

A wide-angle photograph of a large audience seated in rows of blue theater-style seats, facing a stage where a speaker is presenting. The stage has a dark blue wall with vertical panels and horizontal acoustic panels. A projector screen is visible on the right. A large white sign on the left side of the stage reads "THE RESEARCH METHOD".

JL.
/pgats

TERÇAS DE PROGRAMAÇÃO



AGENDA

- 1** Problema
- 2** Escrita de testes de unidade
- 3** Resolução dos problemas
- 4** Estudo dos conceitos

PROBLEMA #1

Neste kata você criará uma função que pega uma lista de inteiros e strings e retorna uma nova lista sem as strings.

Fonte: codewars.com

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

TESTES

Entrada	Resultado Esperado
[1, 2, 8, 'papi', 9, 12]	[2, 8, 9, 12]
['maria', 'helena', 'ata', '5']	[]
[3, 21, 23, 28]	[3, 21, 23, 28]
[]	[]

AGENDA

- 1 Problema**
- 2 Escrita de testes de unidade**
- 3 Resolução dos problemas**
- 4 Estudo dos conceitos**

CÓDIGO

```
function retornaListaSemString(lista) {  
  const listaDeNumeros = []  
  
  for (let item = 0; item < lista.length; item++) {  
    if (typeof lista[item] == 'number') {  
      listaDeNumeros.push(lista[item])  
    }  
  }  
  
  return listaDeNumeros  
}  
  
module.exports = {  
  retornaListaSemString  
}
```

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

CONCEITOS

1) **for**

Use toda vez que precisar olhar cada item contido na lista.

2) **if**

Use toda vez que precisar verificar (ou checar) algo que está em uma variável ou posição de lista

3) **[]**

Quando for necessário representar uma lista vazia utilize dois colchetes

4) **lista.length**

Use quando quiser saber quantos itens tem na lista

5) **typeof variável**

Use quando quiser saber o tipo do valor que está na variável

6) **lista.push(item)**

Use quando quiser adicionar um novo item na lista

AGENDA

- 1** Problema
- 2** Escrita de testes de unidade
- 3** Resolução dos problemas
- 4** Estudo dos conceitos

PROBLEMA #2

Neste kata, você deve elevar ao quadrado cada dígito de um número e concatená-los. Por exemplo, se executarmos 9119 na função, sairá 811181, porque 9 ao quadrado é 81, 1 ao quadrado é 1, 1 ao quadrado é 1 e 9 ao quadrado é 81. (81-1-1-81)

Fonte: [codewars.com](https://www.codewars.com/kata/51f2d1a0c6989b8a89000000)

AGENDA

- 1 Problema
- 2 Escrita de testes de unidade
- 3 Resolução dos problemas
- 4 Estudo dos conceitos

TESTES

Entrada	Resultado Esperado
395	9425
0	0
-1	0

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

CÓDIGO

```
// Código
function elevarDigitosAoQuadrado(digitosParaElevarAoQuadrado) {
  if (digitosParaElevarAoQuadrado < 0) {
    return 0
  }

  const stringDigitosParaElevarAoQuadrado =
    digitosParaElevarAoQuadrado.toString();
  const quantidadeDeDigitos = stringDigitosParaElevarAoQuadrado.length
  const numerosElevados = []

  for (let iterador = 0; iterador < quantidadeDeDigitos; iterador++) {
    const digitoAtual = stringDigitosParaElevarAoQuadrado[iterador]
    const elevadoAoQuadrado = parseInt(digitoAtual) * parseInt(digitoAtual);
    numerosElevados.push(elevadoAoQuadrado)
  }

  return parseInt(numerosElevados.join(''))
}
```

AGENDA

- 1 Problema
- 2 Escrita de testes de unidade
- 3 Resolução dos problemas
- 4 Estudo dos conceitos

CONCEITOS

- 1) **toString()**
Converter para String
- 2) **parseInt(valorEmTexto)**
Converter para número
- 3) **vetor.push(valor)**
Adicionar novos valores no vetor
- 4) **vetor.join(separador)**
Apresentar os itens do vetor separados por um caractere.

AGENDA

- 1** **Problema**
- 2** **Escrita de testes de unidade**
- 3** **Resolução dos problemas**
- 4** **Estudo dos conceitos**

PROBLEMA #3

Retorne o número de vogais no texto fornecido.

Consideraremos a, e, i, o e u como vogais para este Kata. O texto de entrada conterá apenas letras minúsculas e/ou espaços. Letras acentuadas não fazem parte desse desafio.

