EXAMEN DE PROGRAMACIÓN **15 de Junio 2009** **GRADO EN INGENIERÍA INFORMÁTICA** **DOBLE GRADO EN INGENIERÍA INFORMÁTICA Y ADMINISTRACIÓN DE EMPRESAS** **Leganés y Colmenarejo**	Universidad Carlos III de Madrid	
Apellidos	***Nombre***	
Firma	***NIA***	***Grupo***

LEA ATENTAMENTE ESTAS INSTRUCCIONES ANTES DE COMENZAR LA PRUEBA:

- **Rellene todas las hojas a bolígrafo, tanto los datos personales como las respuestas**
- **No utilice lápiz ni bolígrafo rojo**
- **No olvide rellenar el NIA y el grupo real al que pertenece**
- **El tiempo máximo de realización es de 3 horas**
- **Se permiten apuntes y/o libros para la realización del examen**
- **Para la primera parte del examen (cuestiones 1 a 4) utilice exclusivamente estas hojas, use las caras posteriores para contestar si lo necesita. No se recogerá ninguna otra hoja adicional.**
- **La segunda parte del examen (problemas 1 y 2) deben realizarse en hojas aparte.**

PARTE 1: CUESTIONES

Pregunta 1 (1 Punto).- Indicar si las siguientes afirmaciones son o no ciertas, y **explicar** brevemente por qué.

1.1. **(0,5 puntos)** Una vez definido el tamaño de un array, es imposible cambiarlo a no ser que se vuelva a crear de nuevo, con lo que se perderían sus datos.

1.2. **(0,5 puntos)** Dentro de una clase podemos crear dos métodos con el mismo nombre sin ningún tipo de restricción.

1.1. **Verdadero**, si asignamos memoria para un array no podemos cambiar su tamaño dinámicamente, a no ser que volvamos a asignarle una memoria nueva, con lo que se perderían los datos anteriores.

1.2. **Falso**, existen ciertas restricciones, los métodos de igual nombre no pueden tener el mismo número y tipo de los argumentos, porque el compilador no sabría a cuál de los dos llamar en caso de invocarlo.

Pregunta 2 (1 Punto).- Indicar si las siguientes afirmaciones son o no ciertas, y **explicar** brevemente por qué.

2.1. **(0,5 puntos)** "El siguiente programa imprime 8 por pantalla."

```
public class Pregunta2 {
	public static void main(String[] args) {
		int i = 1;
		while (i<8) {
			int j;
			i++;
			j=i;
		}
		System.out.println(j);
	}}
```

2.2. **(0,5 puntos)** El resultado de ejecutar el siguiente código es:

```
                                true
                                false
public class Pregunta2 {
	public static void main(String[] args) {
		int [] a = {1,3,5}, b = {1,3,5};
		System.out.println(a==b);
		a = b;
		b[2]=4;
		System.out.println(a==b);
}}
```

2.1. Falso. El programa **no compila** porque la variable j se declara dentro del bucle, con lo que no puede utilizarse fuera para ser impresa.

2.2. Falso. Al comparar los dos arrays con == estamos comparando sus direcciones de memoria, no sus valores. Por tanto la primera impresión por pantalla sería **false**, mientras que la segunda sería **true** porque ahí ya se han igualado los arrays.

Pregunta 3 (1 Punto).- Dadas las siguientes clases, **explicar** brevemente cuál es el resultado de ejecutar el método `main` de la clase `UsoPregunta3`.

```
public class Pregunta3 {
	float a, b;

	public float metodo1 (float b){
		if (b>a) {
			return b;
		}
		else {
			return b-a;
		}
}}

public class UsoPregunta3 {
	public static void main (String [] args){
		Pregunta3 obj = new Pregunta3();
		obj.a=5;
		obj.b=6.2F;
		System.out.println(obj.metodo1(obj.a));
		System.out.println(obj.metodo1(obj.b));
		System.out.println(obj.metodo1(obj.b));
		System.out.println(obj.metodo1(obj.a));
}}
```

La variable b dentro del metodo1 se refiere a la que se ha pasado como parámetro (porque no se utiliza this delante de ella). Por tanto la salida por pantalla es:

