Entrega Individual 3

Guillermo Román
groman@fi.upm.es
Julio García
juliomanuel.garcia@upm.es
Lars-Åke Fredlund
lfredlund@fi.upm.es
Manuel Carro
mcarro@fi.upm.es
Marina Álvarez
marina.alvarez@upm.es
Raúl Correal
raul.correal@upm.es
Tonghong Li
tonghong@fi.upm.es

Normas

- **La entrega del ejercicio es individual**
- Fechas de entrega y nota máxima alcanzable:

Hasta el Lunes 3 de Octubre, 23:59 horas	10
Hasta el Martes 4 de Octubre, 23:59 horas	8
Hasta el Miércoles 5 de Octubre, 23:59 horas	6

- Después la máxima puntuación será 0
- Se comprobará plagio y se actuará sobre los detectados

Sistema de Entrega

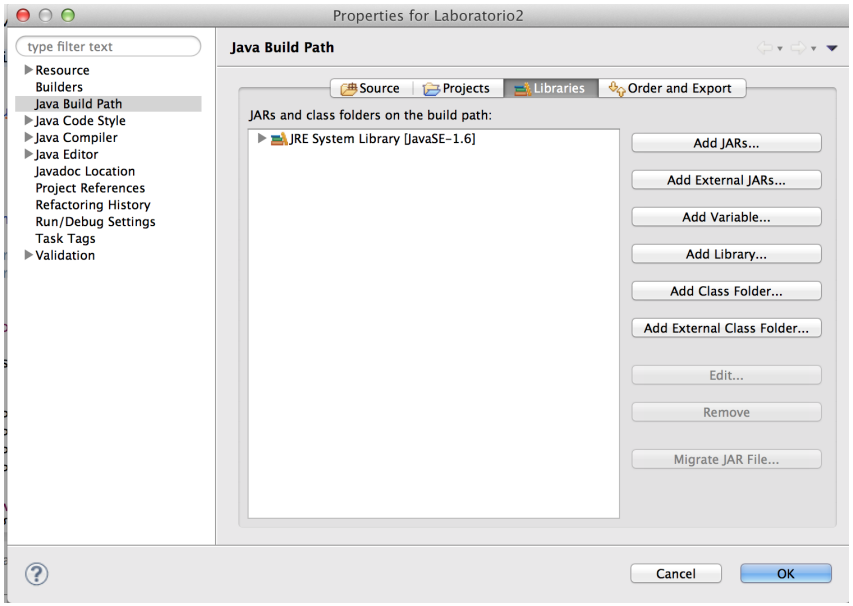
- Todas los ejercicios de laboratorio se deben entregar a través de la web `http://lml.ls.fi.upm.es/~entrega`
- Los ficheros a subir son `OperacionCompactar.java`

Configuración previa al desarrollo del ejercicio.

- Arrancad Eclipse. Es suficiente con que tengáis 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* `aed.invididual3` en el proyecto `aed`, dentro de `src`
- Aula Virtual → AED → Sesiones de laboratorio → Entrega Individual 3 → `EntregaIndividual3.zip`; descomprimidlo
- Contenido de EntregaIndividual3.zip
  - `Tester.java`, `OperacionCompactar.java`
- Descargad también el fichero `positionList.jar`

Configuración previa al desarrollo del ejercicio.

- Importad al paquete `aed.invididual3` los fuentes que habéis descargado (`Tester.java`, `OperacionCompactar.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
- Ejecutad `Tester`. Veréis que imprimen un mensaje de error

Tarea para hoy: Entrega Individual 3

- Realizar en `OperacionCompactar` una implementación del método:

```
public <E> PositionList<E> compactar (PositionList<E> lista)
```

- Recibe una lista genérica de tipo `PositionList` y compacta los elementos consecutivos iguales a una única aparición
  - Debe devolver una *nueva* lista cuya longitud sea el número de elementos después de compactarlos (no puede ser mayor)
  - Si el parámetro `lista` es `null` se lanzará una la excepción `IllegalArgumentException`
  - Los elementos contenidos en `lista` podrán ser `null`

`compactar({2,2,2,2})` $\rightsquigarrow$ `{2}`

`compactar({2,2,null,null,2,3})` $\rightsquigarrow$ `{2,null,2,3}`

`compactar({2,2,1,2})` $\rightsquigarrow$ `{2,1,2}`

`compactar({})` $\rightsquigarrow$ `{}`

`compactar({1,2,3,null,null})` $\rightsquigarrow$ `{1,2,3,null}`

`compactar({null})` $\rightsquigarrow$ `{null}`

`compactar(null)` $\rightsquigarrow$ `IllegalArgumentException`

Comentarios generales

- Para crear una nueva lista podéis usar la clase `NodePositionList<E>`, que implementa el interfaz `PositionList<E>`
- La clase `NodePositionList<E>` dispone de un constructor sin parámetros para crear una lista vacía
- 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