CURRYING

Fácil:

1)O que é currying?

A) Um tipo de estrutura de dados

B) Uma técnica que transforma uma função que toma múltiplos argumentos em uma sequência de funções que tomam um único argumento

C) Um método de ordenação de listas

D) Uma técnica de recursão

RESPOSTA: B

2)Qual é a principal vantagem do currying?

A) Reduzir o consumo de memória

B) Facilitar a reutilização de funções

C) Aumentar a velocidade do programa

D) Melhorar a legibilidade do código

RESPOSTA: B

3)Qual das seguintes expressões em JavaScript representa uma função curried?

A) const add = (x, y) => x + y;

B) const add = x => y => x + y;

C) const add = x, y => x + y;

D) const add = x => { return y => x + y; };

RESPOSTA: B

4)Qual é a saída da expressão const add = x => y => x + y; add(5)(3)?

A) 5

B) 3

C) 8

D) undefined

RESPOSTA: C

5)O que acontece se você aplicar uma função curried com menos argumentos do que ela espera?

A) Retorna um erro

B) Retorna null

C) Retorna uma nova função que espera os argumentos restantes

D) Retorna um valor padrão

RESPOSTA: C

6)Como você faria uma função curried que soma três números em JavaScript?

A) const addThree = (x, y, z) => x + y + z;

B) const addThree = (x) =>(y) => (z) => x + y + z;

C) const addThree = x,y,z => (x + y + z);

RESPOSTA: B

7)O que significa aplicar uma função parcialmente?

A) Chamar a função com todos os argumentos

B) Chamar a função com alguns argumentos e retornar uma nova função

C) Ignorar os argumentos

D) Fazer uma operação assíncrona

8) Como você chamaria uma função divide que aceita dois argumentos, se ela estiver curried?

A) divide(10, 2)

B) divide(10)(2)

C) divide 10 2

D) divide[10, 2]

RESPOSTA: B

9) Como você poderia usar currying para criar uma função que subtrai um número fixo?

A) const subtractFixed = x => y => y - x;

B) const subtractFixed = x => y => x - y;

C) const subtractFixed = y => x => x - y;

D) const subtractFixed = (x, y) => x - y;

RESPOSTA: A

10) Qual é um exemplo de uso prático de currying?

A) Criar funções de configuração

B) Manipular strings

C) Realizar operações assíncronas

D) Todos os anteriores

RESPOSTA: D

MÉDIAS:

1) Qual é o papel da função bind() em relação ao currying?

A) Executar uma função imediatamente

B) Criar uma nova função com alguns argumentos pré-definidos

C) Transformar uma função em um array

D) Retornar o escopo de uma função

RESPOSTA: B

2) Como você descreveria uma função que aceita outra função como argumento e a retorna curried?

A) Função pura

B) Função de ordem superior

C) Função anônima

D) Função recursiva

RESPOSTA: B

3) Qual das seguintes opções representa uma aplicação parcial correta em JavaScript?

A) const partial = (fn, ...args) => (...remainingArgs) => fn(...args, ...remainingArgs);

B) const partial = fn => (...args) => fn(args);

C) const partial = (fn, a, b) => fn(a, b);

D) const partial = (fn) => fn();

RESPOSTA: A

4) Em um código de currying, o que `func(1)(2)(3)` representa?

A) Uma função que soma 1, 2 e 3

B) Uma função que retorna 1

C) Uma função que não pode ser chamada

D) Uma função que retorna 0

RESPOSTA: A

5) Qual é o conceito oposto ao currying?

A) Função de ordem superior

B) Aplicação parcial

C) Uncurrying

D) Recursão

RESPOSTA: C

6) O que será impresso no console se o seguinte código for executado?

```
const add = x => y => x + y;

const add5 = add(5);

console.log(add5(10));
```

A) 5

B) 10

C) 15

D) 0

RESPOSTA: C

7) Qual é a diferença entre bind() e curry()?