0
6,2
6,2
0

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 {
	float b;

	public Pregunta4 (int a){
		b=a;
	}
	public  Pregunta4 (int a, float b){
		this.b=a+b;
	}
	public float metodo1 (float c){
		b=c;
		System.out.println(b);
	}
	public Pregunta4(int d, float h){
		b=h;
	}
}
public class UsoPregunta4 {
	public static void main(String[] args) {
		Pregunta4 ob1, ob2;
		ob1 = new Pregunta4 ();
		ob2 = new Pregunta4 (25);
	}
}
```

1) No puede haber **dos constructores con el mismo número y tipo de parámetros** (el segundo y el tercero). Se resolvería eliminando uno de ellos o cambiando los parámetros.
2) En el método main, al crear el primer objeto de tipo Pregunta4 **se está intentando llamar al constructor sin argumentos**, que no existe puesto que se han creado otros. Se resolvería llamando a otro de los constructores que sí existe.
3) El metodo1 se ha declarado que devuelve un float, **pero no tiene ninguna sentencia de tipo return**. Se resolvería añadiendo dicha sentencia.

PARTE 2: PROBLEMAS

Problema 1 (5 Puntos).- Crear una clase pública denominada `Barco` con las siguientes características:

- **(0,1 puntos)** Debe pertenecer al paquete `junio`.
- **(0,2 puntos)** Debe contener los siguientes atributos públicos, `nombre` de tipo cadena de texto, `capacidad` de tipo entero y `camarotes` de tipo array de cadenas de texto. El nombre del barco no debe poder cambiarse una vez puesto.
- **(0,3 puntos)** Crear un método `getCapacidad` que devuelva la `capacidad` del barco y otro `setCapacidad` que reciba como parámetro la nueva `capacidad` y dé valor al atributo `capacidad`. El segundo método deberá comprobar que la `capacidad` sea mayor que cero y si no ponerla a 20.
- Crear los siguientes constructores
  - **(0,5 puntos)** Un constructor sin parámetros que ponga como `nombre` del barco "`la perla negra`" y como `capacidad` 20 personas. El tamaño del array `camarotes` debe ser también 20 y todas sus posiciones deben contener la palabra "`vacío`".
  - **(0,4 puntos)** Un constructor que reciba parámetros para el `nombre` y la `capacidad` y además cree el array `camarotes` con tamaño igual a la `capacidad` (y relleno de "`vacío`"). Debe comprobar que la `capacidad` es mayor que cero y en caso contrario ponerla a 20.
  - **(0,4 puntos)** Un constructor que sólo reciba la `capacidad` y **usando** el constructor anterior ponga como `nombre` "`Zephyr`"
- **(0,5 puntos)** Crear un método, denominado `colocar`, que reciba como parámetro un `String` con el `nombre` del pasajero y lo coloque en el primer camarote que quede libre (un camarote está libre si está vacío). Debe devolver un `int` con el número de camarote (el primer camarote es el número 0, el segundo es el número 1, etc.) o -1 si no hay camarotes libres.
- **(0,7 puntos)** Crear un método, denominado también `colocar`, que reciba como parámetros un `String` con el `nombre` del pasajero y un `int` con el número de camarote (recordar que el primer camarote es el número 0) e intente colocar a ese pasajero en el camarote. Si el camarote está ocupado deberá colocarlo en el primero disponible (tal y como hacía el método anterior). Deberá devolver un `int` con el número de camarote o -1 si no hay camarotes libres.
- **(0,6 puntos)** Crear un método, denominado `ocupacion`, que no reciba parámetros y devuelva un `array` de `enteros` de dos posiciones en el que la primera sea el número de `camarotes` vacíos y la segunda el de ocupados.
- **(0,5 puntos)** Crear un método, denominado `colocarAleatorio`, que reciba como parámetro el `nombre` de un pasajero y lo coloque en una posición aleatoria del `array camarotes`. Si el camarote de la posición aleatoria seleccionada está ocupado deberá colocarlo en el primero disponible. Deberá devolver un int con el número de camarote o -1 si no hay camarotes libres.
- **(0,4 puntos)** Crear un método denominado `estaOcupado` que reciba un `int` que representa el número de camarote y devuelva un `String` diciendo "`El camarote número" x "está ocupado por " nombre del pasajero`, si el camarote está ocupado, o "`El camarote número" x "está vacío".`
- **(0,4 puntos)** Crear un método denominado `desembarcar` que no reciba parámetros ni devuelva nada y que vacíe el barco (borrando todos los pasajeros del `array` de `camarotes` y poniéndolo a "`vacío`")

Problema 2 (1 Punto).- Crear una clase pública denominada `UsoBarco` con las siguientes características:

- **(0,2 puntos)** Debe pertenecer al paquete `usojunio`.
- **(0,1 puntos)** Debe contener un método `main`.

- **(0,3 puntos)** Declarar dentro del método `main` tres variables de tipo `Barco`. Crearlas usando un constructor distinto para cada una.
- **(0,4 puntos)** Insertar un pasajero en el primer barco creado usando el primer método `colocar` y otro usando el segundo. Crear un bucle que imprima por pantalla el estado (usando el metodo `estaOcupado`) de todos los camarotes del barco.

PROBLEMA 1

