

## Programación de Sistemas Grado en Ingeniería de Sistemas de Comunicaciones

Leganés, 15 de marzo de 2019  
Duración de la prueba: 20 min

Examen parcial 1 (teoría)  
Puntuación: 3 puntos sobre 10 del examen

*Sólo una opción es correcta en cada pregunta. Cada respuesta correcta suma 0,3 puntos. Cada respuesta incorrecta resta 0,1 puntos. Las preguntas no contestadas no suman ni restan puntos.*

Marca:  Anula:  No uses:   

- Marca la respuesta a cada pregunta con una equis (“X”) en la tabla de abajo.
- Si marcas más de una opción o ninguna opción, la pregunta se considera no contestada.
- Rellena **tus datos personales** antes de comenzar a realizar el examen.

Nombre:





Grupo:





Firma:

NIA: 

--	--	--	--	--	--	--	--	--	--

	A	B	C	D		A	B	C	D
1					6				
2					7				
3					8				
4					9				
5					10				

1.- Dadas las clases `ClaseA` y `ClaseB` que hereda de `ClaseA`. Si queremos ampliar su funcionalidad con una nueva clase `ClaseC` y dos Interfaces `I1` e `I2`. Indica cuál de las siguientes sentencias daría un error de compilación.

- (a) `*** public class ClaseB extends ClaseA, ClaseC`
- (b) `public class ClaseA implements I1`
- (c) `public class ClaseA extends ClaseC`
- (d) `public class ClaseB implements I1, I2`

2.- Dado el siguiente código. Indica cuántas veces se ejecuta el bucle y cuánto vale el array `datos` tras invocar al siguiente método para `datos = {1,1,1,1,1}`.

```
public static void metodo(int[] datos){  
    for(int i=0; i<datos.length;i++){  
        datos[i] = 3;  
        i= i+2;  
    }  
}
```

- (a) `*** 2 veces, 3,1,1,3,1`
- (b) `5 veces, 3,3,3,3,3`
- (c) Ninguna opción es correcta.
- (d) `3 veces, 3,1,3,1,3`

3.- Dado el siguiente código indique cuál de las siguientes afirmaciones es *incorrecta*.

```
public class ClaseA{  
    private int a;  
    public ClaseA(int a){this.a=a;}  
    public ClaseA(){this(null);}  
}  
public class ClaseB extends ClaseA{...}
```

- (a) Podemos crear un constructor en la clase `ClaseB` que desde su primera línea haga una llamada a `super()`.
- (b) La clase `ClaseB` tiene un constructor por defecto.
- (c) La clase `ClaseA` tiene sobrecarga de constructores.
- (d) `*** La clase ClaseB puede sobrescribir el constructor por defecto de ClaseA`

4.- Indica cuál de las siguientes afirmaciones es *incorrecta*.

- (a) Los constructores se pueden sobrecargar.
- (b) Para que haya sobrescritura tiene que haber herencia.

- (c) \*\*\* Para que haya sobrecarga tiene que haber herencia.
- (d) Cuando hay herencia se puede invocar al método `m()` de la clase padre desde el método `m()` de la clase hija con `super.m()` aunque ambos métodos tengan el mismo nombre y número de parámetros en ambas clases.

5.- Dado el siguiente código. Indica cuál de las siguientes sentencias es *incorrecta*.

```
public Interface I{
    void metodo1();
}
public abstract class ClaseA implements I{
    public void metodo2() {System.out.println("ejecutando metodo2");}
}
public class ClaseB extends ClaseA{
    public void metodo1(){System.out.println("ejecutando metodo1");}
    public void metodo2(){super.metodo2()}
}
```

- (a) `I i = new ClaseA()`
- (b) `ClaseA a = new ClaseB()`
- (c) \*\*\* `ClaseB b = (ClaseB) new ClaseA();`
- (d) `I i = new ClaseB()`

6.- Dado el siguiente código indica cual de las siguientes sentencias para modificar los valores de `a`, `b` y `c` es *incorrecta*.

```
public class ClaseA{
    protected int a = 1;
    public static int b = 1 ;
    public static final int C = 1;
    public void setA(int a){this.a = a;}
    public static void main(String[] args){
        ClaseA miObjeto = new ClaseA();
        //sentencias para modificar los valores
    }
}
```

- (a) `miObjeto.setA(3);`
- (b) `ClaseA.b = 3;`
- (c) `miObjeto.a = 3;`
- (d) \*\*\* `ClaseA.C = 3;`

7.- Indica qué devuelve el siguiente método recursivo para `m(0)`, `m(2)`, `m(3)`.

```

public static int m(int a){
    int resultado = 0;
    if(a<=0){
        resultado = 1;
    }else{
        resultado = m(0) * m(a-1) * m(a-2);
    }
    return resultado;
}

```

- (a) Es un caso de recursión anidada.
- (b) 0,0,0
- (c) \*\*\* 1,1,1
- (d) La recursión no termina.

8.- Indica cuál de las siguientes afirmaciones sobre pruebas de programas es incorrecta.

- (a) Las pruebas de cobertura de métodos son un caso de pruebas de caja blanca.
- (b) \*\*\* Las clases de equivalencia también se denominan pruebas de excepciones.
- (c) Las pruebas alfa y beta son pruebas que se hacen con el cliente.
- (d) Se pueden hacer pruebas funcionales integrando varios módulos y equivalen a las pruebas de caja negra.

9.- Dado el siguiente código indica cual de las siguientes afirmaciones es *incorrecta*.

```

public class ClaseA{
    private char a;
    public ClaseA(char a){setA(a);}
    public void setA(char a){this.a=a;}
    public static void main(String[] args){
        ClaseA miObjeto = new ClaseA('r');
    }
}

```

- (a) El estado del objeto `miObjeto` es privado.
- (b) La clase `ClaseA` no tiene constructor por defecto.
- (c) \*\*\* No se puede llamar al método `setA` desde el constructor.
- (d) El comportamiento del objeto `miObjeto` es público.

10.-Cuál de las siguientes afirmaciones sobre recursión es *cierta*

- (a) La recursión en cascada implica dos métodos que se llaman entre sí.
- (b) La recursión por la cola es normalmente más eficiente que un bucle.
- (c) La recursión anidada también se puede llamar lineal por la cola.
- (d) \*\*\* La recursión no por la cola requiere realizar operaciones pendientes después de la última llamada recursiva.