FUNÇÕES DE ALTA ORDEM
Fácil:

1) O que caracteriza uma função de alta ordem?

A) Uma função que retorna um número.
B) Uma função que pode receber outras funções como argumentos ou retornar funções como resultado.
C) Uma função que só pode ser utilizada em um determinado escopo.
D) Uma função que não aceita parâmetros.

RESPOSTA: B

2) Qual das seguintes opções é um exemplo de uma função de alta ordem em JavaScript?

   A) `function add(a, b) { return a + b; }`
   B) `let sum = 0;`
   C) `function greet() { console.log("Hello!"); }`
   D) `function applyFunction(f, x) { return f(x); }`

   RESPOSTA: D

3) Em Python, qual função de alta ordem é utilizada para aplicar uma função a todos os itens de um iterável?

   A) `map()`
   B) `filter()`
   C) `reduce()`
   D) `apply()`

4) Qual é o efeito do seguinte código em JavaScript?

   ```
   const numbers = [1, 2, 3];

   const doubled = numbers.map(n => n * 2);
   ```

   A) `doubled` será `[2, 4, 6]`.

   B) `doubled` será `[1, 2, 3]`.

   C) O código irá gerar um erro.

   D) `doubled` será `[1, 4, 9]`.

   RESPOSTA: A

5) O que faz a função `filter()` em uma lista?

A) Ela aplica uma função a cada elemento e retorna um novo valor.
B) Ela transforma todos os elementos em um único valor.
C) Ela retorna apenas os elementos que satisfazem uma condição específica.
D) Ela ordena a lista

RESPOSTA: C

6) Qual dos seguintes exemplos ilustra o uso da função `reduce()` em JavaScript?

A) `array.reduce(x => x + 1)`
B) `array.reduce(x => x > 0)`
C) `array.reduce((acc, x) => acc + x, 0)`
D) `array.reduce(x => console.log(x))`

RESPOSTA: C

7) **No código abaixo, o que a função *`double`* faz?**

```
def double(x):
    return x * 2
numbers = [1, 2, 3]
result = list(map(double, numbers))
```

A) Retorna a soma dos números.

B) Retorna a lista original.

C) Retorna uma nova lista com os números dobrados.

D) Gera um erro.

RESPOSTA: C

8) Qual das seguintes opções é uma vantagem do uso de funções de alta ordem?

A) Aumenta a complexidade do código.

B) Permite a reutilização de código e abstração.

C) Reduz a legibilidade do código.

D) Diminui o desempenho do programa.

RESPOSTA: B

9) Qual das seguintes funções é uma aplicação comum de funções de alta ordem?

A) Aritmética básica.
B) Mapeamento de uma lista para transformar seus elementos.
C) Estruturas condicionais.
D) Declaração de variáveis.

RESPOSTA: B

10) Qual das seguintes afirmações sobre funções de alta ordem é verdadeira?

A) Elas não podem ser usadas com arrays.
B) Elas podem encapsular comportamento e criar funções personalizadas.
C) Elas não aceitam funções como argumentos.
D) Elas não podem retornar funções.

RESPOSTA: B

Média

1) O que é *closure* em programação funcional?

A) Uma função que não retorna valores.

B) Uma função que captura o ambiente em que foi criada.

C) Uma função que não pode ser chamada.

D) Uma função que cria uma nova variável.

RESPOSTA: B

2) Qual é a função de alta ordem que pode ser usada para aplicar uma função a cada elemento de uma lista e retornar um valor acumulado?

A) `map`

B) `filter`

C) `reduce`

D) `apply`

`RESPOSTA: C`

3) `Como você pode usar a função filter para obter todos os números ímpares de uma lista?`

```
A) filter( x: x % 2 == 0, lista)

B) filter( x: x % 2 != 0, lista)

C) filter( x: x > 0, lista)

D) filter( x: x < 0, lista)
```

4) Dada a função:

```
const filtrar = [1,2,3,4,5]
```

const filtrarPares= filtrar.filter(x= x % 2 == 0, números)), o que ela retorna quando chamada?

