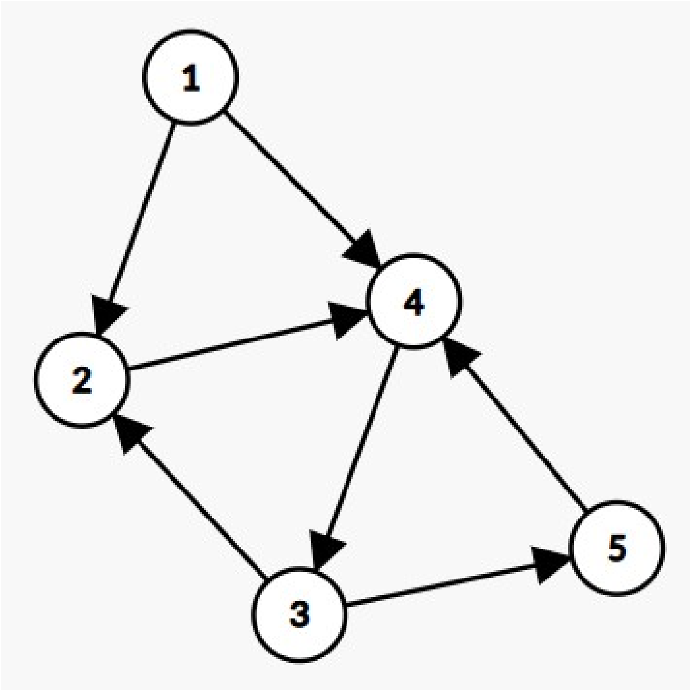


Imagen 4: Grafo número 3

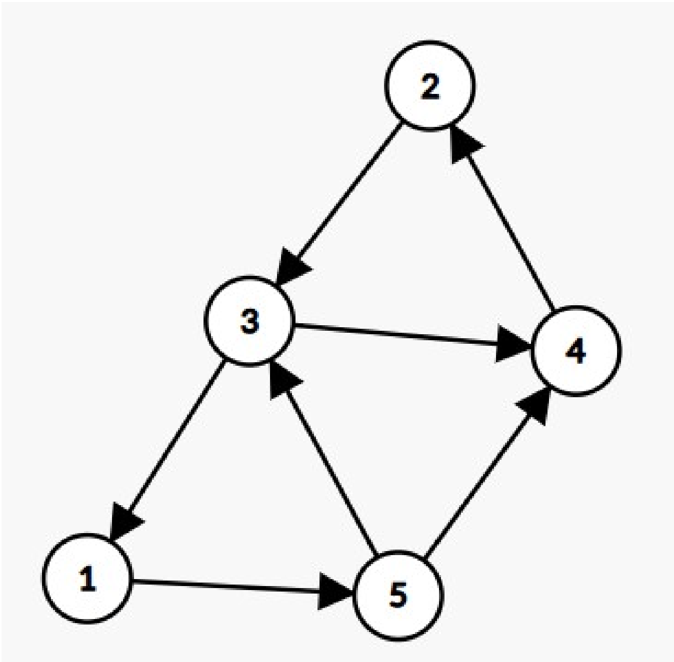


Imagen 5: Grafo número 4

2.- ¿Es un grafo cíclico o acíclico? En caso de ser cíclico, describa todos los ciclos en el grafo.

a. Grafo 1: es un grafo cíclico. El ciclo es 2,3 3,5 5,2
b. Grafo 2: es un grafo cíclico. Los ciclos son 2,4 4,5 5,2 | 2,3 3,4 4,5 5,2
c. Grafo 3: es un grafo cíclico. Los ciclos son 2,4 4,3 3,2 | 4,3 3,5 5,4
d. Grafo 4: es un grafo cíclico. Los ciclos son 1,5 5,3 3,1 | 3,4 4,2 2,3 | 1,5 5,4 4,2 2,3 3,1

3.- ¿Hay vértices fuente? ¿Cuáles son?

a. Grafo 1: el vértice fuente es el vértice número 1.
b. Grafo 2: el vértice fuente es el vértice número 1.
c. Grafo 3: el vértice fuente es el vértice número 1.
d. Grafo 4: no hay vértices fuente.

