Apellidos:

Nombre: D.N.I.:

Introducción a la Programación - E.P. de Ingeniería de Gijón

17 de Enero de 2019

1. (1 p) Suponiendo que ya has declarado una variable de tipo entero x, declara, *con una sola instrucción*, un vector que contenga los siguientes valores reales: el mismo valor que el de x, la mitad del valor de x, el valor constante 3,1416.

Solución:
```
double[] vector = {x, x/2.0, 3.1416};
```

2. (2 p) Dada la siguiente definición de variables y sus valores iniciales,

```
String s=new String("Hola"), t=new String("Hola"), r='5'; int x=5, y=7;
```

indica para las siguientes expresiones, si son o no correctas (SI/NO), en caso de resultar incorrectas JUSTIFICA por qué lo son, y en el caso de ser correctas indica el TIPO y el VALOR que producen.

Nota: Los dígitos, al igual que las letras del alfabeto, ocupan posiciones consecutivas en la tabla de códigos.

		Tipo	Valor	Motivo
`s == t`	☒Si ☐No	boolean	false	
`12 = x + y`	☐Si ☒No			La constante 12 no es L-value, no se le puede asignar valor.
`r - x == '0'`	☒Si ☐No	boolean	true	
`x = y = s + t`	☐Si ☒No			No se puede asignar un objeto `String` a variables `int`

3. (3 p) Suponiendo que $N \geq 0$ es **par** indica cuántos asteriscos imprime cada uno de los bucles siguientes, así como el valor de la variable i **a la salida del bucle**.

`int i = 1;`
`while (i != N) {`
`  System.out.print('*');`
`  i+=2;`
`}`
	`for (int i = 1; i <= 2*N; i+=2) {`
	`  System.out.print('*');`
	`}`
		`int i = 0;`
		`do {`
		`  i += 2;`
		`  System.out.print('*');`
		`} while (i <= N);`

¿Cuántos asteriscos? Sol.: Infinitos ¿Valor de i? Sol.: Ninguno, nunca sale	¿Cuántos asteriscos? Sol.: N ¿Valor de i? Sol.: Ninguno, i no existe	¿Cuántos asteriscos? Sol.: $\frac{N}{2}+1$ ¿Valor de i? Sol.: $N+2$

4. (6 p) Implementa un método `seno(x, n)` que calcule de forma aproximada el seno del ángulo x (expresado en radianes) utilizando los n primeros términos de la siguiente serie de McLaurin:

$$\text{sen}(x) = \sum_{i=1}^{\infty} \frac{(-1)^{i-1}}{(2i-1)!} x^{2i-1} = x - \frac{1}{3!}x^3 + \frac{1}{5!}x^5 - \frac{1}{7!}x^7 + \dots$$

Notas: Atención a la alternancia de signo en la suma de términos. Puedes utilizar `Math.pow()` y `factorial()`, aunque éste último deberás implementarlo también.

Solución:
```
public static long factorial(int n) {
    long f = 1;
    for (int i=2; i <= n; i++)
        f *= i;
    return f;
}
public static double seno(double x, int n) {
    // Secuencia: (1, +1), (2, -1), (3, +1), (4, -1), ... hasta que i > n
    double seno_aprox = 0.0;
    // Primer elemento
    int i = 1;
    int signo = +1;
    // Mientras no fin secuencia
    while (i <= n) {
        // tratar elemento (acumular suma)
        seno_aprox += signo * 1.0/factorial(2*i-1) * Math.pow(x, 2*i-1);
        // elemento siguiente
        i++;
        signo = -signo;
    }
    return seno_aprox;
}
```

5. (2 p) Dadas las variables enteras i, j y k, y utilizando sintáxis de lenguaje Java escribe las condiciones booleanas equivalentes a las siguientes expresiones:

   El valor de i está en el intervalo $[j, k]$

   Solución: `i >= j && i <= k`

   i es múltiplo de j pero j no es divisor de k

   Solución: `i%j == 0 && k%j != 0`

   i, j y k son las longitudes de los lados de un triángulo (la suma de dos cualesquiera es mayor que la tercera)

   Solución: `i+j > k && j+k > i && k+i > j`

   Recorrer una distancia de j cm. requiere dar k pasos de i cm. cada uno y todavía nos quedarían 8 cm. por recorrer

   Solución: `j/i == k && j%i == 8` `/* o también vale */` `(j-8)/i == k`

6. (6 p) Una empresa de transporte aplica distintas tarifas dependiendo del tipo de carga y la zona de destino según la siguiente tabla. Se pide hacer un **programa de consola completo** (importando las clases que necesites) que pida por teclado los valores de **tipo** y **zona** e imprima la tarifa que debe aplicarse (A, B, D, E o F). Se valorará especialmente el diseño del esquema condicional.
   Nota: No hace falta incluir código para comprobar sin son correctos.

