

Laboratorio A.E.D. Ejercicio Individual 6

Guillermo Román

guillermo.roman@upm.es

Lars-Åke Fredlund

lfredlund@fi.upm.es

Manuel Carro

mcarro@fi.upm.es

Marina Álvarez

marina.alvarez@upm.es

Julio García

juliomanuel.garcia@upm.es

Tonghong Li

tonghong@fi.upm.es

Sergio Paraiso

sergio.paraiso@upm.es

Juan José Moreno

juanjose.moreno@upm.es

Luis Miguel Danielsson

lm.danielsson@alumnos.upm.es

Normas

- ▶ **La entrega del ejercicio es individual**
- ▶ Fechas de entrega y la penalización aplicada a la puntuación obtenida sobre 10:

Hasta el Jueves 9 de Diciembre, 12:00 horas	0 %
Hasta el Viernes 10 Diciembre, 12:00 horas	20 %
Hasta el Viernes 10 Diciembre, 23:59 horas	40 %
- ▶ Después la puntuación máxima será 0
- ▶ Se comprobará plagio y se actuará sobre los detectados.

Entrega

- ▶ Todos los ejercicios de laboratorio se deben entregar a través de

`http://vps142.cesvima.upm.es`

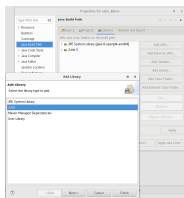
- ▶ El fichero que hay que subir es `Suma.java`.

Configuración previa

- ▶ Arrancad Eclipse
- ▶ Si trabajáis en portátil, podéis utilizar cualquier versión reciente de Eclipse. Es suficiente con que instaléis la *Eclipse IDE for Java Developers*.
- ▶ Cambiad a “Java Perspective”.
- ▶ Debéis tener instalado al menos Java JDK 8.
- ▶ Cread un proyecto Java llamado aed:
 - ▶ Seleccionad separación de directorios de fuentes y binarios.
 - ▶ **No debéis elegir la opción de crear el fichero**
`module-info.java`
- ▶ Cread un *package* `aed.individual6` en el proyecto aed, dentro de `src`
- ▶ Aula Virtual → AED → Laboratorios y Entregas Individuales → Individual 6 → Individual6.zip; descomprimidlo
- ▶ Contenido de Individual6.zip:
 - ▶ `Suma.java`, `TesterInd6.java`

Configuración previa

- ▶ Añadid al proyecto aed la librería JUnit 5



- ▶ Project → Properties → Java Build Path. Se abrirá una ventana como la de la izquierda;
- ▶ Usad la opción “Add Library...” → Seleccionad “JUnit” → Seleccionad “JUnit 5”
- ▶ Si vuestra instalacion distingue ModulePath y ClassPath, instalad en ClassPath
- ▶ En la clase TesterInd6 tenéis las pruebas, para ejecutarlas, abrid el fichero TesterInd6, pulsando el botón derecho sobre el editor, seleccionar “Run as...” → “JUnit Test”
- ▶ NOTA: Si al ejecutar, no aparece la vista “JUnit”, podéis incluirla en “Window” → “Show View” → “Java” → “JUnit”

Documentación de la librería aedlib.jar

- ▶ La documentación de la API de aedlib.jar está disponible en
<http://costa.ls.fi.upm.es/entrega/aed/docs/aedlib/>
- ▶ También se puede añadir la documentación de la librería a Eclipse (*no es obligatorio*):
 - ▶ En el “Package Explorer”: “Referenced Libraries” → aedlib.jar y elige la opción “Properties”. Se abre una ventana donde se puede elegir “Javadoc Location” y ahí se pone como “javadoc location path:”

<http://costa.ls.fi.upm.es/entrega/aed/docs/aedlib/>
y presionar el boton “Apply and Close”

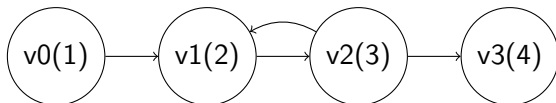
Tarea: implementar el método `sumVertices`

```
static <E> Map<Vertex<Integer>,Integer>  
    sumVertices(DirectedGraph<Integer,E> g)
```

Dado un grafo dirigido G , cuyos vértices contienen elementos de tipo `Integer`, el método devuelve un objeto de tipo `Map<Vertex<Integer>,Integer>` donde:

- ▶ las claves serán todos los vértices del grafo
- ▶ el valor asociado a cada clave (es decir, a cada vértice) v será la suma de los elementos – que son de tipo `Integer` – de los vértices alcanzables (“reachable”) desde v
- ▶ **notad** que el método no debe cambiar el grafo; se comprueba que los elementos de los vértices y las aristas no cambian.
- ▶ no es permitido añadir nuevos atributos.

Ejemplo



- ▶ En la figura, el vértice $v_0(1)$ contiene el elemento 1, el vértice v_1 contiene 2, el vértice v_2 contiene 3, y v_3 contiene 4.
- ▶ El map devuelto por el método debe contener:

v_0	10
v_1	9
v_2	9
v_3	4

- ▶ v_0 , v_1 , v_2 y v_3 son alcanzables desde v_0
- ▶ sólo v_1 , v_2 y v_3 son alcanzables desde v_1 y v_2
- ▶ sólo v_3 es alcanzable desde v_3 .

El Tester

- ▶ Notad que para ayudarlos, el tester imprime los grafos en un formato sencillo, mostrando todos los vértices y los nodos a los que llegan las aristas que salen de cada uno.
- ▶ Por ejemplo, el grafo de la figura anterior se imprime de la siguiente forma:

```
v0(1): -->v1
v1(2): -->v2
v2(3): -->v1, -->v3
v3(4):
```

- ▶ El proyecto debe compilar sin errores y debe cumplirse la especificación de los métodos a completar.
- ▶ Debe ejecutar `TesterInd6` correctamente y sin mensajes de error
 - ▶ Nota: una ejecución sin mensajes de error no significa que el método sea correcto (es decir, que funcione bien para cada posible entrada).