Fonte: codewars.com

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

TESTES

Entrada	Resultado Esperado
tercas	2
uva bonita e elegante	10
qhftwkImpfrz	0
qhftwk Impfrz	0
aeiou	5
bcd fghjklmnpqrstvxyz	0

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

CÓDIGO

```
const contadorVogais = palavra => {
  const letrasDaPalavra = palavra.split('')
  const oQueSaoVogais = ['a', 'e', 'i', 'o', 'u']
  let contador = 0

  for (let letra of letrasDaPalavra) {
    if (oQueSaoVogais.includes(letra)) {
      contador++
    }
  }

  return contador
}
```

AGENDA

- 1 Problema**
- 2 Escrita de testes de unidade**
- 3 Resolução dos problemas**
- 4 Estudo dos conceitos**

CONCEITOS

1) **For of**

Laço que interage com um vetor, rodando N vezes, onde N é a quantidade de itens contidos no vetor. A cada iteração, o item atual é atribuído a variável que está antes do "of"

2) **.split(separador)**

Converte uma string em um vetor, separando os caracteres a partir de um separador passado por parâmetro.

3) **.includes(item)**

Verifica se um item está incluso dentro de um vetor.

AGENDA

- 1 Problema**
- 2 Escrita de testes de unidade**
- 3 Resolução dos problemas**
- 4 Estudo dos conceitos**

PROBLEMA #4

Um Número Narcisista (ou Número de Armstrong) é um número positivo, a soma de seus próprios dígitos, cada um elevado à potência do número de dígitos em uma determinada base. Por exemplo, o número 153 possui 3 dígitos e é narcisista pois 1 ao cubo é 1, 5 ao cubo é 125 e 3 ao cubo é 27, logo, $153 = 1 + 125 + 27$.

Fonte: codewars.com

AGENDA

- 1 Problema
- 2 Escrita de testes de unidade
- 3 Resolução dos problemas
- 4 Estudo dos conceitos

TESTES

Entrada	Resultado Esperado
111	não narcisista
407	narcisista
0	narcisista
-567	não narcisista

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

CÓDIGO

```
function verificaNarcisista(numero) {  
    if (numero < 0)  
        return 'não narcisista'  
  
    if (numero.toString().length == 1)  
        return 'narcisista'  
  
    let acumulador = 0  
    for (let itemDoVetor of  
        numero.toString().split('')) {  
        acumulador += Math.pow(itemDoVetor,  
            numero.toString().length)  
    }  
  
    if (acumulador == numero)  
        return 'narcisista'  
  
    return 'não narcisista'  
}
```

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

CONCEITOS

1) Math.pow(numero, potencia)

Calcula e retorna a potência de um número

2) Uso de acumuladores

```
let acumulador = 0  
for (let numero of vetorDeNumeros) {  
    acumulador += numero  
}  
console.log(acumulador)
```

3) vetor.split(separador)

Converte uma string em um vetor, separando os caracteres a partir de um separador passado por parâmetro.

4) vetor.reduce((acumulador, itemVetor, indiceAtual, vetor) => formula, 0)

Reduz o vetor a um único valor calculado segundo a formula. O acumulador, no exemplo acima, começa com 0. O indiceAtual representa o item e o vetor é uma réplica do vetor.

AGENDA

1 **Problema**

2 **Escrita de testes de unidade**

3 **Resolução dos problemas**

4 **Estudo dos conceitos**

PROBLEMA #5

Eu tenho um gato e um cachorro. Comprei-os ao mesmo tempo. Isso foi há anos atrás. Retorne suas respectivas idades agora.

Regras importantes:

- O primeiro ano humano equivale a 15 anos de gato ou cachorro
- O segundo ano humano equivale a 9 anos de gato ou cachorro
- Cada novo ano humano vale 4 anos de gato e 5 anos de cachorro

Fonte: codewars.com

AGENDA

- 1 **Problema**
- 2 **Escrita de testes de unidade**
- 3 **Resolução dos problemas**
- 4 **Estudo dos conceitos**

TESTES

Entrada	Resultado Esperado
Valor de anos igual a zero: Anos = 0	{ }
Passar pela primeira regra: Anos = 1	{ cachorro: 15, gato: 15 }
Passar pela segunda regra: Anos = 2	{ cachorro: 24, gato: 24 }
Passar pelas duas regras: Anos = 10	{ cachorro: 64, gato: 56 }

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

CÓDIGO

```
function calculaIdadePets(quantidadeDeAnos) {  
    if (quantidadeDeAnos <= 0) {  
        return {}  
    }  
  
    switch (quantidadeDeAnos) {  
        case 1: return {  
            cachorro: 15,  
            gato: 15  
        }  
        case 2: return {  
            cachorro: 24,  
            gato: 24  
        }  
        default: return {  
            cachorro: 24 + ((quantidadeDeAnos - 2) * 5),  
            gato: 24 + ((quantidadeDeAnos - 2) * 4)  
        }  
    }  
}
```

AGENDA

- 1 **Problema**
- 2 **Escrita de testes de unidade**
- 3 **Resolução dos problemas**
- 4 **Estudo dos conceitos**

CONCEITOS

1) Objetos em Javascript

```
const objeto = {  
    propriedade1: "valor",  
    propriedade2: "valor"  
}
```

Propriedades podem ser acessadas através da sintaxe `objeto.propriedade1`

Um objeto vazio pode ser declarado ao usar o abrir e fechar de uma chave: {}

2) Switch/Case

```
switch (valor) {  
    case 'valor': // comando  
    case 1: // comando  
    default: // Se não foi nenhum dos outros  
}
```

AGENDA

- 1 Problema**
- 2 Escrita de testes de unidade**
- 3 Resolução dos problemas**
- 4 Estudo dos conceitos**

PROBLEMA #6

Dado uma lista com números definidos como inteiros e outros que foram definidos como texto, retorne a soma deles como se todos fossem inteiros.

