Universidad Autónoma de Madrid

Escuela Politécnica Superior

Análisis y Diseño de Software 2015-2016

Práctica 1: Introducción a Java

Inicio: Semana del 1 de febrero.

Duración: 1 semana.

Entrega: Semana del 8 de febrero, el día anterior al inicio de la siguiente práctica según grupo, menos los grupos de los lunes que entregarán el mismo lunes antes de las 8:45.

Peso de la práctica: 5%

El objetivo de esta práctica es aprender el funcionamiento de algunas de las herramientas de la Java Development Toolkit (JDK), comprender el esquema de funcionamiento de la máquina virtual de Java y escribir tus primeros programas en Java.

Apartado 1: Hola Mundo

Con un editor de texto, teclea el siguiente programa, y guárdalo con el nombre *HolaMundo.java*

```
public class HolaMundo {

   public static void main(String[] args) {
      System.out.println("Hola mundo!");   // muestra el string por stdout
   }
}
```

En la línea de comandos, ejecuta la siguiente sentencia:

```
javac HolaMundo.java
```

para "compilar" la clase *HolaMundo* y generar el fichero *HolaMundo.class* correspondiente. Ejecuta el fichero *.class* mediantela sentencia:

```
java HolaMundo
```

Nota: El nombre del fichero *.java* tiene que ser el mismo que la clase que contiene respetando mayúsculas y minúsculas. Asegúrate que los programas *javac* y *java* estén en el PATH.

Apartado 2: Generación de Documentación.

El programa *javadoc* permite generar documentación HTML de los distintos programas fuente leyendo los comentarios del código. Por ejemplo, modifica el programa anterior incluyendo los siguientes comentarios, añadiendo tu nombre como autor:

```
/**
 * @author Estudiante EPS <estudiante.eps@uam.es>
 *
 * Esta aplicación muestra el mensaje "Hola mundo!" por pantalla
 */
public class HolaMundo {

 /**
  * Punto de entrada a la aplicación.
```

```
   *
   * Este método imprime el mensaje "Hola mundo!"
   *
   * @param args Los argumentos de la línea de comando
   */
    public static void main(String[] args) {
       System.out.println("Hola mundo!");   // muestra el string por stdout
    }
}
```

Genera la documentación del programa mediante la sentencia:

```
javadoc HolaMundo.java
```

Nota: Es conveniente almacenar los ficheros de documentación en un directorio separado. Por ejemplo, *javadoc –d doc HolaMundo.java* los almacena en el directorio *doc*, creando el directorio si no existe.

Abre la página *index.html* para ver la documentación generada. Tienes más información sobre el formato adecuado para comentarios *JavaDoc* en: http://en.wikipedia.org/wiki/Javadoc

---

Apartado 3: Uso de librerías básicas

El siguiente programa muestra cómo se reciben los parámetros por la línea de comandos.

```
/**
 *  @author Estudiante EPS <estudiante.eps@uam.es>
 *
 * Esta aplicación calcula el logaritmo Neperiano de su primer parámetro.
 */
public class Logaritmo {

  /**
   * Punto de entrada a la aplicación.
   *
   * Este método imprime el logaritmo Neperiano del número que se le pasa como entrada
   *
   * @param args Los argumentos de la línea de comando. Se espera un número como primer parámetro
   */
   public static void main(String[] args) {
      if (args.length<1) {
         System.out.println("Se espera un numero como parametro.");
         return;
      }

      String arg = args[0];   // una variable String que obtiene el primer parametro
      double x  = Double.parseDouble(arg);   // una variable double, que convierte arg a numerico

      System.out.println("El logaritmo de " + arg + " es: " + Math.log(x));
   }
}
```

Recuerda grabar el programa en un fichero llamado *Logaritmo.java*.

Ejecuta el programa con distintos parámetros (numéricos o no), o sin parámetros. ¿Qué sucede?

Apartado 4: Tu primer programa Java (4 puntos)

Modifica el programa anterior para que reciba exactamente 2 parámetros y calcule y muestre por pantalla la media de ambos.

La estructura del condicional en *Java* es exactamente la misma que en *C*:

```
if ( <expresión lógica> ) {
   <bloque de instrucciones "cierto">
} else {
  <bloque de instrucciones "falso">
}
```

Nota: La comprobación de igualdad en Java se escribe "==", y la de desigualdad "!=".

Apartado 5: Tu segundo programa Java (6 puntos)

Modifica el programa anterior para que reciba un número arbitrario de parámetros y calcule y muestre por pantalla la media de todos ellos. Además, el programa deberá escribir el número de parámetros recibidos, y si éste es par o impar.

De nuevo, la estructura de un bucle *while* es la misma que en *C*:

```
while ( <expresión lógica> ) {
   <cuerpo del bucle>
}
```

Nota: El operador módulo es "%".

Apartado 6: Ejercicio opcional (1 punto)

Modifica el programa anterior para que reciba como parámetros de entrada un máximo de 20 números, los ordene y muestre por pantalla de menor a mayor, y una vez ordenados, obtenga la mediana (el número que se encuentra en la posición central de la lista ordenada, teniendo en cuenta que si la longitud de la lista es par la mediana será la media de los dos elementos centrales), mostrándola por pantalla también.

No utilices las librerías *Java* de ordenación; codifica tú mismo el algoritmo de ordenación (usa cualquiera de los algoritmos aprendidos en Análisis de Algoritmos).

Java permite una estructura "clásica" de bucle *for* (con gestión explícita de índices) como ésta:

```
for ( <var inicialización>; <condición>; <actualización> ) {
   <cuerpo del bucle>
}
```

No obstante, siempre que sea posible, *es recomendable utilizar* la estructura de bucle *for* "mejorado" que no necesita índices explícitos y es la siguiente:

```
for ( <tipo de dato> <variable> : <array o lista> ) {
   <cuerpo del bucle>
}
```

Por ejemplo:

```
for ( String s : args ) {
   System.out.println(s); // no hay índice para recorrer el array
}
```

La reserva de memoria para un vector de *double* de tamaño 20 se realiza de la siguiente manera:

```
double[] numeros = new double[20];
```

Como veremos en próximos temas, la liberación de memoria en Java es automática, así que no necesitas liberar memoria al final del programa.

Normas de entrega:

- Se deberán entregar los apartados 4 y 5 (y opcionalmente el 6)
- El nombre de los alumnos debe ir en la cabecera *JavaDoc* de todas las clases entregadas
- La entrega la realizará uno de los alumnos de la pareja a través de Moodle
- Se debe entregar un único fichero ZIP / RAR con todo lo solicitado, que deberá llamarse de la siguiente manera: GR<numero_grupo>_<nombre_estudiantes>.zip. Por ejemplo Marisa y Pedro, del grupo 2261, entregarían el fichero: GR2261_MarisaPedro.zip
- La estructura de los ficheros entregados deberá ser la siguiente:
    - **src**. Ficheros fuente
    - **doc**. Documentación *JavaDoc* generada