Apellidos:

Nombre: D.N.I.:

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

14 de Enero de 2021

1. (2 %) Escribe una expresión para declarar un vector de números reales cuyo tamaño se lee de un objeto Scanner llamado `teclado`, que supondremos definido previamente.

**Solución:**

```
double[] v = new double[teclado.nextInt()];
```

2. (5 %) Dada la siguiente definición de variables y sus valores iniciales,

```
int x=9, y=3; double f=2.0; char c='a';
```

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 |
|---|---|---|---|---|
| `f = f / f;` | ☒Si ☐No | double | 1.0 | |
| `f = 1 / y;` | ☒Si ☐No | double | 0.0 | |
| `1 / y` | ☒Si ☐No | int | 0 | |
| `(char)(c+y)` | ☒Si ☐No | char | 'd' | |
| `c == 'c'` | ☒Si ☐No | boolean | `false` | |
| `y = c == 'a'` | ☐Si ☒No | | | No se puede asignar un `boolean` a una variable `int` |

3. (12 %) Implementa el método estático `piAprox()` para calcular (y retornar) de forma aproximada el valor de $\pi$. Para ello debes utilizar la serie conocida como *Producto de Wallis*:

$$\frac{2}{1}\cdot\frac{2}{3}\cdot\frac{4}{3}\cdot\frac{4}{5}\cdot\frac{6}{5}\cdot\frac{6}{7}\cdot\frac{8}{7}\cdot\frac{8}{9}\cdot\frac{10}{9}\cdot\frac{10}{11}\cdots=\frac{\pi}{2}$$

El método debe recibir un parámetro $n \geq 1$ que indica el número de términos del producto que debemos utilizar para calcular la aproximación. Por ejemplo, si se invoca `double myPi = piAprox(5)` el método debería retornar el resultado de utilizar los 5 primeros términos del producto, es decir,

$$\pi = 2\cdot\left(\frac{2}{1}\cdot\frac{2}{3}\cdot\frac{4}{3}\cdot\frac{4}{5}\cdot\frac{6}{5}\right)$$

**Solución:**

```
public static double piAprox(int n) {
    int numerador = 2;
    int denominador = 1;
    double producto = 1.0;
    for (int i = 1; i <= n; i++) {
        producto *= (double) numerador / denominador;
        // System.out.printf("%d) %d/%d\n", i, numerador, denominador);
        if (i%2 == 0)
            numerador += 2;
        else
            denominador +=2;
    }
    return 2 * producto;
}
```

4. (12 %) Diseñar el método público y estático `Seguidos` que, dado un vector no vacío de números enteros, calcule *la **longitud** de la subsecuencia más larga de números consecutivos e iguales.*

   Ejemplo. Para el vector 2, 1, 1, 2, 2, 5, 4, 4, 4 el método debe retornar 3 (ya que el número 4 aparece 3 veces seguidas, no hace falta retornar el 4, solamente el 3)

**Solución:**

```
public static int Seguidos(int[] v) {
    int max = 1;
    int cont = 1;
    int i = 1;
    while ( i < v.length ) {
       if (v[i] == v[i-1]) {
          cont++;
          if ( cont > max ) max = cont;
       }
       else cont = 1;
       i++;
    }
    return max;
}
```

5. (8 %) Escribir un programa completo en el que se pidan por teclado tres números reales a modo de longitudes de lados y se indique si podrían formar un triángulo o no. Además, en caso afirmativo, el programa debe indicar qué tipo de triángulo sería: *equilatero* (tres lados iguales), *isósceles* (dos lados iguales) o *escaleno* (tres lados distintos).

**Solución:**

```
import java.util.Scanner;

public class Triángulos {
    public static void main(String[] args) {
        Scanner t = new Scanner(System.in);
        System.out.print("Introduce tres números reales:");
        int a = t.nextDouble();
        int b = t.nextDouble();
        int c = t.nextDouble();

        // ¿forman un triángulo?
        if ((a+b <= c) || (a+c <= b) || (b+c <= a))
            System.out.println("No forman un triángulo");
        else
            if ((a == b) && (b == c))
                System.out.println("Triángulo equilátero");
            else
                if ((a == b) || (a == c) || (b == c))
                    System.out.println("Triángulo isósceles");
                else
                    System.out.println("Triángulo escaleno");
    }

}
```

6. (8 %) Un almacén de fontanería ofrece descuentos en función del nivel de compra de los clientes, de acuerdo a la siguiente tabla:

| | Herramientas | Materiales |
|---|---|---|
| Ocasional | 5% | 5% |
| Habitual | 10% | 20% |
| Profesional | 15% | 20% |

Diseña el esquema condicional que determine el descuento a aplicar en función del tipo de cliente y el producto comprado. Puedes suponer que partes de una variable `cliente` cuyo valor puede ser `'O'`, `'H'` o bien `'P'`, y de una variable `'producto'` cuyo valor puede ser `'H'` o `'M'`.

