2. Representación de grafos

Representaciones

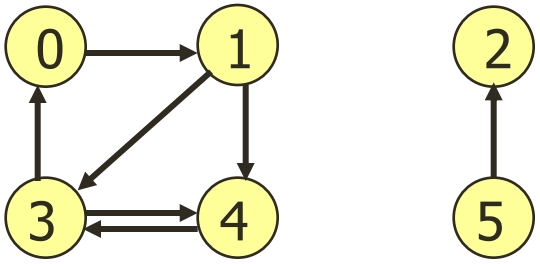
- Existen dos formas fundamentales de representar un grafo:
  - Si el grafo es ***disperso*** ($|A| <<< |V|^2$): listas de adyacencia
  - Si el grafo es ***denso*** ($|A| \approx |V|^2$): matriz de adyacencias

2. Representación de grafos

Matriz de adyacencias

- Un grafo $G = (V, A)$ se representa como una ***matriz*** de $|V| \times |V|$ elementos de tipo *boolean*
  - Si $(u, v) \in A \rightarrow G[u, v] = true$ (si no $G[u, v] = false$)
  - Coste espacial $O(|V|^2)$
  - Tiempo de acceso O(1)

Ejemplo:



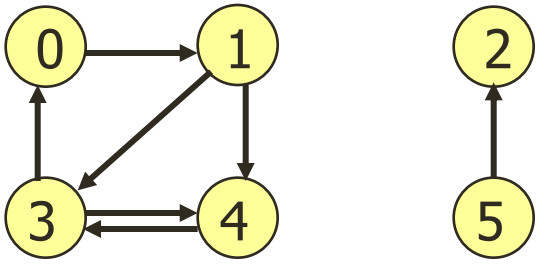
	0	1	2	3	4	5
0	false	true	false	false	false	false
1	false	false	false	true	true	false
2	false	false	false	false	false	false
3	true	false	false	false	true	false
4	false	false	false	true	false	false
5	false	false	true	false	false	false

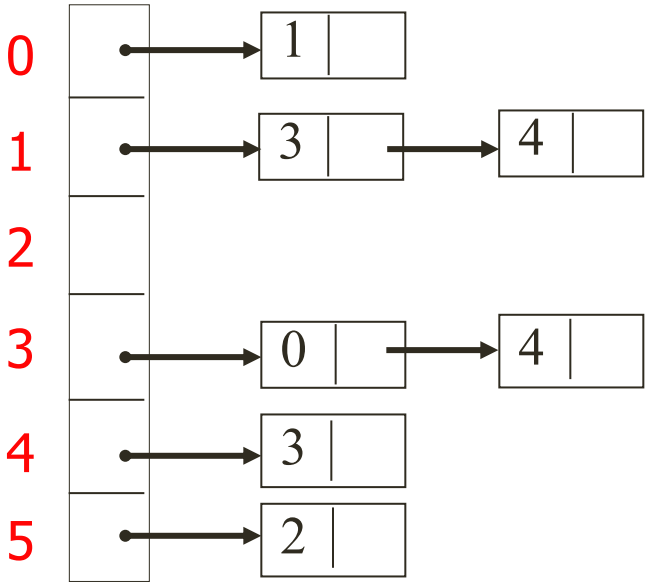
2. Representación de grafos

Listas de adyacencia

- Un grafo $G = (V, A)$ se representa como un ***array*** de $|V|$ ***listas*** de vértices
  - $G[v]$, $v \in V$, es la lista de los vértices adyacentes a v
  - Coste espacial $O(|V| + |A|)$
  - Tiempo de acceso O(grado de G) / O(grado de salida de G) si dirigido

Ejemplo:





2. Representación de grafos

Representaciones

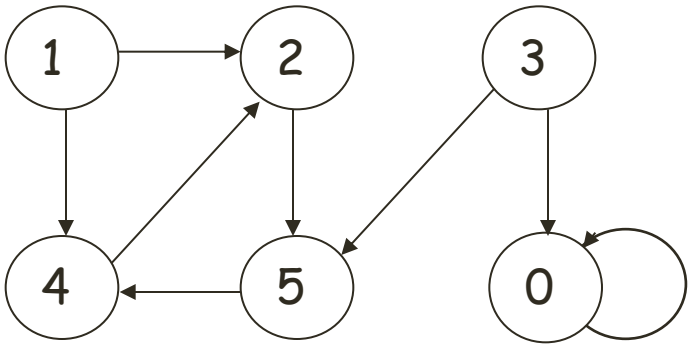
- Existen dos formas fundamentales de representar un grafo:
  - Si el grafo es ***disperso*** ($|A| <<< |V|^2$): listas de adyacencia
  - Si el grafo es ***denso*** ($|A| \approx |V|^2$): matriz de adyacencias

2. Representación de grafos

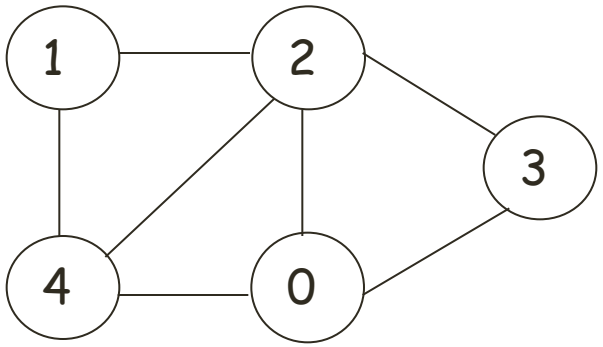
Ejercicios

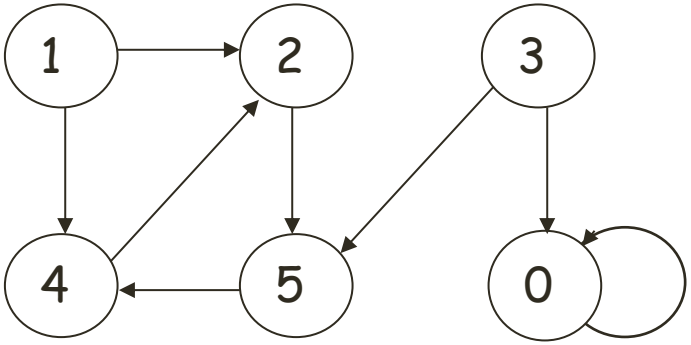
Ejercicio. Representad los siguientes grafos mediante una matriz de adyacencia y mediante listas de adyacencia.

a)



b)

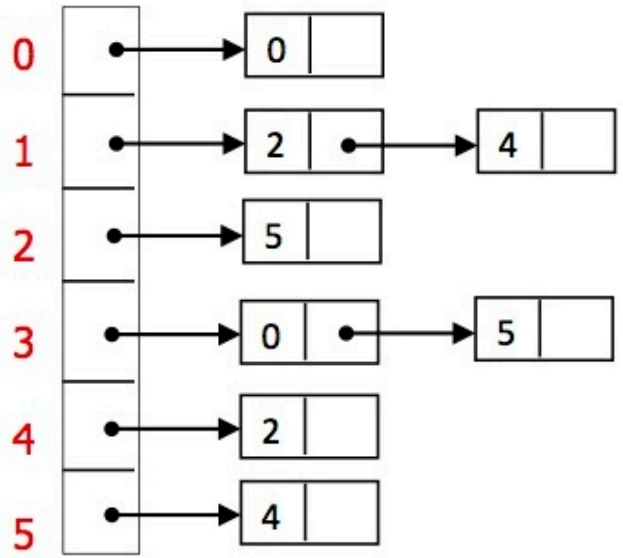


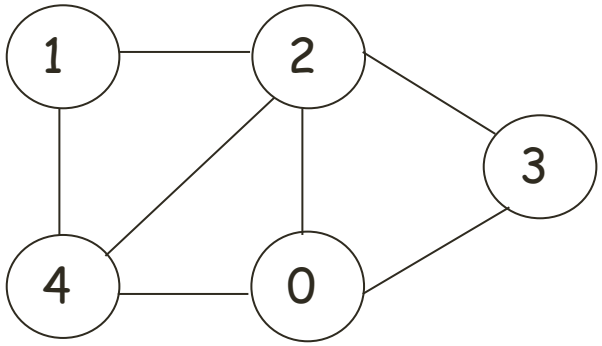


SOLUCIÓN:

a)

