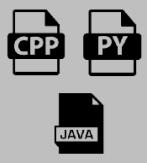
Taller en Sala Nro. 1
Implementación de Grafos

En la vida real, los grafos se utilizan en videojuegos para representar las rutas que pueden tomar los caracteres, como por ejemplo League of Legends y World of Warcraft, así https://goo.gl/images/R8WyGP

 **Trabajo en Parejas**	 **Hoy, plazo máximo de entrega**	 **Docente entrega código suelto en GitHub**
 **Sí .cpp, .py o .java**	 **No .zip, .txt, html o .doc**	 **Alumnos entregan código suelto por GitHub**

Ejercicios a resolver

1. Extiendan la clase abstracta *Digraph,* llámenla *DigraphAM* e implementen grafos con la estructura de datos Matrices de Adyacencia Etiquetadas

2. Creen la clase *DigraphAL* e implementen grafos con la estructura de datos Listas de Adyacencia. Al igual que *DigraphAM*, *DigraphAL* también extiende la clase abstracta *Digraph*.

3. [Ejercicio Opcional] Generen una representación usando Graphviz de los grafos anteriores.

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

Ayudas para resolver los Ejercicios

Ayudas para el Ejercicio 1

Pista: Asuman que los identificadores de los nodos son enteros positivos que caben en un *int e inician en 0*. Y:

- ☑ Los pesos de los grafos son enteros que caben en un *int*.
- ☑ En los grafos sin peso todas las aristas tienen peso 1.
- ☑ Una arista de peso 0 significa que no hay conexión (es decir, no es una arista)

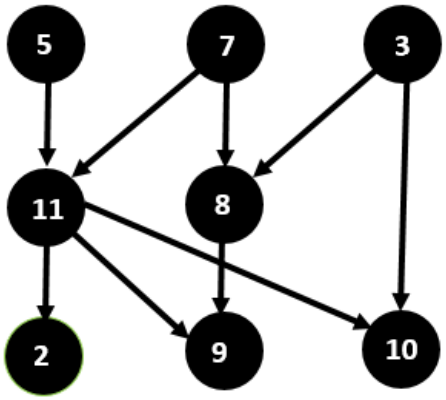
Pista: Vean en Guía de Laboratorios, numeral 4.10, *"Cómo hacer clases abstractas"*

Error Común 1: El método *getSuccessors* debe retornar los identificadores de los vértices, no los pesos de los arcos, ordenados de menor a mayor. Si no hay ninguno, retorne *null*. Utilice *Collections.sort(a)* para ordenar un *ArrayList<Integer> a*.

Error Común 2: El método *getSucessors* retorna los vecinos de un nodo, es decir, los nodos adyacentes. No es un recorrido en profundidad.

Como un ejemplo, para el grafo de la imagen, esta debe ser la matriz:

Grafo



DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

Matriz

	0	1	2	3	4	5	6	7	8	9	10	11
0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	0	1	0
4	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	1	0	0	1
8	0	0	0	0	0	0	0	0	0	1	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	1	0	0	0	0	0	0	1	1	0

Ayudas para el Ejercicio 2

Error Común 1: El método *getSuccessors* debe retornar los identificadores de los vértices, no los pesos de los arcos.

Error Común 2: El método *getSucessors* retorna los vecinos de un nodo, es decir, los nodos adyacentes. No es un recorrido en profundidad.

Como un ejemplo, para el grafo de la imagen del punto 1, esta debe ser la representación con listas:

0→
1→
2→
3→ 8, 10
4→
5→ 11
6→
7→ 8, 11
8→ 9
9→
10→
11→ 2, 9, 10

La implementación de los puntos 3-6 se hacen en la clase “Recorridos.java”.

Ayudas para el Ejercicio 3 [Opcional]

Pista: Para visualizar el grafo, utilice **http://www.webgraphviz.com/**

Como un ejemplo para grafo no dirigido:

```
graph {
  a -- b;
  a -- c;
  a -- e;
  b -- c;
  c -- d;
  c -- e;
}
```

Como un ejemplo, para grafo dirigido:

```
digraph G {
```

```
    "Welcome" -> "To" [label = "a"]
    "To" -> "Web"
    "To" -> "GraphViz!"

}
```

Error Común:

¿Alguna inquietud?

CONTACTO

Docente Mauricio Toro Bermúdez
Teléfono: (+57) (4) 261 95 00 **Ext.** 9473
Correo: mtorobe@eafit.edu.co
Oficina: 19- 627

Agende una cita con él a través de **http://bit.ly/2gzVg10** , en la pestaña *Semana. Si no da clic en esta pestaña, parecerá que toda la agenda estará ocupada.*