Tipo Carga \ Zona Destino	1	2	3	4	5
1	A	B	B	B	B
2	A	F	F	F	C
3	A	E	F	F	C
4	A	E	E	F	C
5	A	D	D	D	C

Solución:

```
import java.util.Scanner;
public class Tarifas {

    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);
        System.out.println("Introduce Carga y Destino: ");
        int tipo = teclado.nextInt();
        int zona = teclado.nextInt();
        char tarifa;

        if (zona == 1)
            tarifa = 'A';
        else
            if (tipo == 1)
                tarifa = 'B';
            else
                if (zona == 5)
                    tarifa = 'C';
                else if (tipo == 5)
                    tarifa = 'D';
                else
                    if (tipo > zona)
                        tarifa = 'E';
                    else
                        tarifa = 'F';

        System.out.printf("La tarifa aplicable es la %c\n", tarifa);
    }
}
```

7. (6 p) Escribe el método público y estático **esHexadecimal()** que recibe un objeto String y retorna cierto si la cadena representa un número en base 16, o falso en caso contrario. Para representar un número en hexadecimal necesitamos los dígitos (0 al 9) así como las letras de la A a la F. Ejemplos: "50" retorna cierto, "J673" retorna falso, "b" retorna cierto, "A1Fc" retorna cierto. Puedes crear métodos adicionales si lo consideras apropiado.

Solución:

```
public static boolean esDígitoHex(char c) {
    return (c >= '0' && c <='9') || c >= 'a' && c <= 'f')  || (c >= 'A' && c <= 'F');
}

public static boolean esHexadecimal(String hex) {
    // Primer elemento
    int i = 0;
    // Mientras NO fin de secuencia Y NO elemento encontrado
    while (i < hex.length() && esDígitoHex(hex.charAt(i)))
        // elemento siguiente
        i++;
    // si llega al final de la secuencia es que no hay errores , luego ES cierto que es hexadecimal
    return i >= hex.length();
}
```

8. (6 p) Escribir el método público y estático **lt2vector()**, que recibe una matriz cuadrada de reales y retorna un vector de reales con los valores del triángulo inferior de la matriz. El vector debe tener la dimensión adecuada y se rellenará recorriendo la matriz por filas, tal como se muestra en el ejemplo:

$$M = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{pmatrix} \quad \Longrightarrow \quad V = \begin{pmatrix} 5 & 9 & 10 & 13 & 14 & 15 \end{pmatrix}$$

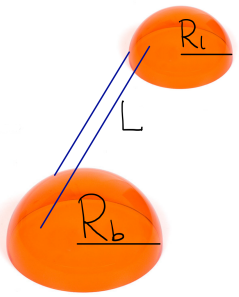
Solución:

```
public static double[] lt2vec(double[][] m) {
    // dado que la matriz m es cuadrada, la longitud del vector es ...
    int len = (m.length*m.length - m.length)/2;
    double[] vector = new double[len];
    int k = 0; // índice para rellenar el vector
    // recorremos la matriz triangular inferior
    for (int i = 1; i < m.length; i++)
        for (int j = 0; j < i; j++) {
            vector[k] = m[i][j];
            k++;
        }
    return vector;
}
```

9. (8 p) Implementa la clase `LámparaDespacho` para representar lámparas de escritorio con la forma que se muestra en la figura. Cada lámpara viene determinada por tres valores dados en milímetros: `Rb`$\in [50, 90]$, `Rl`$\in [25, 40]$ y `L`$\in [120, 500]$.

Deben implementarse los siguientes métodos públicos:

- Cuatro constructores: *constructor por defecto*, con los valores mínimos en cada atributo; *con un parámetro*, que será el valor para `Rb` (los demás atributos tendrán el mínimo valor posible); *con tres parámetros*; y constructor *copia*.
- Métodos `set()` y `get()`. Cuando se suministre un valor fuera del rango permitido para cualquier atributo, éste tomará el mínimo valor posible.
- Métodos `áreaBase()` y `áreaLuz()` que retornan la superficie de la semiesfera correspondiente (área de una esfera: $4\pi r^2$).
- Método `precioVentaRecomendado()`, que retorna el coste de los materiales de la lámpara más un 10 % de beneficio empresarial. El material de los brazos cuesta 38€/m y el de las planchas para hacer las semiesferas cuesta 50€/m^2.
- Método `másBarata()`, para comparar el precio de venta recomendado entre dos lámparas. Ejemplo: `l1.másBarata(l2)` retorna `true` si el precio de `l1` < precio de `l2`.



