Tecnología de la Programación **10 de mayo de 2023**
Duración: 3h Evaluación Final.

NOTA IMPORTANTE: Aquellos ejercicios, que, a pesar de estar resueltos correctamente, no estén razonados por el alumno, se considerarán no válidos. Comenta el código. Justifica, en su caso, el valor de retorno de los métodos. **Gestiona siempre todos los errores.**

Apellidos y Nombre: .....

☐ Se considera la **clase** `Figura` que representa una figura geométrica en el plano cartesiano cuyo punto "central" coincide con el origen de coordenadas del plano. Esta clase **contabiliza** todas las figuras que se vayan creado de cada tipo y tiene un **método** que retorna esta información. También tiene los **métodos** para calcular el área y el perímetro, así como los **atributos** que almacenan dichos valores. Se tienen las **interfaces**: `ITransformación` que permite expandir y contraer la figura, `IModificación` que permite escalar y rotar la figura, y `ITraslación` que permite trasladar el `Punto` "central" de la figura a cualquier lugar del plano. Se definen las **clases hijas** de `Figura`: `Círculo` y `Cuadrado`. Un círculo puede expandir o contraer su radio por un factor. Un cuadrado puede escalar su lado y rotar ciertos grados (dado en radianes y en el rango $(-\pi, \pi]$). Un cuadrado tiene como clase hijas un `Rombo` (con dos diagonales, la mayor y la menor) y un `Rectángulo` (con un lado, que es la altura, y un ancho). Un rectángulo también puede trasladar su punto "central". Los círculos, cuadrados, rectángulos y rombos tienen un **método** que muestra información de la figura (su tipo y atributos) - este método no es `__str__()` pero lo usa.

☐ También queremos gestionar una lista de figuras mediante una **cola de prioridad**. Una cola de prioridad se comporta como una cola pero donde sus elementos siempre están ordenados de acuerdo a una clave (de mayor a menor) y, en este caso, la clave será **el perímetro** de la figura. Cada vez que se extrae un elemento se extrae el primero de la cola (el de mayor clave) y cada vez que se incluye un elemento se coloca en una posición acorde al orden establecido. La cola de prioridad se implementará mediante **una lista simplemente enlazada** y permitirá al usuario agregar figuras, eliminar figuras existentes y mostrar/guardar información de las figuras. Para ello tendrás en cuenta todas las clases anteriores junto con las clases `Nodo` y `ListaFiguras`.

☐ Con todo esto se pide lo siguiente:

1. **(.5 puntos)** ¿Por qué `Rectángulo` es clase derivada de `Cuadrado` y no a la inversa?
2. **(1 punto)** Construye el diagrama UML de lo que se pide indicando en cada caja sus atributos y tipo junto con los métodos que se declaren o definan. Para cada uno indica su modificador de visibilidad. Recuerda, no se reescriben en las clases hijas los miembros públicos y protegidos.
3. **(.5 puntos)** Observa que todos los atributos necesarios para las figuras son positivos, salvo las coordenadas de los puntos y los ángulos. Crea tu propia jerarquía de excepciones.
4. **(2.5 puntos)** Escribe la parte del programa que incluye todo salvo `Nodo` y `ListaFiguras`.
   (.5 puntos) Haz un uso correcto de las excepciones propias.
5. Escribe la parte del programa que incluye solo a las clases `Nodo` y `ListaFiguras`.
   - **(.5 punto)** Explica la estructura para trabajar con una lista simplemente enlazada CON cabecera.
   - **(.5 puntos)** Uso correcto de las excepciones en los siguientes apartados.
   - **(1 punto)** El método `agregar_figura()` será el encargado de añadir una nueva figura a la lista de figuras en la cola de prioridad. **NO está permitido** el uso de algoritmos de ordenación.
   - **(1 punto)** El método `eliminar_figura()`. Sin parámetros eliminará la primera figura de mayor prioridad. Con parámetro, eliminará la primera figura que coincida con el perímetro dado.
6. **(1 punto)** Construye el método `imprimir_figuras()` de `ListaFiguras`. Sin parámetro mostrará la información de todas las figuras. Con parámetro, un string, guardará la información en un el fichero dado. Indica solo qué parte de la jerarquía de las figuras requieren modificarse, SIN reescribir las clases.
7. **(1 punto)** Haz un programa donde se gestione, usando un menú de texto, la inclusión de figuras en una lista. También tendrá la opción de salir.
8. **Extra (+1 punto)** Resuelve el último apartado usando iteradores.
9. **Extra (+1 punto)** Modifica el método de eliminar para que en el caso de empate, varias figuras con el mismo perímetro, seleccione aleatoriamente a una de ellas y la elimine.

Nota: Sobreescribe todos los métodos mágicos que consideres necesarios. Cuantos más sobreescribas, mejor.
Nota: Declara los tipos en todos los métodos que uses.
`def set(x):` está mal, `def set(x:int): -> None` está bien.