**Solución:**

```
if (cliente == 'O')
    descuento = 5;
else
    if (producto == 'M')
        descuento = 20;
    else
        if (cliente == 'H')
            descuento = 10;
        else
            descuento = 15;
```

7. (15 %) Diseñar el método público y no estático `Alternadas` que dado un String que contiene una palabra devuelva `true` si las vocales y consonantes están alternadas, es decir, que no hay dos vocales, ni dos consonantes seguidas, y `false` en caso contrario. Se permite usar, sin programarlo, el método `public boolean esVocal(char )` que recibe un parámetro de tipo `char` y devuelve cierto si es una vocal y falso en caso contrario.

Ejemplos. "hola" → `true`, "adiós" → `false`, ya que hay 2 vocales seguidas, "palabra" → `false`, hay dos consonantes (*b* y *r*) seguidas.

**Solución:**

```
public boolean Alternadas(String s) {
    i = 1;
    while ( i < s.length() &&
            esVocal(s.charAt(i)!= esVocal(s.charAt(i-1)) {
       i++;
    if (i < s.length()) return false;
    else return true;
}
```

8. (10 %) Escribir el método público y no estático `Abajo` que, dada una matriz de enteros no vacía, la modifica, desplazanado hacia abajo todos los valores de cada columna que son distintos de 0, quedando por tanto abajo los valores distintos de 0 y arriba los ceros.

Antes

| | | | | |
|---|---|---|---|---|
| 0 | 3 | 6 | 1 | 2 |
| 2 | 3 | 6 | 0 | 0 |
| 0 | 0 | 0 | 12 | 0 |
| 0 | 0 | 0 | 0 | 0 |

$\Longrightarrow$

Después

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 3 | 6 | 1 | 0 |
| 2 | 3 | 6 | 12 | 2 |

**Solución:**

```
public void abajo(int [][] m) {
   for (int j = 0; j < m[0].length); j++) {
       int i = m.length - 1,
       sgte = m.length - 2;
       while ( sgte >= 0 ) {
          if ( m[i][j] == 0 ) {
             if ( m[sgte][j] != 0 ) {
                m[i][j] = m[sgte][j];
                m[sgte][j] = 0;
                i--; sgte--;
             }
             else sgte--;
          }
          else {
             i--; sgte--;
          }
       }
   }
}
```

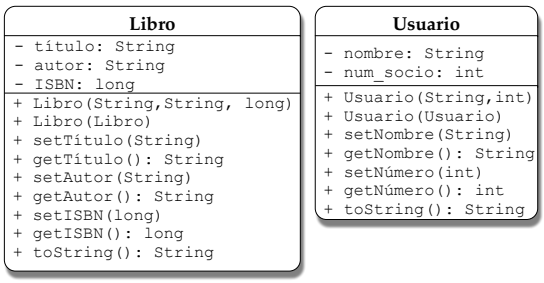
9. (16 %) Implementa la clase `Colchón` para representar colchones. La información que debe recoger cada objeto de esta clase consiste en las dimensiones en cm. (largo, ancho y grosor) así como el tipo de material, que puede ser un valor entero entre 1 y 4 (1=espuma, 2=visco, 3=látex, 4=muelles)

- Constructores: por defecto (con los siguientes valores para cada atributo: largo=190, ancho=90, grosor=21, tipo=1), el de copia, con 1 entero para inicializar el tipo (los otros tres atributos tomarán los valores por defecto antes indicados), con 4 enteros para poder inicializar los 4 atributos.
- Métodos `get()` y `set()`. Ten en cuenta que el largo sólo puede ser 180, 190 o 200, el ancho sólo puede ser 90, 135 o 200, el grosor será de, al menos, 10 cm. y el tipo sólo puede ser 1, 2, 3 o 4.
- Método `calculaVolumen()`, que calcula el volumen del colchón.
- Método `esMásVoluminoso()` que recibe un objeto `Colchón` y devuelve cierto si el objeto con el que se llama al método tiene un volumen mayor que el objeto que se pasa como parámetro, y falso en caso contrario.
- Método `toString()`, que devuelve un String con la información del objeto `Colchón` con el siguiente formato: `"Tipo: espuma. Medidas: 190 cm. x 135 cm. x 32 cm."`. (largo x ancho x grosor)

