Práctica Calificada 4

Curso: CC211 - A1 **Ciclo: 2019.1**

Debe enviar UN SOLO ARCHIVO de nombre `Exam.java`, no más; SE REVISARÁ SOLAMENTE EL ARCHIVO `Exam.java`.

1. (3 ptos.) Cree una clase llamada `Point` para realizar operaciones vectoriales usuales del plano cartesiano. Use variables `double` para representar las variables de instancia de la clase: `coordinate0` y `coordinate1`. Proporcione un constructor que permita que un objeto de esta clase se inicialice al ser declarado. Proporcione métodos `static` que realicen cada una de las siguientes operaciones:

   a) retorne el producto interno de dos objetos `Point`

   b) retorne la resta vectorial de dos objetos `Point`

   c) retorne la distancia de dos objetos `Point`

2. (7 ptos.) Defina una clase llamada `Triangle` para representar a un triángulo del plano cartesiano. Los atributos de un triángulo son sus tres vértices y sus tres lados. Proporcione un constructor que reciba tres argumentos del tipo `Point` y verifique si dicha terna son los vértices de un triángulo; caso contrario, se debe lanzar una excepción del tipo `IllegalArgumentException`. Proporcione métodos `public` que verfiquen si dicho triángulo es:

   a) recto

   b) equilatero

   También proporcione métodos `getPerimeter` y `getArea` que retornen el perímetro y el área, respectivamente, de dicho triángulo. [**Sugerencia:** Sean $v0$, $v1$ y $v2$ vectores de $\mathbb{R}^2$. Sean $u := (u_1, u_2) := v1 - v0$ y $v := (v_1, u_2) := v2 - v0$. La teoría dice que el área del paralelogramo formado por los vectores u y v está dado por

   $$S := \frac{|u_1 v_2 - u_2 v_1|}{2}.$$

   S es igual a cero si y solamente si los puntos $v0$, $v1$ y $v2$ NO forman un triángulo, i.e., son colineales.]

3. (4 ptos.) Implemente una clase llamada `RightTriangle` como una extensión de `Triangle`. Proporcione un constructor que reciba tres argumentos del tipo `Point` y verifique si dicha terna son los vértices de un triángulo rectángulo; caso contrario, se debe lanzar una excepción del tipo `IllegalArgumentException`. Sobrescriba el método `getArea` con una implementación distinta a la de la clase `Triangle`.

4. (4 ptos.) Implemente una clase llamada `EquilateralTriangle` como una extensión de `Triangle`. Proporcione un constructor que reciba tres argumentos del tipo `Point` y verifique si dicha terna son los vértices de un triángulo equilátero; caso contrario, se debe lanzar un excepción del tipo `IllegalArgumentException`. Sobrescriba los métodos `getArea` y `getPerimeter` con implementaciones distintas a las de la clase `Triangle`.

5. (2 ptos.) Cree

   a) un objeto del tipo `RightTriangle` de vértices $(0, 0)$, $(3, 0)$, $(0, 4)$ y

   b) un objeto del tipo `EquilateralTriangle` de vértices $(-1, 0)$, $(1, 0)$, $(0, \sqrt{3})$.

   Finalemente, muestre la representación en cadena de dichos objetos:

```
Right triangle
vertex0 = (0.00,0.00)
vertex1 = (0.00,4.00)
vertex2 = (3.00,0.00)
edge0 = 5.00
edge1 = 3.00
edge2 = 4.00
area: 6.00
perimeter: 12.00

Equilateral triangle
vertex0 = (1.00,0.00)
vertex1 = (0.00,1.73)
vertex2 = (-1.00,0.00)
edge0 = 2.00
edge1 = 2.00
edge2 = 2.00
area: 3.46
perimeter: 6.00
```

20 de mayo de 2019

`Print only when necessary`.