Apellidos:

Nombre: D.N.I.:

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

16 de Enero de 2020

1. (1 p) Declara, con una sola instrucción, un vector de cadenas de caracteres cuyos valores iniciales han de ser "Hola", "ke", "ase".

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

```
int x=9, y=2; 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
`y ** 3`	☐Si ☐No			
`c < d`	☐Si ☐No			
`y = x + 1.0;`	☐Si ☐No			
`y != c`	☐Si ☐No			
`f + 1.0f`	☐Si ☐No			
`x/2.0 + 1`	☐Si ☐No			

3. (2 p) Dado un valor $N >= 1$, contesta a las siguientes preguntas

   (a) ¿Qué imprime el siguiente código?

```
for (int i = 1, a = 1; i <= N; i++, a += i) {
    System.out.printf("%d %d\n", i, a);
}
```

   (b) ¿Cuánto vale i justo después de salir del bucle?

   (c) ¿Cuánto vale N justo después de salir del bucle?

   (d) El código siguiente ¿produce la misma salida que el del apartado a? ¿por qué?

```
int i = 1, a = 1;
while (i <= N) {
    System.out.printf("%d %d\n", i, a);
    a = a + i;
    i = i + 1;
}
```

   (e) ¿Cuánto vale i justo después de salir del bucle?

4. (3 p) Implementa un método estático que, dado un número entero N, calcule el resultado de la serie siguiente **sin utilizar `Math.pow()`**:

$$\sum_{i=0}^{N}(-1)^i\frac{1}{2^i} = 1 - \frac{1}{2} + \frac{1}{4} - \frac{1}{8} + \frac{1}{16} - \cdots$$

5. (2 p) Escribe en Java las condiciones equivalentes a las siguientes expresiones:

   La matriz M no es cuadrada y su número de filas es par

   El número entero n tiene más de m dígitos

   Los números reales x e y son aproximadamente iguales (con un margen de error de 1 milésima)

   Las variables enteras a, b, c y d contienen las longitudes de los lados de un rectángulo o romboide (lados iguales dos a dos) pero NO de un cuadrado ni de un rombo (cuatro lados iguales)

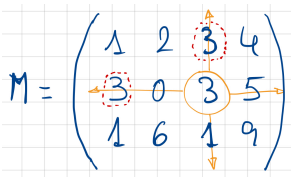
6. (3 p) Haz un programa completo (incluyendo las librerías necesarias) que pida una cantidad en € y muestre por pantalla el mínimo número de monedas necesarias para alcanzar dicho importe (recuerda que las monedas de curso legal son de 2€, 1€, 50c, 20c, 10c, 5c, 2c y 1c). Se recomienda encarecidamente trabajar con números enteros, pasando los euros a céntimos, para evitar errores de redondeo.

   Ejemplo:

```
Importe en EUR: 11,47
5 monedas de 2€
1 monedas de 1€
2 monedas de 20c
1 monedas de 5c
1 monedas de 2c
```

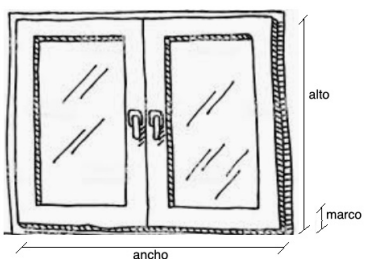
7. (3 p) Implementa el método **`primerPrimo()`** que recibe un vector de enteros como parámetro y retorna el índice más pequeño de la componente que contiene un número primo, o -1 si no hay ninguna. Se recomienda la implementación del método auxiliar **`esPrimo()`** que retorna cierto cuando se le pasa un número primo como parámetro y falso cuando éste no es primo.

8. (3 p) Si definimos como *área de influencia* de un elemento en una matriz a aquellos elementos que están por encima, por debajo, a su izquierda y a su derecha, se pide que implementes un método que, dado un determinado elemento de una matriz de enteros, retorne el número de elemento en su área de influencia que sean iguales a él. Ejemplo: para el elemento de la fila 1, columna 2 de la matriz M el método debería retornar el valor 2.



9. (4 p) Se pide hacer la clase `Ventana` para representar ventanas de 2 hojas. La clase tiene tres atributos enteros con las dimensiones en cm de la ventana: ancho, alto, marco.

- Constructores: por defecto (con valores para cada atributo de 120, 100, 6), el de copia, con 2 enteros para inicializar ancho y alto (por defecto para el marco: 6 cm), con 3 enteros para poder inicializar los 3 atributos.
- Métodos `get()` y `set()`.
- Método `calculaSuperficieCristales()`, que devuelve el valor de la siguiente fórmula:
$$(ancho - 4 \cdot marco) \cdot (alto - 2 \cdot marco).$$
- Método `tieneMásCristal()` que recibe un objeto Ventana y devuelve cierto si el objeto con el que se llama al método tiene una superficie de cristal 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 Ventana, por ejemplo, "120 cm. ancho x 100 cm. alto, Marco 6 cm.".



10. (3 p) Implementa la clase **`PrismaUniforme`** para representar prismas rectos cuyas bases son polígonos regulares y cuyas caras son cuadrados. Un prisma recto requiere, por tanto dos atributos de tipo `PolígonoRegular`:

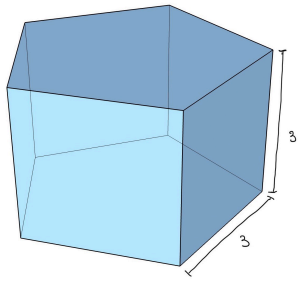
- `base`: representa las bases (como ambas son iguales, basta almacenar una)
- `cara`: representa las caras (como todas son iguales, basta almacenar una)

Deberás implementar el código necesario para que el siguiente programa produzca la salida que se muestra, excepto el código de la clase `PolígonoRegular`, que se presupone ya correctamente implementada. (Nota: Si p es un `PolígonoRegular` de 3 lados de longitud 1 unidad, la instrucción `System.out.println(p)` produce la salida: `Polígono regular de 3 lados de longitud 1,00 (Área: 0,43, Perímetro: 3,00)`)

PolígonoRegular
– n_lados: int – longitud: double
+ PolígonoRegular(int, double) + setNLados(int) + setLongitud(double) + getNLados(): int + getLongitud(): double + perímetro(): double + apotema(): double + área(): double + toString(): String

```
public class Ejemplo {
    public static void main(String[] args) {
        PrismaUniforme r = new PrismaUniforme(5, 3.0);
        System.out.println(r);
    }
}
```

```
Bases(2): Polígono regular de 5 lados de longitud 3,00 (Área: 15,48, Perímetro: 15,00)
Caras(5): Polígono regular de 4 lados de longitud 3,00 (Área: 9,00, Perímetro: 12,00)
Área Total: 75,97
Volumen: 46,45
```



Prisma recto uniforme con base pentagonal (5 lados de longitud 3 unidades)