A) bind() altera o contexto de uma função, enquanto curry() transforma a forma como a função é chamada

B) Ambos são iguais

C) curry() altera o contexto de uma função, enquanto bind() transforma a forma como a função é chamada

D) Nenhum dos dois é útil

RESPOSTA: A

8) O que será impresso no console com o seguinte código?

```
const add = x => y => z => x + y - z;

console.log(add(6)(2)(3));
```

A) 6

B) 3

C) 1

D) 5

RESPOSTA: D

9)Qual das seguintes é uma abordagem comum para currying em JavaScript?

A) Usar funções nomeadas

B) Usar funções anônimas

C) Usar métodos de array

D) Usar classes

RESPOSTA: B

10) O que será impresso neste código?

```
const subtract = a => b => a - b;

const subtractFrom10 = subtract(10);

console.log(subtractFrom10(4));
```

A) 4

B) 6

C) 10

D) 14

RESPOSTA: B

Difíceis

1) Qual é o impacto do currying na legibilidade do código?

A) Torna o código menos legível

B) Pode melhorar a legibilidade ao expor operações mais simples

C) Não tem impacto na legibilidade

D) Apenas torna o código mais longo

RESPOSTA: B

2) Qual das seguintes linguagens de programação suporta currying nativamente?

A) Java

B) Python

C) Haskell*

D) C++

RESPOSTA: C

3)Em Haskell, como você define uma função curried que multiplica três números?

A) multiply x y z = x * y * z

B) multiply x = \y -> \z -> x * y * z*

C) multiply = \x y z -> x * y * z

D) multiply x y z = x * (y * z)

RESPOSTA: B

4) Como você pode utilizar currying para criar funções específicas a partir de funções genéricas?

A) Aplicando todos os argumentos de uma vez

B) Aplicando alguns argumentos e armazenando os restantes em uma nova função curried

C) Ignorando argumentos desnecessários

D) Criando variáveis globais

RESPOSTA: B

5) Por que o currying é frequentemente usado em bibliotecas de programação funcional?

A) Para aumentar a complexidade do código

B) Para permitir composição de funções de forma mais flexível*

C) Para melhorar a performance de execução

D) Para reduzir a quantidade de código

RESPOSTA: B

6) Em qual contexto o currying é menos útil?

A) Quando se utiliza programação imperativa*

B) Em sistemas altamente modulares

C) Em aplicações que requerem composição de funções

D) Em linguagens de programação funcional

RESPOSTA: A

7) Qual das seguintes expressões utiliza currying para filtrar uma lista de números pares?

const isEven = (num) => num % 2 === 0;

const filter = (fn) => (arr) => arr.filter(fn);

A) filter(isEven)([1, 2, 3, 4])

B) filter([1, 2, 3, 4])(isEven)

C) filter(isEven([1, 2, 3, 4]))

D) filter(isEven, [1, 2, 3, 4])

RESPOSTA: A

8) Qual é a saída da seguinte função?

```
const compose = (f, g) => (x) => f(g(x));

const double = (x) => x * 2;

const square = (x) => x * x;

const doubleThenSquare = compose(square, double);

console.log(doubleThenSquare(3));
```

A) 6

B) 9

C) 36

D) 12

RESPOSTA: C

9) Como você poderia aplicar currying para criar uma função de incremento?

```
const increment = (x) => x + 1;
```

A) const curriedIncrement = (x) => () => increment(x)

B) const curriedIncrement = increment.bind(null, 1)

C) const curriedIncrement = () => increment(1)

D) const curriedIncrement = (x) => increment(x + 1)

RESPOSTA: A

10) Qual é o resultado da seguinte expressão?

```
const log = (x) => {
  console.log(x);
  return x;
};

const curriedLog = (x) => (y) => log(x + y);

curriedLog(5)(10);
```

A) 15

B) 5

C) 10

D) undefined

RESPOSTA: A