4.- ¿Hay vértices sumidero? ¿Cuáles son?

a. Grafo 1: el vértice fuente es el vértice número 4.
b. Grafo 2: no hay vértices sumidero.
c. Grafo 3: no hay vértices sumidero.
d. Grafo 4: no hay vértices sumidero.

5.- ¿Cuáles son los vértices descendientes de 2?

a. Grafo 1: los vértices 2, 3, 4, 5.
b. Grafo 2: los vértices 2, 3, 4, 5.
c. Grafo 3: los vértices 2, 3, 4, 5.
d. Grafo 4: los vértices 1, 2, 3, 4, 5.

6.- ¿Cuántos componentes fuertemente conectados hay en el grafo?

a. Grafo 1: 3 componentes: ({1}, {4}, {2,3,5})
e. Grafo 2: 3 componentes: ({1}, {2,4,5}, {2,3,4,5})
f. Grafo 3: 3 componentes: ({1}, {2,4,3}, {4,3,5})
g. Grafo 4: 1 componentes: ({1,5,3}, {2,3,4}, {1,5,4,2,3})

7.- Si un grafo no dirigido y conectado contiene un camino de Hamilton, éste es exactamente igual a su correspondiente camino de Euler.

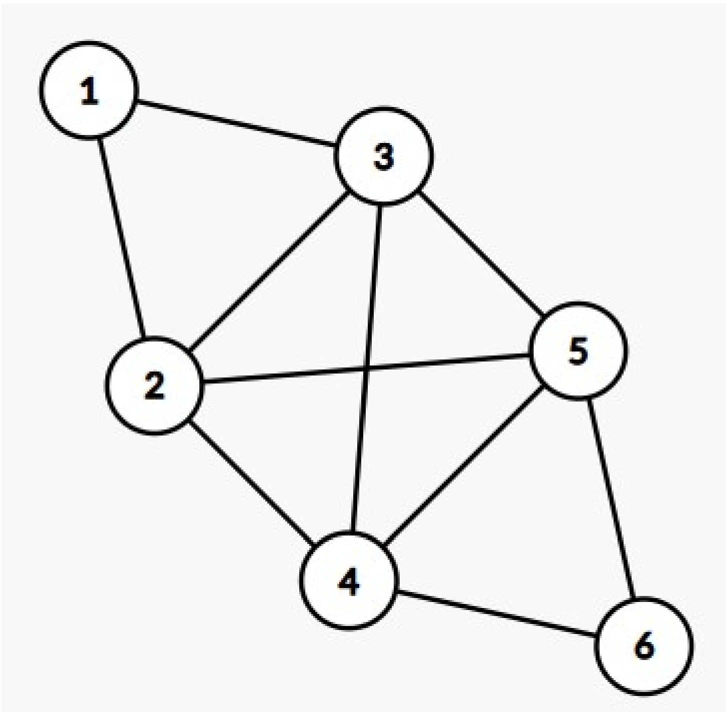


Imagen 6: Ejemplo de grafo no dirigido

La afirmación es falsa, pues si un grafo no dirigido y conectado contiene un camino de Hamilton, esto no implica que el camino de Hamilton sea exactamente igual a su camino de Euler. En el ejemplo planteado, existe el siguiente camino de Hamilton:

- 1,2 2,4 4,6 6,5 5,3

Para el camino, se visitan todos los vértices, pasando solamente una vez por cada uno. Sin embargo, en ese mismo camino no se visitan varias aristas, como la 4,5 o la 3,4 , por lo cual no hay camino de Euler. Para que exista un camino de Euler, el camino debe pasar por todas las aristas una sola vez.

8.- Un grafo dirigido de N vértices, con un vértice fuente y un vértice sumidero, puede estar fuertemente conectado.