**Solución:**

```
public class Colchón {
    private int largo = 190, ancho = 90, grosor = 21, tipo = 1;

    public Colchón(int largo, int ancho, int grosor, int tipo) {
        setLargo(largo);
        setAncho(ancho);
        setGrosor(grosor);
        setTipo(tipo);
    }

    public Colchón() {
        this(190, 90, 21, 1); // también podríamos no hacer nada, puesto que ya están inicializados correctamente
    }

    public Colchón(int tipo) {
        this(190, 90, 21, tipo);
    }

    public Colchón(Colchón c) {
        this(c.getLargo(), c.getAncho(), c.getGrosor(), c.getTipo());
    }

    public int getLargo() {  return largo; }
    public int getAncho() {  return ancho; }
    public int getGrosor() { return grosor; }
    public int getTipo() {  return tipo;  }

    public void setLargo(int largo) {
        if (largo == 180 || largo == 190 || largo == 200)
            this.largo = largo;
    }
    public void setAncho(int ancho) {
        if (ancho == 90 || ancho == 135 || ancho == 200)
            this.ancho = ancho;
    }
    public void setGrosor(int grosor) {
        if (grosor >= 10)
            this.grosor = grosor;
    }
    public void setTipo(int tipo) {
        if (tipo > 0 && tipo < 5)
            this.tipo = tipo;
    }

    public int calculaVolumen() {
        return this.getLargo() * this.getAncho() * this.getGrosor();
    }

    public boolean esMásVoluminoso(Colchón c) {
        return this.calculaVolumen() > c.calculaVolumen();
    }

    @Override
    public String toString() {
        String material = "";
        switch(this.getTipo()) {
        case 1: material = "espuma"; break;
        case 2: material = "visco"; break;
        case 3: material = "látex"; break;
        case 4: material = "muelles"; break;
        }
        return String.format("Tipo: %s. Medidas %d cm. x %d cm. x %d cm.",
               material, this.getLargo(), this.getAncho(), this.getGrosor());
    }
}
```

10. (12 %) Se pide implementar la clase **Préstamo**, que debe registrar el **usuario**, el **libro** y la fecha (**día**, **mes** y **año**) del préstamo realizado en una biblioteca. Teniendo en cuenta que las clases **Usuario** y **Libro** ya están implementadas y se corresponden con los diagramas adjuntos, implementa la clase **Préstamo** con dos constructores y el método **toString()** para que el programa que se que se incluye a modo de ejemplo funcione correctamente y produzca la salida mostrada más abajo (no es necesario que implementes los get() y set(), aunque puedes usarlos como si estuviesen implementados. Tampoco es necesario controlar si la fecha es correcta).




```
public class ProgramaBiblioteca {

   public static void main(String[] args) {
      Usuario u1 = new Usuario("Pepito Pérez", 647593);
      Libro l1 = new Libro("Patria", "Fernando Aramburu", 9788490663196L);

      Préstamo p1 = new Préstamo(u1, l1, 10, 1, 2021);
      Préstamo p2 = new Préstamo("Manuela Felgueroso", 456753,
                                 "Los pilares de la tierra", "Ken Follett", 9788466341783L,
                                 14, 1, 2021);

      System.out.printf("PRÉSTAMOS:\n%s\n%s\n", p1, p2);
   }
}
```

```
PRÉSTAMOS:
Préstamo del día 10/01/2021 a Pepito Pérez (647593):
    Título: Patria. Autor: Fernándo Aramburu. ISBN: 9788490663196
Préstamo del día 14/01/2021 a Manuela Felgueroso (456753):
    Título: Los pilares de la tierra. Autor: Ken Follett. ISBN: 9788466341783
```

**Solución:**

```
public class Préstamo {
    private Usuario usuario;
    private Libro libro;
    private int día, mes, año;

    public Préstamo(Usuario usuario, Libro libro, int día, int mes, int año) {
        setFecha(día, mes, año);
        this.usuario = usuario; // también vale this.usuario = new Usuario(usuario);
        this.libro = libro;
    }

    public Préstamo(String nombre, int número,
                    String título, String autor, long ISBN,
                    int día, int mes, int año) {
        setFecha(día, mes, año);
        this.usuario = new Usuario(nombre, número);
        this.libro = new Libro(título, autor, ISBN);
    }

    // suponemos que existen los get() y set() necesarios

    @Override
    public String toString() {
        return String.format("Préstamo del día %02d/%02d/%04d a %s:\n\t%s",
               this.getDía(), this.getMes(), this.getAño(), this.usuario, this.libro);
    }

}
```