Solución:

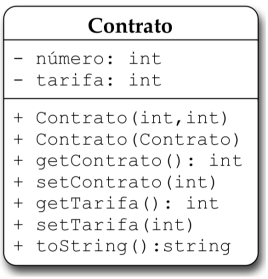
```
public class Lámpara {
    private final static double minRb=50, maxRb=90, minRl=25, maxRl=40, minL=120, maxL=500;
    private double Rb, Rl, L;

    public Lámpara(double rb, double rl, double l) {
        this.setRb(rb);
        this.setRl(rl);
        this.setL(l);
    }
    public Lámpara() {
        this(minRb, minRl, minL);
    }
    public Lámpara(double rb) {
        this(rb, minRl, minL);
    }
    public Lámpara(Lámpara X) {
        this(X.getRb(), X.getRl(), X.getL());
    }
    public double getRb() {
        return Rb;
    }
    public double getRl() {
        return Rl;
    }
    public double getL() {
        return L;
    }
    public void setRb(double rb) {
        if (rb >= minRb && rb <= maxRb)
            Rb = rb;
        else
            Rb = minRb;
    }
    public void setRl(double rl) {
        if (rl >= minRl && rl <= maxRl)
            Rl = rl;
        else
            Rl = minRl;
    }
    public void setL(double l) {
        if (l >= minL && l <= maxL)
            L = l;
        else
            L = minL;
    }
    public double áreaBase() {
        return 2*Math.PI*this.getRb()*this.getRb();
    }
    public double áreaLuz() {
        return 2*Math.PI*this.getRl()*this.getRl();
    }
    public double precioVentaRecomendado() {
        double coste_brazos = 38/1000.0 * this.getL() * 2; // son dos brazos
        double coste_base = 50/1000000.0 * this.áreaBase();
        double coste_luz = 50/1000000.0 * this.áreaLuz();
        return 1.10 * (coste_brazos + coste_base +coste_luz);
    }
    public boolean másBarata(Lámpara l) {
        return this.precioVentaRecomendado() < l.precioVentaRecomendado();
    }

}
```

10. (6 p) Escribir la clase `Cliente` que representa los datos de un cliente de una compañía de luz y gas. Los datos que tiene un objeto `Cliente` son:
    - El `nombre` del titular: cadena de caracteres.
    - La `dirección` de su vivienda: cadena de caracteres.
    - Los atributos `luz` y `gas`: son objetos de la clase `Contrato`. Tiene dos atributos: el número de contrato y el código de la tarifa, ambos enteros.
    - El consumo: cantidad en euros gastada por el cliente.

    Notas: No hay que implementar los métodos `get()` y `set()`, pero debes usarlos para acceder a los atributos en el resto de métodos como si existiesen. En el constructor, los atributos luz y gas NO hay que inicializarlos con su método `set` correspondiente; si lo haces así, entonces debes programar además dichos métodos. `cargarFactura()` y `cambiarTarifa()` tienen que comprobar que el núm. de contrato es del cliente y el importe positivo. Hay que emplear siempre los métodos de la clase `Contrato`, por ejemplo, `toString()` con el atributo `luz` de `c1` imprimiría: "No: 123879 Tarifa: 2".



Los métodos a implementar deben hacer funcionar el código siguiente:

```
Cliente c1=new Cliente("Ana","Uría,6 2ºI",123879, 2, 345536,4);
                        //nombre, dirección, núm. Luz, tarifa Luz, núm. Gas, tarifa Gas
c1.cargarFactura(123879,35.6); // consumoTotal = 35.6
c1.cargarFactura(346456,124.96); //no se carga, no coincide núm de contrato
c1.cambiarTarifa(345536,3); //la tarifa de gas pasa a ser 3
System.out.print(c1); //Nombre: Ana Dirección: Uría 6, 2ºI
                        //Contrato Luz No: 123879 Tarifa: 2
                        //Contrato Gas No: 345536 Tarifa: 3
                        //Consumo: 35.6
```

Solución:

```
public class Cliente {
    //Atributos
    private String nombre, dirección;
    private Contrato luz, gas;
    private float consumo;
    //Constructor
    public Cliente(String nombre, String dirección,int ncLuz,
                   int tLuz, int ncGas, int tGas) {
        this.setNombre(nombre);
        this.setDirección(dirección);
        this.luz = new Contrato(ncLuz,tLuz);
        this.gas = new Contrato(ncGas,tGas);
        this.setConsumo(0);
    }
    //La cantidad debe ser positiva y el no de contrato válido
    public void cargarFactura(int nc, int importe) {
        if ( (this.getLuz().getNúmero() == nc ||
              this.getGas().getNúmero() == nc ) && importe > 0)
            this.setConsumo( this.getConsumo() + importe );
    }
    //Se comprueba que el núm. de contrato es válido
    public void cambiarTarifa(int nc, int tarifa) {
        if (this.getLuz().getNúmero() == nc )
            this.getLuz().setTarifa(tarifa);
        else if (this.getGas().getNúmero() == nc )
                this.getGas().setTarifa(tarifa);
    }
    //imprime todos los datos
    @Override
    public String toString(){
       String s=String.format("Nombre: %s", this.getNombre());
       s+=String.format("\tDirección: %s", this.getDirección());
       s+=String.format("\nContrato Luz %s", this.getLuz(),
       s+=String.format("\nContrato Gas %s", this.getGas(),
       s+=String.format("\nConsumo %f", this.getConsumo());
       return s;
    }
}
```