La afirmación es falsa. El grafo puede contener componentes fuertemente conectados si tiene ciclos o más conexiones internas. Sin embargo, todo el grafo no puede ser fuertemente conectado. Esto se debe a que, para tener un grafo fuertemente conectado, debe ser posible viajar desde cualquier vértice del grafo a cualquier otro vértice y poder regresar al punto de origen. Cuando hay un vértice fuente, se puede salir de ese vértice pero no regresar ya que no hay ningún vértice que apunte a él. Si hay un vértice sumidero, se puede llegar a dicho vértice, pero no hay forma de hacer algún retorno.

9.- Sólo se puede definir camino(s) o circuito(s) de Euler en un grafo con un único componente conectado.

La afirmación es verdadera. Esto se debe a que si un grafo tiene varios componentes desconectados, no se puede pasar por todas las aristas del grafo. Para cumplir un camino o circuito de Euler, es necesario pasar por todas las aristas del grafo.

10.- La matriz de adyacencia de un grafo no dirigido es simétrica por la diagonal.

La afirmación es verdadera. Debido a que en los grafos no dirigidos no hay dirección de navegación, la matriz de adyacencia es simétrica por la diagonal.

Ejemplo:

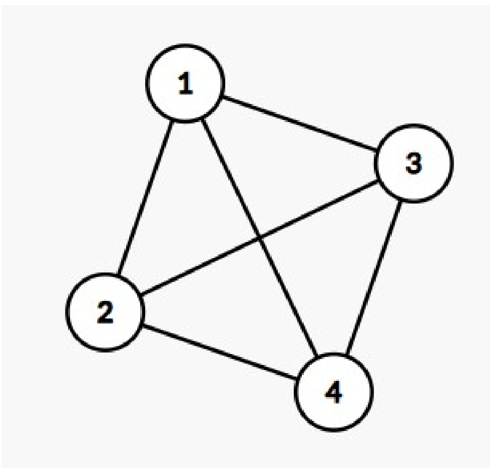


Imagen 7: Ejemplo de grafo no dirigido

La matriz de adyacencia del grafo es la siguiente:

Vértice	1	2	3	4
1	0	1	1	1
2	1	0	1	1
3	1	1	0	1
4	1	1	1	0

Tabla 1: Matriz de adyacencia de Imagen 7

Como se puede ver, el grafo ejemplo tiene una matriz simétrica ya que, para todos los M[i][j] = M[j][i].

11.- Un grafo dirigido está fuertemente conectado cuando existe un camino entre cada par de vértices, sin tener en cuenta las direcciones de las conexiones.

La afirmación es falsa. Esto se debe a que, para que un grafo está fuertemente conectado, cada vértice del grafo debe poder alcanzarse desde otro vértice, siguiendo las direcciones de las conexiones, pues en los grafos dirigidos se deben seguir las direcciones para poder navegar o recorrer el grafo.

12.- El algoritmo de Dijkstra genera un árbol de recubrimiento de costos mínimos, así como el algoritmo de Prim.

La afirmación es falsa. El algoritmo de Dijkstra genera un árbol de caminos más cortos. Sin embargo, no genera un árbol de recubrimiento de costos mínimos a diferencia del algoritmo de Prim, que sí lo hace.

13.- La matriz de caminos de un grafo con N vértices y M aristas se calcula sumando la matriz identidad de tamaño NxN con la matriz de adyacencia del grafo.

Una matriz de caminos muestra qué nodos son alcanzables (ya sea de forma directa o indirecta) desde otros nodos. De acuerdo a esto, la afirmación es falsa. Esto se debe a que, si se suma la matriz de identidad a la matriz de adyacencia del grafo, se obtienen las conexiones directas que hay entre los nodos, más sin embargo no encuentra todos los caminos posibles.

