Interface

Interface é um tipo que define um conjunto de operações que uma classe (ou struct) deve implementar.

A interface estabelece um **contrato** que a classe (ou struct) deve cumprir.

```
interface IShape {

    double Area();
    double Perimeter();
}
```

Pra quê interfaces?

- Para criar sistemas com **baixo acoplamento** e **flexíveis**.

Problema exemplo

Uma locadora brasileira de carros cobra um valor por hora para locações de até 12 horas. Porém, se a duração da locação ultrapassar 12 horas, a locação será cobrada com base em um valor diário. Além do valor da locação, é acrescido no preço o valor do imposto conforme regras do país que, no caso do Brasil, é 20% para valores até 100.00, ou 15% para valores acima de 100.00. Fazer um programa que lê os dados da locação (modelo do carro, instante inicial e final da locação), bem como o valor por hora e o valor diário de locação. O programa deve então gerar a nota de pagamento (contendo valor da locação, valor do imposto e valor total do pagamento) e informar os dados na tela. Veja os exemplos.

Example 1:

```
Enter rental data
Car model: Civic
Pickup (dd/MM/yyyy hh:mm): 25/06/2018 10:30
Return (dd/MM/yyyy hh:mm): 25/06/2018 14:40
Enter price per hour: 10.00
Enter price per day: 130.00
INVOICE:
Basic payment: 50.00
Tax: 10.00
Total payment: 60.00
```

Calculations:

Duration = (25/06/2018 14:40) - (25/06/2018 10:30) = 4:10 = 5 hours
*Basic payment = 5 * 10 = 50*

*Tax = 50 * 20% = 50 * 0.2 = 10*

Example 2:

```
Enter rental data
Car model: Civic
Pickup (dd/MM/yyyy hh:mm): 25/06/2018 10:30
Return (dd/MM/yyyy hh:mm): 27/06/2018 11:40
Enter price per hour: 10.00
Enter price per day: 130.00
INVOICE:
Basic payment: 390.00
Tax: 58.50
Total payment: 448.50
```

Calculations:

Duration = (27/06/2018 11:40) - (25/06/2018 10:30) = 2 days + 1:10 = 3 days
*Basic payment = 3 * 130 = 390*

*Tax = 390 * 15% = 390 * 0.15 = 58.50*