


FINAL ORDINARIA : TRIMESTRE 1 (1H 30') TRIMESTRE 2 (130')	26/06/2020 - 16:00
Nombre y Apellidos:	DNI/NIE: Firma:
1º Desarrollo de Aplicaciones Web (Vespertino) Módulo: Entornos de Desarrollo	 IES Alonso de Avellaneda (Alcalá)

Se realizará la Parte del Trimestre 1 en primer lugar. Al finalizar se hará un descanso de 10 min para luego continuar con la Parte del Trimestre 2.

Es necesario obtener una calificación igual o mayor que 5 en ambos trimestres para poder superar el módulo. En caso contrario, se iría a la prueba final extraordinaria

Los alumnos que tengan pendiente sólo un trimestre deberán obtener una nota igual o mayor que 5 para poder superar el módulo, en caso contrario, deberán presentarse a la prueba final extraordinaria.

Si el/los ejercicio/s resultan plagio de recursos de Internet o de otras pruebas será invalidado

Trimestre 1

Ejercicios (10 puntos) (Recogida de los fuentes del programa en repositorio github de cada alumno)

Todos los ficheros necesarios están en `/home/alumnot/Descargas`.

1. Escribe los comandos y muestra las salidas para instalar el JDK con extensión `.rar` en `/usr/lib/jvm`. Muestra el contenido de dicha carpeta al finalizar (1p)
2. Realiza los comandos y muestra las salidas para descomprimir, instalar y ejecutar Eclipse desde terminal (1p)
3. Configurar las variables de entorno de Java. Muestra con comandos la salida del fichero que modificaste y la versión de Java (1p)
4. Instala el plugin de Umllet en Eclipse y crea un pequeño diagrama de clases que establezca herencia entre dos clases ejemplo Clase1 y Clase2. Desinstala el plugin (1p).
5. Suponemos que tenemos una clase HolaMundo.java (1p):

```
public class HolaMundo { // Clase principal
    public static void main(String[] args) {
        System.out.println(";Hola mundo!");
    }
}
```

Se ha generado el siguiente código en consola. Explica qué tipo de código es, define qué significa y qué comando se utiliza para generarlo. (1p)

```
public class HolaMundo {
    public HolaMundo();
    Code:
```

```

    0: aload_0
    1: invokespecial #1                      // Method java/lang/Object."
<init>":()V
    4: return

    public static void main(java.lang.String[]);
    Code:
        0: getstatic     #2                      // Field
java/lang/System.out:Ljava/io/PrintStream;
        3: ldc          #3                      // String ¡Hola mundo!
        5: invokevirtual #4                      // Method
java/io/PrintStream.println:(Ljava/lang/String;)V
        8: return
}

```

6. Dados el siguiente árbol de directorios y clases, mediante comandos de terminal y de java, crear estructura de carpetas, compilación y ejecución desde el **terminal** respetando la jerarquía de paquetes. Entrega los comandos que realizas y sus salidas y los paquetes creados. así como el cambio que realices en las clases si fueran necesarios. (3p)

```

es
├── man
│   └── Test1.java
├── dev
│   └── Test2.java
└── admon
    └── Test3.java

public class Test1 {
    public void imprimir() {
        System.out.println("Test1 man");
    }
}

public class Test2 {
    public void imprimir(){
        System.out.println("Test2 dev");
    }
}

public class Test3 {

    public static void main(String[] args) {
        Test1 t1 = new Test1();
        Test2 t2 = new Test2();
        System.out.println("Prueba General de Test3");
        t1.imprimir();
        t2.imprimir();
    }
}

```

7. Realiza el procedimiento de creación del archivo jar del proyecto anterior con el MANIFEST y ejecútalo. Lista el contenidos del jar y extráelo. Adjunta los comandos y el proyecto total del fichero jar. (2p)