A) [1, 3, 5]

B) [2, 4]

C) [1, 2, 3, 4, 5]

D) None

RESPOSTA: B

5) Como se chama o processo de passar uma função como argumento para outra função?

A) Recursão

B) Composição de funções

C) Higher-order function

D) Encapsulamento

RESPOSTA: C

6) Dada a lista `[1, 2, 3]`, qual é a saída de `lista.map(( x= x**2), [1, 2, 3])`?

A) `[1, 2, 3]`

B) `[1, 4, 9]`

C) `[2, 3, 6]`

D) `[3, 2, 1]`

`RESPOSTA: B`

7) Em Haskell, qual das seguintes expressões representa uma função de alta ordem que recebe outra função como argumento?

A) `f x = x + 1`
B) `apply f x = f x`
C) `g = \x -> x * 2`
D) `let x = 5`

`RESPOSTA: B`

8) O que a função reduce requer como seu primeiro argumento?

A) Um iterável.

B) Uma função que aceita dois argumentos.

C) Uma lista de números.

D) Uma string.

RESPOSTA: B

9) Qual a diferença entre map e filter?

A) map retorna uma lista, filter não retorna nada.

B) map aplica uma função a todos os elementos, enquanto filter aplica uma condição.

C) map aceita apenas uma função, enquanto filter aceita várias.

D) Não há diferença, são a mesma coisa.

RESPOSTA: B

10) Quais das seguintes opções são exemplos de funções de alta ordem?

A) `sum`, `max`, `min`

B) `filter`, `map`, `reduce`

C) `print`, `input`, `str`

D) `def`, `return`, `lambda`

RESPOSTA: B

Difícil:

1)O que acontece quando você passa uma função como argumento para outra função em JavaScript?

A) A função é executada imediatamente

B) A função é armazenada como um valor e pode ser chamada dentro da função que a recebeu

C) A função é ignorada

D) A função não pode ser passada como argumento

2) O que ocorre se a função filter é aplicada a uma lista com todos os elementos falsos?

A) Retorna a lista original.

B) Retorna `None`.

C) Retorna uma lista vazia.

D) Retorna uma lista com `False`.

RESPOSTA: C

3) O que acontece quando uma função pura é usada como argumento de uma função de alta ordem?

A) O comportamento da função muda.

B) A função pode ser chamada múltiplas vezes sem efeitos colaterais.

C) A função não pode ser usada em mapeamentos.

D) A função deve retornar None.

RESPOSTA: B

4) Qual é o resultado da seguinte operação

const arr = [1, 2, 3]

const result = arr.reduce((acc, curr) => acc + curr, 0)

console.log(result);?

A) 0

B) 3

C) 6

D) 1

RESPOSTA: C

5) Qual é a saída do seguinte código:

const data = [1, 2, 3];

*const result = data.filter(n => n > 2).map(n => n * 2);*

console.log(result);?

A) [2]

B) [6]

C) [4, 6]

D) [ ]

RESPOSTA: B

6) O que acontece se você tentar usar `reduce` em um array vazio sem valor inicial?

A) Retorna `undefined`

B) Lança um erro

C) Retorna `null`

D) Retorna um array vazio

RESPOSTA: B

7) Qual é o resultado de:

const fn = (x) => (y) => x + y;

const add5 = fn(5)

console.log(add5(10));

A) 15

B) 10

C) 5

D) NaN

RESPOSTA: A

8) Em JavaScript, qual função de alta ordem pode ser utilizada para transformar todos os elementos de um array?

A) `filter`

B) `reduce`

C) `map`

D) `every`

Qual é o resultado da seguinte expressão:

const result = [1, 2, 3].map(x => x * 2).filter(x => x > 3) ?

A) [2, 4, 6]

B) [4, 6]

C) [6]

D) [ ]

RESPOSTA: B

10) Como você pode criar uma função que pode ser chamada várias vezes com o mesmo argumento em JavaScript?

A) Usando bind

B) Usando closure

C) Usando this

D) Usando prototype

RESPOSTA: B