Fonte: codewars.com

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

TESTES

Entrada	Resultado Esperado
Somar variação entre inteiros e strings numeros = [1,'2',3,'4']	10
Somar números que são strings numeros = ['2','4']	6
Somar apenas números numeros = [3,5,99]	107
Não calcular valores não numéricos numeros = ['a','y','*',1,'5']	6
Ignorar strings vazias numeros = ['',1,'5']	6

AGENDA

- 1 Problema**
- 2 Escrita de testes de unidade**
- 3 Resolução dos problemas**
- 4 Estudo dos conceitos**

CÓDIGO

```
function somarNumeros(numeros) {  
  let resultadoDaSoma = 0  
  
  for (let numero of numeros) {  
    if (!isNaN(numero)) {  
      resultadoDaSoma += Number(numero)  
    }  
  }  
  
  return resultadoDaSoma  
}
```

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

CONCEITOS

1) isNaN(valor)

Retorna true se o valor não for um número (tanto para string quanto para number)

2) if (funcao() === false)

Sempre que fizer uma comparação em relação a false, prefira usar o not (!) ficando assim:

```
if (!funcao())
```

3) Number(valor)

Se quiser converter um valor para numérico, utilize a classe Number passando o valor no construtor.

AGENDA

- 1 Problema**
- 2 Escrita de testes de unidade**
- 3 Resolução dos problemas**
- 4 Estudo dos conceitos**

PROBLEMA #7

Funcionários tem nome, salário e percentual de bônus combinados na contratação.

Quando o faturamento é maior que 100 mil reais, eles recebem bônus com base no percentual combinado em relação ao seu salário.

Escreva uma função que receba uma lista de funcionários e o faturamento da empresa e retorne o valor total de despesa com salários que a empresa terá naquele mês.

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

TESTES

Entrada	Resultado Esperado
Faturamento: 100.000,01 Pessoas: [{nome:'J',salario:1000;bonus:10}, {nome:'H',salario:500;bonus:20}]	Total a Pagar: 1700
Faturamento: 100.000,01 Pessoas: [{nome:'J',salario:1000;bonus:10}]	Total a Pagar: 1100
Faturamento: 100.000,01 Pessoas: []	Total a Pagar: 0

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

TESTES

Entrada	Resultado Esperado
Faturamento: 99.999,99 Pessoas: [{nome:'J',salario:1000;bonus:10}, {nome:'H',salario:500;bonus:20}]	Total a Pagar: 1500
Faturamento: 99.999,99 Pessoas: [{nome:'J',salario:1000;bonus:10}]	Total a Pagar: 1000
Faturamento: 99.999,99 Pessoas: []	Total a Pagar: 0

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

TESTES

Entrada	Resultado Esperado
Faturamento: 100.000,00 Pessoas: [{nome:'J',salario:1000;bonus:10}, {nome:'H',salario:500;bonus:20}]	Total a Pagar: 1500
Faturamento: 100.000,01 Pessoas: [{nome:'J',salario:1000;bonus:0}]	Total a Pagar: 1000

AGENDA

- 1 Problema**
- 2 Escrita de testes de unidade**
- 3 Resolução dos problemas**
- 4 Estudo dos conceitos**

CÓDIGO

```
function calcularDespesasSalario(pessoas,  
faturamentoMensal) {  
    let valorAPagar = 0  
  
    for (pessoa of pessoas) {  
        valorAPagar += pessoa.salario  
  
        if (faturamentoMensal > 100000.00) {  
            valorAPagar += pessoa.salario *  
                pessoa.bonus / 100  
        }  
    }  
  
    return valorAPagar  
}
```

AGENDA

1 Problema

2 Escrita de testes de unidade

3 Resolução dos problemas

4 Estudo dos conceitos

CONCEITOS

1) For of

Laço que interage com um vetor, rodando N vezes, onde N é a quantidade de itens contidos no vetor. A cada iteração, o item atual é atribuído a variável que está antes do "of"

2) Array de Objetos

Uma lista com diversos objetos:

```
const pessoas = [  
  { nome:'J', salario: 1000, bonus: 10 },  
  { nome:'H', salario: 500, bonus: 20 }  
]
```

3) Acumuladores

Ao acumular, não se esqueça de criar uma variável fora do laço, iniciada com o número 0 e dentro do laço, colocar a variável seguida de +=

4) Calculo de percentual com dois inteiros

percentual = total * valor / 100



DÚVIDAS?