Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica

IE-0217 Estructuras de abstractas datos y algoritmos para ingeniería

Laboratorio 2: Herencia, polimorfismo, sobrecarga y plantillas

M. Sc. Ricardo Román Brenes - ricardo.roman@ucr.ac.cr

II-2019

Tabla de contenidos

---

1. Enunciado

Diseñe, desarrolle y documente (Doxygen) una serie de clases que modelen figuras geométricas en dos dimensiones.

1.1. Vértices

Cree una clase **Vertice**, la cual modele un punto en un espacio 2D. Esta clase debe tener como atributos:

- float x
- float y
- unsigned int identificador

Y como métodos:

- Vertice()
- ~Vertice()
- string operator () // Transforma el objeto en una representación en texto.
- double operator>>(const Vertice& rhs) // Calcula la distancia entre 2 vértices.

1.2. Clase base

Cree una clase base llamada **Figura** que tenga como atributos:

- string nombre.
- string color.
- unsigned int identificador.
- Vectice* puntos
- Un número **estático** autoincrementado cada vez que se genera una nueva figura.

Y como métodos:

- Figura()
- ~Figura()
- double superficie() = 0
- double perimetro() = 0
- string operator () = 0 //transforma el objeto en una representación en texto.

1.3. Clases derivadas

Además cree clases para las siguientes figuras:

- Círculo
- Rectángulo
- Triangulo

Estas clases heredan de la clase Figura y contienen sus características propias (equiláteros, radio, largo del lado...) y además reimplementan los métodos virtuales puros de la clase base.
Además la clase triángulo debe tener otras 3 especializaciones:

- Equilátero
- Isósceles
- Escaleno

Que a su vez reimplementan los metodos virtuales puros que la clase Triangulo les pasa de Figura.

1.4. Sobrecarga de operadores

Cree una clase llamada `Impresora`, la cual está emplantillada con un tipo **T**. Esta clase se encarga de imprimir los objetos, sin importar su tipo, así como escribirlos en archivos de texto. Así, sus métodos son:

- void imprimirObjeto(const T& objeto)
- void escribirObjeto(const T& objeto, string rutaArchivo)

También programe un constructor por copia y el operador $=$ para todas las clases.

1.5. Diagrama de clases

Por último construya el diagrama de clases de estas cuatro clases siguiendo el estándar de UML[1].

2. Consideraciones

- Haga grupos de 2 o 3 personas.
- Genere un reporte PDF en LaTeX que incluya su código, el diagrama de clases, y sus conclusiones.
- Suba su código y documentación (doxygen, README, INSTALL) al GitLab respectivo de su grupo y laboratorio.
- Cada estudiante debe subir el reporte a Schoology
- Recuerde que por cada día tardío de entrega se le rebajaran puntos de acuerdo con la formula: 4^d, donde $d > 1$ es la cantidad de días tardíos.

[1] `https://en.wikipedia.org/wiki/Unified_Modeling_Language`