Laboratorio A.E.D. Viernes 13:00 - 15:00 y 15:00 - 17:00

Guillermo Vigueras
guillermo.vigueras@imdea.org
Julio García
juliomanuel.garcia@upm.es
Lars-Åke Fredlund
lfredlund@fi.upm.es
Manuel Carro Liñares
mcarro@fi.upm.es
Marina Álvarez
marina.alvarez@upm.es
Tonghong Li
tonghong@fi.upm.es

Normas.

- **¡Solo debe entregar una persona por grupo!**.
- Fechas de entrega y nota máxima alcanzable:

Hasta el martes 24 de noviembre, 15:00 horas	10
Hasta el miércoles 25 de noviembre, 15:00 horas	8
Hasta el jueves 26 de noviembre, 13:00 horas	6
Hasta el viernes 27 de noviembre 13:00 horas	4

  Después la puntuación máxima será 0.
- Se comprobará plagio y se actuará sobre los detectados.
- Usad las horas de tutoría para preguntar sobre programación – son oportunidades excelentes para aprender.

Sistema de Entrega

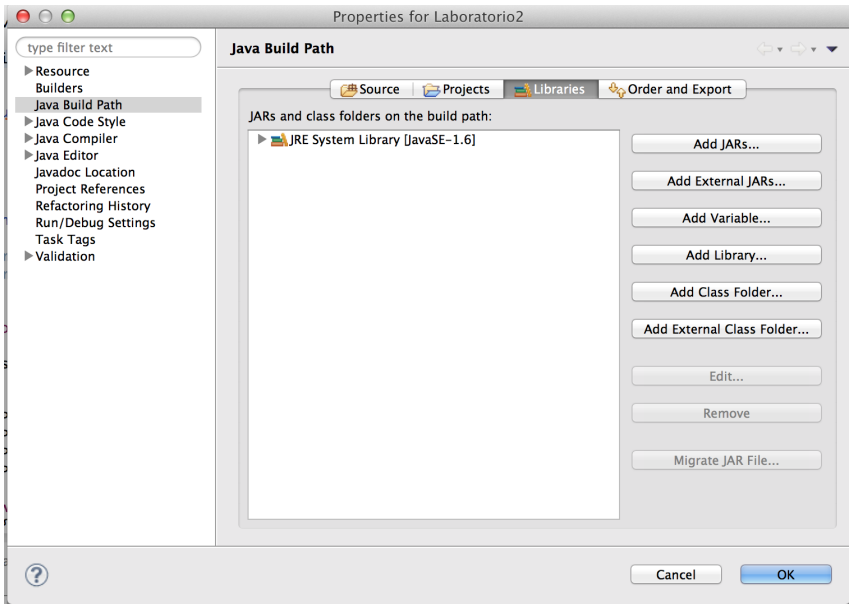
- Todas los ejercicios de laboratorio se deben entregar a través de la web `http://lml.ls.fi.upm.es/~entrega`.
- Hoy, el fichero que hay que subir es `PositionListSet.java`.

Configuración previa al desarrollo del ejercicio.

- Arrancad Eclipse. Debéis tener un acceso directo.
- Si trabajáis en portátil, podéis utilizar cualquier versión relativamente reciente de Eclipse. Debería valer cualquier versión entre la versión 3.7 (Indigo) o 4.3 (Kepler). Es suficiente con que instaléis la *Eclipse IDE for Java Developers*.
- Cambiad a "Java Perspective".
- Cread un proyecto Java llamado `aed`:
  - Seleccionad separación de directorios de fuentes y binarios.
- Cread un *package* `sets` en el proyecto `aed`, dentro de `src`.
- Aula Virtual → AED → Sesiones de laboratorio → Laboratorio 7 → `Laboratorio7.zip`; descomprimidlo.
- Contenido de Laboratorio7.zip:
  - `Tester.java`, `Set.java` y `PositionListSet.java`
  - `positionList.jar`

Configuración previa al desarrollo del ejercicio.

- Importad al paquete `sets` las fuentes que habéis descargado (`Tester.java`, `Set.java` y `PositionListSet.java`).
- Añadid al proyecto `aed` la librería `positionList.jar` que habéis descargado. Para ello:
- Project → Properties. Se abrirá una ventana como esta:



- Java Build Path → Libraries → Add external JARs → Seleccionad el fichero `positionList.jar` que os habéis descargado.
- Intenta a compiler el `Tester` y los otros ficheros. Veréis que `PositionListSet.java` contiene errores.

Tarea para hoy

- Implementar un estructura de datos “conjunto” especificado por la interfaz `Set<E>`, usando un `PositionList`
- Un conjunto (`https://es.wikipedia.org/wiki/Conjunto`) es una estructura de datos donde los elementos no son repetidos
- El fichero `Set.java` enumera los métodos que deberíais implementar en el fichero `PositionList.java`; solo hay que modificar el fichero `PositionList.java`

The Set<E> interface

```
package sets;

interface Set<E> extends Iterable<E> {
    boolean isEmpty();    // es vacio?
    int size();           // tamano

    boolean add(E elem);         // anade un elemento elem
    boolean remove(E elem);      // borra un elemento elem
    boolean contains(Object o); // contiene el conjunto o?

    Set<E> union(Set<E> set);         // union de conjuntos
    Set<E> intersection(Set<E> set); // interseccion
}
```

Ejemplo

```
Set<Integer> s  =  new PositionListSet<Integer>();
s.isEmpty();      ===>  true
s.size();         ===>  0
s.add(0);         ===>  true (0 fue anadido)
s.add(0);         ===>  false (0 ya esta en el conjunto)
s.remove(0);      ===>  true (se borro 0)
s.contains(0);    ===>  false (0 fue borrado)

Set<Integer> t = new PositionListSet<Integer>();
s.add(2);
t.add(1);
Set<Integer> u = s.union(t);          ==> nuevo conjunto
Set<Integer> v = u.intersection(t);   ==> nuevo conjunto

u.contains(1);       ===>  true (como esta en t)
u.contains(2);       ===>  true (como esta en s)
v.contains(1);       ===>  true (1 esta en ambos u y t)
v.contains(2);       ===>  false (2 no esta en t)
```

Importante

- `PositionListSet` deberia ser capaz de guardar `null`:

```
Set<Integer> s  =  new PositionListSet<Integer>();
s.contains(null);   ===>  false
s.add(null)         ===>  true
s.contains(null);   ===>  true
```

- Para poder implementar la interfaz `Set<E>` es necesario implementar un iterador sobre conjuntos. Lo mas fácil es reusar el iterador de `PositionLists`:

```
public Iterator<E> iterator() {
  return elements.iterator();
}
```

- Los métodos `union` y `intersection` devuelven un conjunto nuevo, y no cambian sus argumentos.

Notas

- El proyecto debe compilar sin errores y debe cumplirse la especificación de los métodos a completar.
- Debe ejecutar `Tester` correctamente 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).
- Todos los ejercicios se comprueban manualmente antes de dar la nota final.