PRUEBA 2 PROGRAMACIÓN Diciembre 2008 INGENIERÍA INFORMÁTICA Leganés	 Universidad Carlos III de Madrid	
Apellidos	***Nombre***	
Firma	***NIA***	***Grupo***

LEA <u>ATENTAMENTE</u> ESTAS INSTRUCCIONES ANTES DE COMENZAR LA PRUEBA:

- **Rellene todas las hojas a <u>bolígrafo,</u> tanto los datos personales como las respuestas**
- **No utilice lápiz ni bolígrafo rojo**
- **No olvide rellenar el NIA y el grupo <u>real</u> al que pertenece**
- **El tiempo máximo de realización es de 1 hora**
- **El único material permitido sobre la mesa es la hoja de test y un bolígrafo**
- **Utilice exclusivamente esta hoja de test para las respuestas, use las caras posteriores para contestar si lo necesita. <u>No se recogerá</u> ninguna otra hoja adicional.**

PARTE 1: CUESTIONES

Pregunta 1 (1 Punto).- Indicar si la siguiente afirmación es cierta, y **<u>explicar</u>** brevemente por qué.
"Cuando creamos un constructor dentro de una clase pasamos a tener dos, el que acabamos de crear y el que proporciona Java por defecto"

Falso. Cuando creamos un constructor en una clase el que proporciona Java por defecto desaparece. Debido a eso es conveniente crear siempre al menos dos constructores, siendo uno de ellos sin parámetros, similar al que proporciona Java por defecto.

Pregunta 2 (1 Punto).- Indicar si la siguiente afirmación es cierta, y **explicar** brevemente por qué.
"Para crear objetos de una clase en otra clase situada en un paquete diferente es suficiente con que la primera clase haya sido declarada como `public`"

Falso. Es necesario también importar la clase ya que pertenece a otro paquete.

Pregunta 3 (1 Punto).- Dada la clase `Pregunta3` indicar cuál es el resultado de ejecutar el método `main` de la clase `UsoPregunta3`, y **explicar** brevemente por qué.

```
public class Pregunta3 {
      public int atributo1;

      public void compara (int atributo1, long a2){
            if (atributo1>a2)
                  {atributo1=(int)a2/2;                    }
            }}

public class UsoPregunta3 {
      public static void main (String [] args){
            Pregunta3 a = new Pregunta3();
            a.atributo1 = 2;
            a.compara(5,3);
            System.out.println(a.atributo1);
            }

}
```

La segunda clase crea un objeto de la clase `Pregunta3` y establece el valor del `atributo1` a 2. Luego llama al método `compara` con los argumentos (5,3). El método `compara` tiene un parámetro que se llama igual que el `atributo1`, y que lo oculta dentro del método por lo tanto el valor del `atributo1` no cambia dentro del método y se imprime 2 por pantalla.

Pregunta 4 (1 Punto).- Encontrar y **explicar** los 3 errores de compilación que aparecen en el siguiente código Java. ¿Cómo los resolvería?

```
public class Pregunta4 {
	int a;
	private boolean b;
	protected char c;

	public Pregunta4 (int a1, char c1){
		a=a1;
		b=true;
		c=c1;}
	public Pregunta4 (int a1, char c1, boolean b1){
		b=b1;
		this(a1,c1);}
	public Pregunta4 (int a2, char c2){
		a=a2;
		b=true;
		c=c2;}
	public Pregunta4 (int a, boolean b, char c){
		a=a;
		this.b=b;
		this.c=c;}

	public getA (){
		return a;}}
```

1) El primer constructor y el tercero reciben parámetros del mismo tipo, aunque con distintos nombres. Habría que quitar uno de los dos constructores.
2) En el segundo constructor, el `this` debe ser la primera instrucción.
3) En el método get falta especificar el tipo que devuelve, habría que poner `public int getA ()`

Pregunta 5 (1 Punto).- Crear un método que reciba un array de caracteres de **cualquier** longitud y devuelva otro array en el que los elementos estén en orden inverso. Ejemplo: recibe {‘a’, ‘b’, ‘c’, ‘d’} y devuelve {‘d’,’c’,’b’,’a’}

```
public char [] metodo (char [] a){
	//Creamos un array auxiliar para copiar el original
	char [] aux = new char [a.length];
	//Usamos un bucle para copiar
	for (int i=0; i<a.length; i++)
		aux[a.length-1-i]=a[i];
	return aux;

}
```