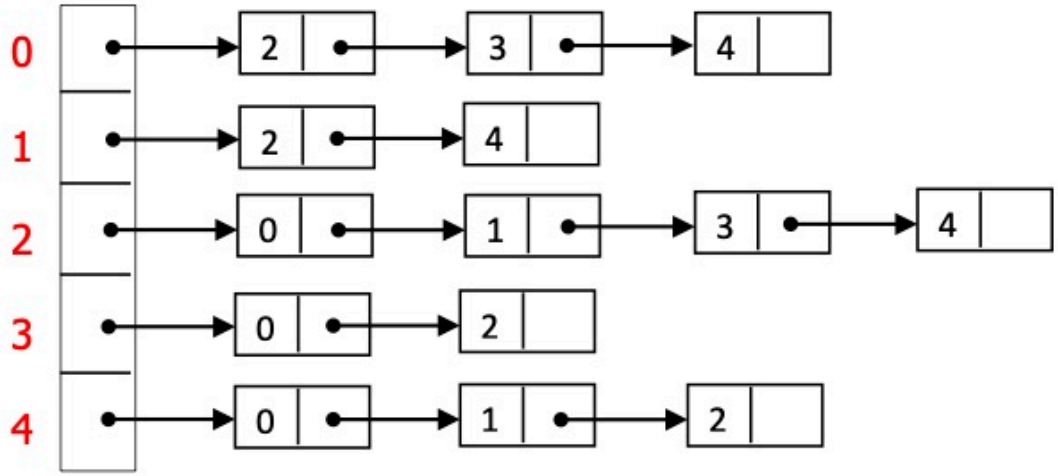
	0	1	2	3	4	5
0	true	false	false	false	false	false
1	false	false	true	false	true	false
2	false	false	false	false	false	true
3	true	false	false	false	false	true
4	false	false	true	false	false	false
5	false	false	false	false	true	false





b)

	0	1	2	3	4
0	false	false	true	true	true
1	false	false	true	false	true
2	true	true	false	true	true
3	true	false	true	false	false
4	true	true	true	false	false



2. Representación de grafos

Funcionalidad básica de un grafo

- Vamos a crear la clase abstracta *Grafo* para que defina la funcionalidad básica de un grafo
  - No utilizamos una *interfaz* ya que escribiremos el código de algunos métodos, como los recorridos, que son independientes de la implementación utilizada y del tipo de grafo
- La funcionalidad básica incluye:
  - Modificadores: inserción de aristas (con o sin pesos)
  - Consultores: número de vértices/aristas, búsqueda de aristas
  - Recorridos: en profundidad y en anchura

2. Representación de grafos

La clase Grafo: consultores

```
public abstract class Grafo {
  // Devuelve el número de vértices del grafo
  public abstract int numVertices();

  // Devuelve el número de aristas del grafo
  public abstract int numAristas();

  // Comprueba la existencia de la arista (i,j)
  public abstract boolean existeArista(int i, int j);

  // Recupera el peso de la arista (i,j)
  public abstract double pesoArista(int i, int j);

  // Devuelve una lista con los adyacentes del vértice i
  public abstract ListaConPI<Adyacente> adyacentesDe(int i);
```

2. Representación de grafos

La clase Grafo: modificadores

```
public abstract class Grafo {
  ...

  // Añade la arista (i,j) a un grafo sin pesos
  public abstract void insertarArista(int i, int j);

  // Añade la arista (i,j) con peso p a un grafo ponderado
  public abstract void insertarArista(int i, int j,
                                      double p);
```

- El método para insertar aristas está sobrecargado para permitir la inserción de aristas tanto en un grafo sin pesos como en uno ponderado