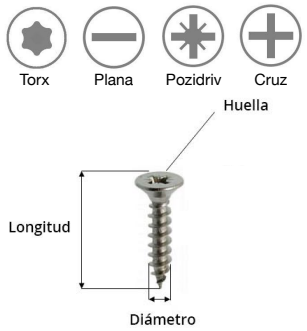
Alumno/a: _____

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

7 de Julio de 2020

1. (3,25 p.) Implementa la clase `Tornillo`. Los atributos que definen un tornillo son: **longitud**, **diámetro** (ambos son números reales cuyas unidades son mm.) y **huella** (que puede ser una de los cuatro que se muestra en la imagen). Debes implementar:

   (a) Cuatro constructores:
   - por defecto (con valores longitud=30,0, diámetro=6,0 y huella "Plana")
   - con dos parámetros para la longitud y el diámetro (huella será "Cruz")
   - con tres parámetros
   - constructor copia.

   (b) Métodos `set()` y `get()`, tales que no se puedan asignar valores incorrectos a ninguno de los atributos (la huella tiene que ser una de las indicadas, y los valores numéricos deben ser mayores que 0).

   (c) Método `esCompatible()`, que devuelve cierto si la huella del tornillo forma parte o es igual a la huella de otro tornillo que se pasa como parámetro. Así, un tornillo de huella Plana es compatible con otro de Plana, de Pozidriv o de Cruz, uno de Cruz es compatible con otro de Cruz o Pozidriv. Los Pozidriv y los Torx sólo son compatibles con otro de su misma huella.

   (d) Método `toString()`, que retorne un Strign con la información de los atributos del tornillo.

2. (0,50 p.) Escribir un trozo de código (no un programa completo), donde se creen cuatro objetos de la clase `Tornillo`, uno con cada constructor, denominados `t1`, `t2`, `t3`, y `t4`. El código debe imprimir la información de uno de ellos y también se debe comprobar la compatibilidad del `t1` con el `t4`.

3. (1,50 p.) Implementa el método público y estático `extenderVector()` que recibe como parámetro un vector de números reales y retorna otro vector de mayor tamaño, de manera que entre cada par de componentes del original se intercala su media.

   Ejemplo: vector original: {3.5, 4.5, 8} $\longrightarrow$ vector devuelto por el método: {3.5, 4.0, 4.5, 6.25, 8.0}

4. (2,75 p.) Se pide hacer un programa de consola completo (importando las clases necesarias) para procesar una secuencia de datos meteorológicos usados para generar estadísticas en clubes de tenis. El programa debe pedir, en primer lugar, el nº de días para los que se van a introducir datos y después debe leer los datos meteorológicos de cada día. Los datos meteorológicos de cada día están formados por tres valores:
   - Clima: puede tomar 3 valores, *Soleado*, *Lluvioso* o *Nublado*,
   - Humedad relativa: un valor real entre 0 y 100, y
   - Velocidad del viento: un valor entero en Km/hora.

   Después de leer los datos meteorológicos de todos los días el programa debe imprimir las siguientes estadísticas:
   - Porcentaje de días en los que no llueve.
   - Porcentaje de días soleados.
   - Porcentaje días en los que el viento es flojo o moderado (cuando la velocidad del viento es menor o igual que 38 Km/h).
   - Porcentaje de días en las que la humedad es alta (superior al 60%).
   - Porcentaje de días aptos para jugar al tenis. Un día es adecuado para jugar al tenis cuando o bien el clima es Nublado, o bien el Clima es Soleado y la Humedad no es alta, o bien el Clima es Lluvioso pero el viento es flojo o moderado.

Ejemplo:

Entrada de datos:
Número de días: 6
Datos:
Nublado 50.5 25
Soleado 80.2 15
Soleado 50.1 30
Lluvioso 90.7 20
Lluvioso 80.4 50
Nublado 45.2 37

Salida:
% de días sin lluvia: 66.66%
% de días soleados: 33.33%
% de días viento flojo o moderado: 83.33%
% de días humedad alta: 50.00%
% días aptos para el tenis: 66.66%

Notas: **NO SE PUEDEN USAR VECTORES. Los datos se deben leer y procesar según se leen**. Se valorará el diseño del programa en su conjunto, y el correcto uso de las variables y tipos elegidos para representar la información. El usuario introducirá los datos correctamente, no se debe incluir código para comprobar si son correctos.

5. (2 p.) Implementa **de manera eficiente** el método público y estático `índiceMayorCP()`, que recibe como parámetro un vector de números enteros *ordenado de menor a mayor* y retorna el **índice de la componente de mayor valor que además es un cuadrado perfecto**. En caso de que no haya ninguna componente que sea cuadrado perfecto retornará el valor $-\mathbf{1}$.

   Ejemplo: para el vector $v = \{2, 4, 5, 11, 49, 50, 100, 103\}$ el método retornará 6 puesto que el mayor cuadrado perfecto es 100 y está en $v[6]$.

   Nota: para averiguar si un número es cuadrado perfecto debes verificar que el valor de su raíz cuadrada (número real) no cambia cuando lo transformas a número entero. Por ejemplo, 103 NO es cuadrado perfecto puesto que $\sqrt{103} \neq (\text{int})\sqrt{103}$ (es decir, $10,148891565 \neq 10$). Sin embargo, 100 SI es un cuadrado perfecto ya que $\sqrt{100} = (\text{int})\sqrt{100}$ $(10,000000 = 10)$.