```
package junio;

import java.util.Random;

public class Barco {
	public final String nombre;
	public int capacidad;
	public String[] camarotes;

	public int getCapacidad() {
		return capacidad;
	}

	public void setCapacidad(int capacidad) {
		if(capacidad > 0) {
			this.capacidad = capacidad;
		}
		else {
			this.capacidad = 20;
		}
	}

	public Barco() {
		this.nombre = "la perla negra";
		this.capacidad = 20;
		this.camarotes = new String[capacidad];
		for(int i=0; i<camarotes.length; i++) {
			this.camarotes[i] = "vacio";
		}
	}

	public Barco(String nombre, int capacidad) {
		this.nombre = nombre;
		setCapacidad(capacidad);
		this.camarotes = new String[capacidad];
		for(int i=0; i<camarotes.length; i++) {
			this.camarotes[i] = "vacio";
		}
	}

	public Barco(int capacidad) {
		this("Zephyr", capacidad);
	}

	public int colocar(String pasajero) {
		for(int i=0; i<capacidad; i++) {
			if(camarotes[i].equals("vacio")) {
				camarotes[i] = pasajero;
				return i+1;
			}
		}
		return -1;
	}

	public int colocar(String pasajero, int numCamarote) {
		for(int i=numCamarote; i<capacidad; i++) {
			if(camarotes[i].equals("vacio")) {
				camarotes[i] = pasajero;
				return i;
			}
		}
```

```
			for(int i=0; i<numCamarote; i++) {
				if(camarotes[i].equals("vacio")) {
					camarotes[i] = pasajero;
					return i;
				}
			}
			return -1;
		}

		public int[] ocupacion() {
			int libres = 0, ocupados = 0;
			for(int i=0; i<capacidad; i++) {
				if(camarotes[i].equals("vacio")) {
					libres++;
				}
				else {
					ocupados++;
				}
			}
			return new int[] {libres, ocupados};
		}

		public int colocarAleatorio(String pasajero) {
			Random rnd = new Random(System.currentTimeMillis());
			int aleatoria = rnd.nextInt(capacidad);
			return colocar(pasajero, aleatoria);
		}

		public String estaOcupado(int numCamarote) {
			if(camarotes[numCamarote].equals("vacio")) {
				return "El camarote número " + numCamarote + " está vacio";
			}
			return "El camarote número " + numCamarote + " está ocupado por " +
camarotes[numCamarote];
		}

		public void desembarcar() {
			for(int i=0; i<camarotes.length; i++) {
				this.camarotes[i] = "vacio";
			}
		}
}
```

PROBLEMA 2

```
package usojunio;

import junio.Barco;

public class UsoBarco {
	public static void main(String[] args) {
		Barco b1 = new Barco();
		Barco b2 = new Barco("el barco de chanquete", 10);
		Barco b3 = new Barco(30);
		b1.colocar("Pepe");
		b1.colocar("Luis", 3);
		for(int i=0; i<b1.capacidad; i++) {
			System.out.println(b1.estaOcupado(i+1));
		}
	}
}
```