Pregunta 6 (1 Punto).- Indicar y **explicar** cuál es el resultado por pantalla del siguiente programa:

```
public class Pregunta6 {
      public static void main(String[] args) {
            int a = 3;
            while (a < 6){
                  for (int i=3; i>1;i--){
                        if (a%3==0){
                              System.out.println(a*i);}}
                         a++;
            }     } }
```

Es un bucle `for` anidado dentro del bucle `while`. En la primera ejecución, como a es menor que 6, se ejecuta el `for`, e `i` va tomando los valores 3, 2. Dentro del `for` se comprueba si a es divisible por 3, en ese caso se imprime `a*i`. Por lo tanto los valores son:

a	i	a%3	a*i
3	3	0	9
3	2	0	6

A continuación sale del `for`, incrementa a y vuelve al `while`, como a<6 vuelve a entrar en el `for`, pero ahora como `a= 4` no imprime nada. Lo mismo ocurre para `a= 5.` Vuelve a incrementar a y como 6<6 se sale del `for`.

Por lo tanto imprime:

9

6

Pregunta 7 (1 Punto).- Dados los siguientes pares de métodos sobrescritos (de los que sólo se muestran las cabeceras), **explicar** cuáles pueden estar dentro de una misma clase y cuáles no.

```
a) int metodo1 (int a, int b){...}        int metodo1 (int d, int c){...}
b) void metodo1 (int a) {...}             long metodo1 (int a) {...}
c) int metodo1 (int a){...}               int metodo1 (long a) {...}
d) float metodo1 (short a, int b) {...}   void metodo1 (short d, int i) {...}
```

a) No pueden porque tienen parámetros del mismo tipo, da igual que se llamen de manera distinta, Java no sabría a cuál de los dos nos referimos.
b) No pueden porque sólo se diferencian en el valor de retorno y tienen que diferenciarse en el tipo y/o número de parámetros.
c) Si pueden, uno recibe un int y el otro un long, Java sabe a cuál estamos llamando viendo el tipo del parámetro.
d) No pueden, es una combinación de a) y b)

PARTE 2: PROBLEMAS

Problema 1 (3 Puntos).-

Crear una clase llamada `Perro` que tendrá las siguientes características:

- **(0,2 puntos)** Debe tener 3 atributos privados denominados `nombre`, `edad` y `dueño`.
- **(0,2 puntos)** Hacer un método denominado `getNombre` que devuelva el nombre del perro.
- **(0,4 puntos)** Hacer un método `setEdad` que reciba como parámetro la `edad` del perro. Deberá comprobar que el valor recibido es válido.
- Hacer los siguientes constructores (todos deberán comprobar que la edad recibida es correcta)
  - **(0,5 puntos)** Uno que reciba valores para todos los atributos de la clase `Perro`.
  - **(0,4 puntos)** Uno por defecto, sin parámetros que **usando** el anterior cree un `Perro` de 10 años, llamado Lenka y cuyo dueño sea Pepe.
  - **(0,4 puntos)** Uno que sólo reciba valores para el `nombre` y el `dueño` y ponga la `edad` a 1 año.
  - **(0,4 puntos)** Uno de copia que reciba como parámetro un objeto de tipo `Perro` y cree otro con los mismos atributos.

(0,5 puntos) Crear una clase Denominada `UsoPerro` que cree cuatro objetos de tipo `Perro`, uno con cada uno de los constructores anteriores.

```
public class Perro {
	private String nombre, dueño;
	private int edad;

	public String getNombre(){
		return nombre;
	}
	public void setEdad (int e){
		//Un perro vive siempre menos de 30 años
		if (e>=0 && e<30) edad = e;
	}
	// Constructor completo
	public Perro (String n, String d, int e){
		dueño = d;
		nombre = n;
		setEdad(e);
	}
	// Constructor sin parámetros
	public Perro(){
		this ("Lenka","Pepe",10);
	}
	//Tercer constructor
	public Perro (String n, String d){
		this (n,d,1);
	}
	//Constructor de copia
	public Perro (Perro p){
		this(p.nombre,p.dueño,p.edad);
	}
}
public class UsoPerro {

	public static void main(String[] args) {
		Perro p1,p2,p3,p4;
		p1 = new Perro ("Tobi","Juan",5);
		p2 = new Perro();
		p3 = new Perro ("Canelo","Antonio");
		p4 = new Perro (p2);
	}}
```