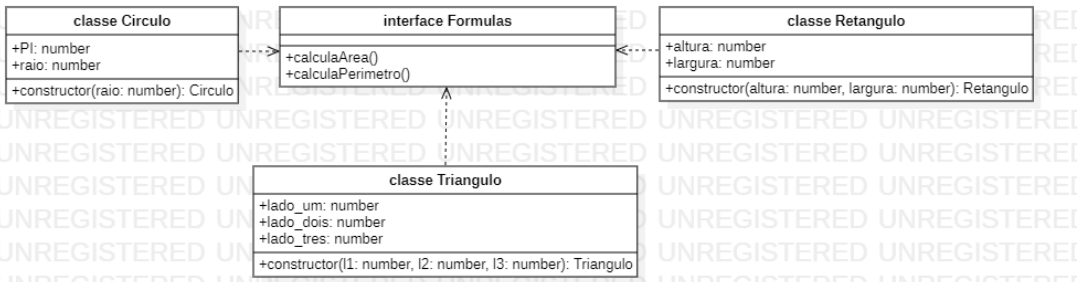
Atividade 5

1. Interface

Interface sugere um novo modelo de classe, baseado na herança, entretanto, não ocorre de fato a herança dos objetos da superclasse, mas sim uma obrigação de que uma classe que implementa a **interface** contenha os métodos definidos por si.

Imagine uma interface **Formulas**, que obrigatoriamente, define que cada classe que implemente a interface, obrigatoriamente tenha os métodos *calcularArea()* e *calcularPerimetro().*



A implementação de uma interface ligada as classes pode ser atribuída pela denotação:

`implements`

2. Desenvolvendo nosso script.

Abra o seu editor de códigos fonte e desenvolva o seguinte script no mesmo diretório que a atividade anterior:

`script-exemplo-interface-formulas.ts`

```
1.    export interface Formulas {
2.
3.        // Definindo métodos obrigatórios pela interface
4.        public calculaArea(): number;
5.        public calculaPerimetro(): number;
6.
7.    }
```

Próxima página, classe Retangulo!

script-exemplo-interface-retangulo.ts

```
8.    import { Formulas } from 'script-exemplo-interface-formulas';
9.
10.   export class Retangulo implements Formulas{
11.
12.       // Definindo atributos
13.       public altura: number;
14.       public largura: number;
15.
16.       constructor(largura: number, altura: number){
17.           this.largura = largura;
18.           this.altura = altura;
19.       }
20.
21.
22.   }
```

Próxima página classe Círculo!

script-exemplo-interface-circulo.ts

```
23.   import { Formulas } from 'script-exemplo-interface-formulas';
24.
25.   export class Circulo implements Formulas{
26.
27.       // Atributos readonly não necessitam de GET e SET
28.       // Simplesmente porque são inalteráveis
29.       public readonly PI: number = Math.PI;
30.
31.       public raio: number;
32.
33.       constructor(raio: number){
34.           this.raio = raio;
35.       }
36.
37.   }
```

Próxima página classe Triangulo!

script-exemplo-interface-triangulo.ts

```
1. import { Formulas } from 'script-exemplo-interface-formulas';
2.
3.     export class Triangulo implements Formulas{
4.
5.         // Definindo atributos
6.         public lado_um: number;
7.         public lado_dois: number;
8.         public lado_tres: number;
9.
10.        constructor(l1: number, l2: number, l3: number){
11.            this.lado_um = l1;
12.            this.lado_dois = l2;
13.            this.lado_tres = l3;
14.        }
15.
16.
17.
18.    }
```

Próxima página como implementar os métodos!

Implemente na classe Circulo os métodos da seguinte maneira:

```
    calculaArea(): number {
        return 2 * this.PI * this.raio**2;
    }

    calculaPerimetro(): number {
        return 2 * this.PI * this.raio;
    }
```

Implemente na classe Retangulo os métodos da seguinte maneira:

```
    calculaArea(): number {
        return this.altura * this.largura;
    }

    calculaPerimetro(): number {
        return (this.largura + this.altura) * 2;
    }
```

Implemente na classe Triangulo os métodos da seguinte maneira:

```
    calculaArea(): number {
        let p: number = this.calculaPerimetro() / 2;

        return (p*(p - this.lado_um)*(p - this.lado_dois)*p - (this.lado_tres) )**1/2;
    }

    calculaPerimetro(): number {
        return this.lado_um + this.lado_dois + this.lado_tres;
    }
```

FIM!