(Siguiente)
Ejemplo:

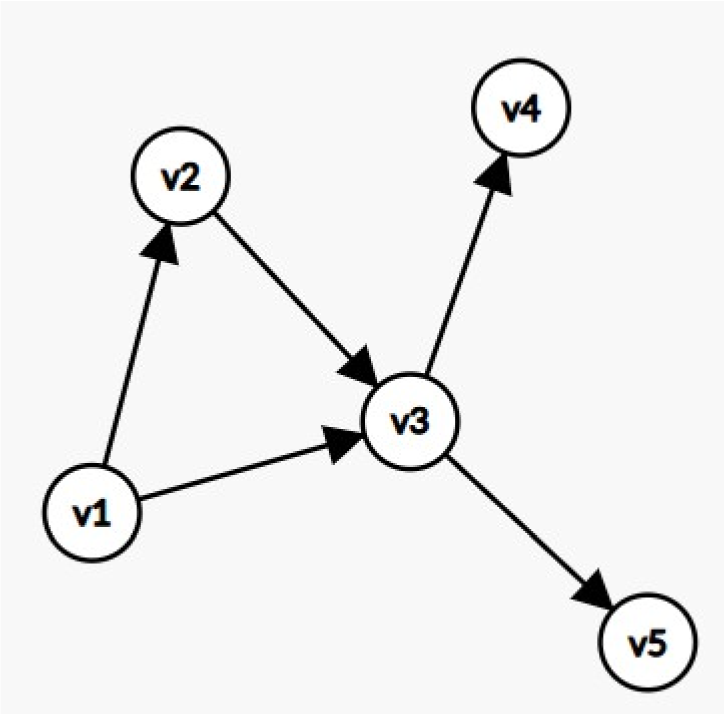


Imagen 8: Ejemplo de grafo dirigido

Vértice	v1	v2	v3	v4	v5
v1	0	1	1	0	0
v2	0	0	1	0	0
v3	0	0	0	1	1
v4	0	0	0	0	0
v5	0	0	0	0	0

Tabla 2: Matriz de adyacencia de Imagen 8

Al sumar la matriz de adyacencia con la matriz de identidad, se obtiene el siguiente resultado:

Vértice	v1	v2	v3	v4	v5
v1	1	1	1	0	0
v2	0	1	1	0	0
v3	0	0	1	1	1
v4	0	0	0	1	0
v5	0	0	0	0	1

Tabla 3: Suma de la matriz de adyacencia de Imagen 8 con su matriz de identidad

Al observar el grafo, se puede ver que, por ejemplo, existe un camino desde V1 para llegar a V5. Sin embargo, la tabla resultante de la suma de las matrices dice que no hay forma de llegar a V5 desde V1, pues M[V1][V5] = 0.

14.- Si la matriz de adyacencia de un grafo es una matriz diagonal inferior, se puede decir que el grafo es dirigido.

Una matriz diagonal inferior es una matriz en la cual todos los elementos por encima de la diagonal principal son cero. Para que un grafo sea no dirigido, su matriz debe ser una matriz simétrica. Por lo tanto, si un grafo tiene una matriz diagonal inferior, se puede decir que el grafo es dirigido y la afirmación es verdadera.

Ejemplo:

