

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

Guillermo Viguera

guillermo.viguera@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

Pablo Nogueira

pnogueira@fi.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 lunes 5 de octubre 13:00 (15:00) horas	10
Hasta el martes 6 de octubre, 13:00 (15:00) horas	8
Hasta el miércoles 7 de octubre, 13:00 (15:00) horas	6
Hasta el jueves 8 de octubre, 13:00 (15:00) horas	4

Después la máxima puntuación será 0.
- ▶ Explicamos la solución tras jueves 8 de octubre.
- ▶ Se comprobará plagio y se actuará sobre los detectados.
- ▶ Usad las tutorías.
- ▶ Usad las horas de tutoría para preguntar sobre programación – son oportunidades excelentes para aprender.

Sistema de Entrega

- ▶ Todos los ejercicios de laboratorio se deben entregar a través de la web <http://lml.ls.fi.upm.es/~entrega>.
- ▶ Fichero(s) a subir en la sesión de hoy: `TimeHMS.java` y `TimeSec.java` .

Configuración previa al desarrollo del ejercicio.

- ▶ Arrancad Eclipse. Debéis tener un acceso directo.
- ▶ Si trabajáis en portátil, podeis utilizar cualquier version relativamente reciente de Eclipse. Debería valer cualquier versión entre la versión 3.7 (Indigo) o 4.3 (Kepler). Es suficiente con que instaleis 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* `TimeInterface` en el proyecto aed, dentro de `src`.
- ▶ Aula Virtual → AED → Sesiones de laboratorio → Laboratorio2 → Laboratorio2.zip; descomprimidlo.
- ▶ Contenido de Laboratorio2.zip
 - ▶ `Tester.java`, `Time.java`, `TimeHMS.java` y `TimeSec.java`

Configuración previa al desarrollo del ejercicio.

- ▶ Importad al paquete `TimeInterface` las fuentes que habéis descargado (`Tester.java`, `Time.java`, `TimeHMS.java` y `TimeSec.java`).
- ▶ Ejecutad `Tester`. Veréis que imprime un mensaje de error.

Tarea para hoy

- ▶ En el laboratorio de hoy vamos a hacer uso de las interfaces Java
- ▶ Para ello vamos a utilizar dos distintas representaciones de tiempo:
 - ▶ Una utiliza horas, minutos y segundos (HH:MM:SS)
 - ▶ La otra expresa el tiempo en segundos
- ▶ Algunos ejemplos de tiempo representados de ambos modos:

HH:MM:SS	Segundos
+ 37:01:31	133291
- 46:34:21	-167661

- ▶ En el formato HH:MM:SS:
 - ▶ Los segundos y minutos están en el rango [0 – 59]
 - ▶ Las horas están en el rango [0 – MAX.INT]

Tarea para hoy

- ▶ Para poder utilizar ambas representaciones de tiempo se han definido dos clases. Cada una tiene sus miembros particulares
- ▶ La clase TimeHMS representa el tiempo en HH:MM:SS (miembros hour, mins y secs)
- ▶ La clase TimeSec representa el tiempo en segundos (miembro totalSecs)
- ▶ Ambas tienen signo (miembro sign). Si es negativo sign=-1, si es positivo sign=1

```
public class TimeHMS
    implements Time {

    private int hour;
    private int mins;
    private int secs;
    private int sign;

    ...
}

public class TimeSec
    implements Time {

    private int totalSecs;
    private int sign;

    ...
}
```

Tarea para hoy

- ▶ Las clases TimeHMS y TimeSec implementan a la interfaz Time
- ▶ De este modo se pueden definir operaciones entre objetos de cada clase aunque tengan distinta representación (e.g. suma)

```
public interface Time{
    public int getHour(); // Devuelve las horas (sin signo)
    public int getMins(); // Devuelve los minutos (sin signo)
    public int getSecs(); // Devuelve los segundos (sin signo)
    public int getSign(); // Devuelve el signo

    public int timeInSeconds(); // Devuelve el tiempo en
                                // segundos (con signo)

    public Time suma(Time t); // Suma de horas
    public Time resta(Time t); // Resta de horas
}
```

Tarea para hoy

- ▶ Tenéis que implementar en las clases TimeHMS y TimeSec:
 - ▶ Los métodos de la interfaz Time
 - ▶ El método equals
- ▶ Veamos algunas salidas de los métodos para ambas representaciones cuando tienen el mismo valor

	t.getHour()	t.getMins()	t.getSecs()	t.getSign()
Time t=-46:34:21	46	34	21	-1
Time t=-167661	46	34	21	-1

- ▶ La salida de los métodos cuando tienen distinto valor

	t.getHour()	t.getMins()	t.getSecs()	t.getSign()
Time t=-35:52:07	35	52	07	-1
Time t=4863	01	21	03	1

Tarea para hoy

- ▶ Veamos algunas salidas del resto de métodos a implementar

	<code>t.timeInSeconds()</code>	<code>t.suma(-35:52:07)</code>	<code>t.suma(4863)</code>
Time t=-35:52:07	-129127	-71:44:14	-34:31:04
Time t=4863	4863	-124264	9726

	<code>t.resta(-35:52:07)</code>	<code>t.resta(4863)</code>
Time t=-35:52:07	00:00:00	-37:13:10
Time t=4863	133990	0

	<code>t.equals(-35:52:07)</code>	<code>t.equals(4863)</code>
Time t=-35:52:07	true	false
Time t=4863	false	true
Time t=-129127	true	false

Tarea para hoy

- ▶ 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.