

Funções

Criado por: Fernando de Lucas da Silva Gomes em 14 de Fevereiro de 2020
Este documento foi criado para uso pessoal em estudo e não deve ser tomado como fonte principal sobre a linguagem Swift.

Em Swift, usamos funções para executar tarefas através de uma estrutura, nela, definimos o nome da função (que será usada para "chamar" aquela função), os valores que aquela função receberá, conhecidos como parâmetros. Você também pode definir o tipo de valor que a função irá retornar, essa última instrução é opcional porém será obrigatória se na função você usar o **return** para retornar um valor daquela função.

No exemplo abaixo, criamos a função **olaPessoa** que recebe o parâmetro **pessoa** como **string** e retorna uma **string**.

```
3 func olaPessoa(pessoa: String) -> String{
4
5     let olaPessoa = "Olá, " + pessoa + "!"
6     return olaPessoa
7 }
```

Para reduzir o tamanho da função retornando já a mensagem completa.

```
3 func olaPessoa(pessoa: String) -> String{
4     return "Olá, " + pessoa + "!"
5 }
```

Funções sem parâmetros

Funções podem ser executadas sem requerir um parâmetro para ela, porém quando "chamarmos" esta função devemos colocar () mesmo que não enviemos valores

```
10 func olaMundo() -> String {
11     return "Olá mundo!"
12 }
13 print(olaMundo())
```

Funções com múltiplos parâmetros

Funções podem ter múltiplos parâmetros que podem ser escritos entre os parênteses da função separando cada parâmetro com vírgula.

Na função abaixo, a função `apresentacao` possui dois parâmetros, o parâmetro `pessoa` que recebe uma `String` e o parâmetro `jaApresentado` que recebe um `booleano`. Neste caso, podemos além disso inserir um valor padrão para o `booleano`, desta forma, certificaremos que, caso não seja passado o valor de `jaApresentado`, o sistema assumirá que ainda não haviam sido apresentados.

```
22 func olaDenovo(pessoa: String) -> String{
23     return "Olá de novo, " + pessoa + "!"
24 }
25
26 func apresentacao(pessoa: String, jaApresentado: Bool = false) -> String{
27     if (jaApresentado){
28         return olaDenovo(pessoa: pessoa)
29     }
30     else{
31         return olaPessoa(pessoa: pessoa)
32     }
33 }
34
35 print(apresentacao(pessoa: "Fernando", jaApresentado: true))
```

Funções com múltiplos retornos

Podemos também estabelecer múltiplos retornos para uma função estabelecendo diferentes retornos, no exemplo abaixo, a função recebe um array e retorna o menor e o maior valor deste array.

```
41
42 func menorAndMaior (array: [Int]) -> (min: Int, max : Int ){
43     var menorAtual = array[0];
44     var maiorAtual = array[0];
45
46     for value in array{
47         if value < menorAtual{
48             menorAtual = value
49         } else
50         if value>maiorAtual{
51             maiorAtual = value
52         }
53     }
54     return (menorAtual, maiorAtual)
55 }
56
57 var lista : [Int]
58 lista = [7,2,8,4,5,1]
59
60 print(menorAndMaior(array: lista))
```