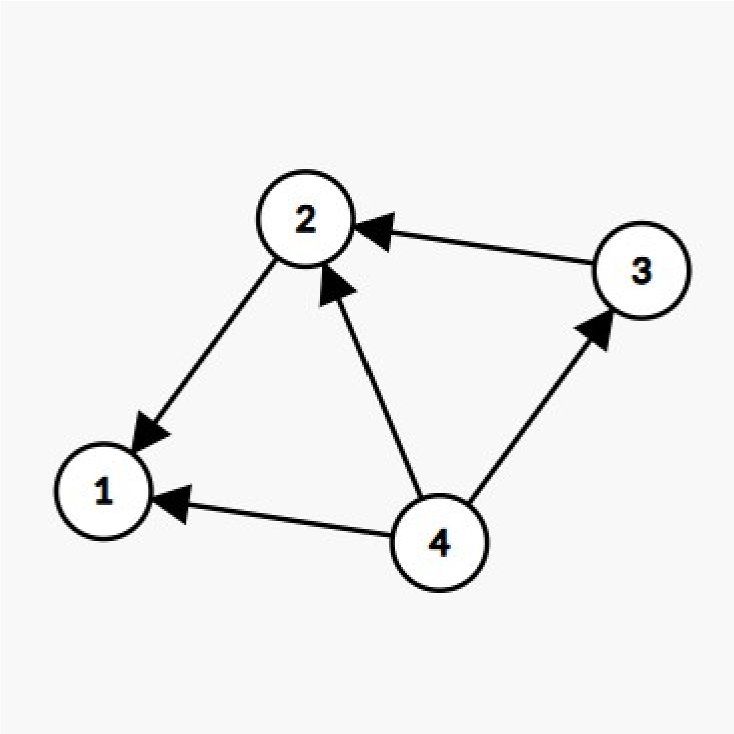


Imagen 9: Ejemplo de grafo dirigido

Vértice	1	2	3	4
1	0	0	0	0
2	1	0	0	0
3	0	1	0	0
4	1	1	1	0

Tabla 4: Matriz de adyacencia (y matriz diagonal inferior) de Imagen 9

En este caso, se puede ver una matriz de adyacencia inferior ya que todos sus elementos por encima de la diagonal son 0. Esta matriz crea el grafo dirigido de la Imagen 9.
TAD Grafo
Datos mínimos:

citas: lista de adyacencia, donde cada elemento representa un nodo (cita) y sus conexiones.
siguienteID: entero, indica el siguiente identificador disponible para un nuevo nodo.
Operaciones:

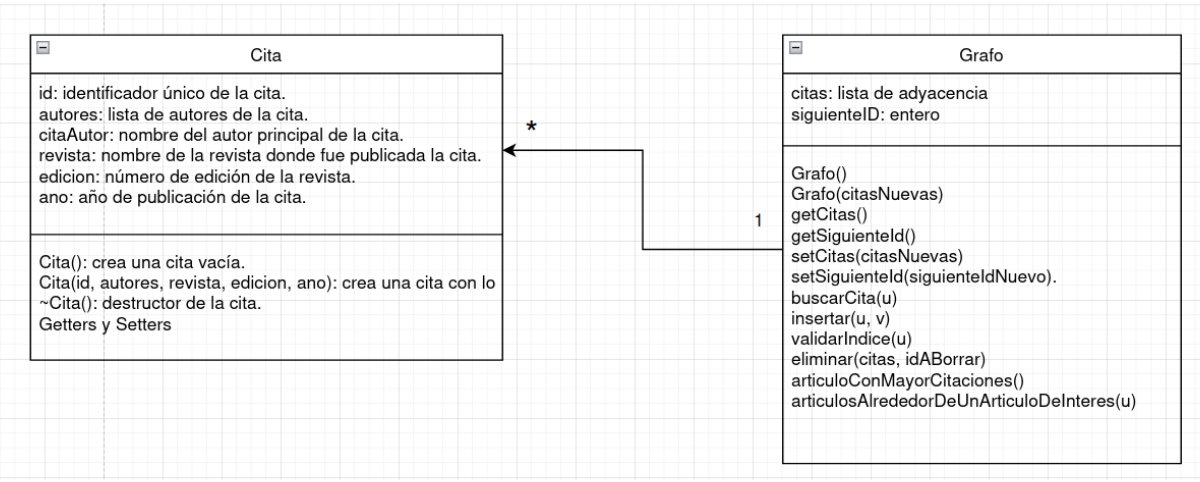
Grafo(): crea un grafo vacío.
Grafo(citasNuevas): crea un grafo con una lista de adyacencia inicial.
getCitas(): retorna la lista de adyacencia del grafo.
getSiguienteId(): retorna el siguiente identificador disponible.
setCitas(citasNuevas): asigna una nueva lista de adyacencia al grafo.
setSiguienteId(siguienteIdNuevo): asigna un nuevo valor al siguiente identificador.
buscarCita(u): retorna el índice de la cita u en la lista de adyacencia, o -1 si no existe.
insertar(u, v): inserta una arista dirigida desde el nodo u hacia el nodo v.
validarIndice(u): verifica si el nodo u existe en la lista de adyacencia; si no, lo agrega.
eliminar(citas, idABorrar): elimina todas las referencias al nodo con identificador idABorrar en la lista de adyacencia.
articuloConMayorCitaciones(): imprime la(s) cita(s) con mayor número de citaciones.
articulosAlrededorDeUnArticuloDeInteres(u): elimina simbólicamente el nodo u, vuelve el grafo no dirigido y encuentra los componentes conexos resultantes.
indiceDeReferenciacion(u): calcula e imprime el índice de referenciación del artículo u.
citacionesIndirectasDesdeUnArticulo(u): imprime la cantidad de citaciones indirectas desde el artículo u.

TAD Cita
Datos mínimos:

id: identificador único de la cita.
autores: lista de autores de la cita.
citaAutor: nombre del autor principal de la cita.
revista: nombre de la revista donde fue publicada la cita.
edicion: número de edición de la revista.
ano: año de publicación de la cita.
Operaciones:

Cita(): crea una cita vacía.
Cita(id, autores, revista, edicion, ano): crea una cita con los datos especificados.
~Cita(): destructor de la cita.
getId(): retorna el identificador de la cita.
getAutores(): retorna la lista de autores de la cita.
getCitaAutor(): retorna el nombre del autor principal.
getRevista(): retorna el nombre de la revista.
getEdicion(): retorna el número de edición.
getAno(): retorna el año de publicación.
setId(idNuevo): asigna un nuevo identificador a la cita.
setAutores(autoresNuevos): asigna una nueva lista de autores.
setCitaAutor(citaAutorNueva): asigna un nuevo autor principal.
setRevista(revistaNueva): asigna un nuevo nombre de revista.
setEdicion(edicionNueva): asigna un nuevo número de edición.
setAno(anoNuevo): asigna un nuevo año de publicación.

DIAGRAMA DE CLASES



Problema 1:

Para este problema tenemos que considerar que este grafo es un “digrafo”, también llamado grafo dirigido. Para esto podríamos pensar que es un nodo sumidero; sin embargo, puede darse el caso que no necesariamente sea el sumidero sino uno intermedio. Esto se puede explicar mejor en un grafo con la forma A→B→C→D, donde tanto el B, C, y D son citados una única vez. Para esto se cuenta en la lista de adyacencia las ocurrencias de este nodo (en este caso una cita) y se guardan todos los que sean iguales a la mayor cantidad.

Problema 2:
Este es el problema mas complejo dado que cuando nosotros eliminamos la cita, es importante tener en cuenta que no es una eliminación real de la lista de adyacencia. Mas bien nosotros establecemos esa fila en tamaño 1, y buscamos las ocurrencias de esa cita en las demás y las eliminamos. Luego nosotros vamos a recorrer la lista de adyacencia creando la arista inversa (En el caso que sea A→B se crea la B→A). Una vez tengamos esta nueva lista aplicamos un DFS o recorrido por profundidad, y cada vez que pare vamos a sumar 1 la cantidad de grupos. Luego lo volvemos aplicar y se repite el proceso hasta que todos los nodos han sido visitados una única vez.

Problema 3:
Para este problema se vuelve a recorrer la lista de adyacencia revisando para que nodo tiene como destino la cita que estamos buscando y almacenamos ese valor. Luego vamos a ver el tamaño de la lista asociada al indice de esa cita en cuestión. Una vez tengamos esos dos valores aplicamos la formula dada en el enunciado con (primer valor / (segundo valor * 0.5)).

Problema 4:
Para poder encontrar las citaciones indirectas, vamos a añadir a otro contenedor los indices guardados por la lista de adyacencia de la cita en cuestion. Posteriormente se procede a recorrer las listas de estos indices unicos. Cada vesz que se encuentre un indice por

primera vez se va a incluir en una lista de respuestas. Por ultimo se imprime el numero de articulos.