

nível —

MEDIUM



Matemática

Manipulação de Tipos

15. Código Romano I

Em uma missão por regiões não mapeadas pela Federação sua equipe encontrou um povoado que utilizava um sistema numérico muito parecido com o romano.

Você foi encarregado de implementar um tradutor capaz de converter qualquer número (em formato de texto) para o seu inteiro decimal correspondente.

Desafio 15: Código Romano I

Escreva uma função que recebe uma string de algarismos romanos e seja capaz de traduzir seu conteúdo retornando o inteiro decimal correspondente.

Os algarismos romanos são:

- I: 1
- V: 5
- X: 10
- L: 50
- C: 100
- D: 500
- M: 1000

Os outros números são formados a partir de dois tipos de notação, a padrão e a subtrativa.

- Notação Padrão: I, II e III (1, 2 e 3), VI, VII e VIII (6, 7 e 8), X, XX e XXX (10, 20 e 30), etc.
- Notação Subtrativa: IV ($1 - 5 = 4$), IX ($1 - 10 = 9$), XLIX ($10 - 50 + 1 - 10 = 49$), XC ($10 - 100 = 90$), CMXCIX ($100 - 1000 + 10 - 100 + 1 - 10 = 999$)

Testes 15: Código Romano I

- Entrada: ('XLVII')

Saída: 47

- Entrada: ('CDXXXVIII')

Saída: 438

- Entrada: ('CMIX')

Saída: 909

- Entrada: ('MMMCMXCIX')

Saída: 3999

Matemática

Manipulação de Tipos

Strings

16. Código Romano II

Enquanto em contato com o mesmo povoado do planeta não mapeado, você e sua equipe interceptaram uma tentativa de comunicação que estava sendo enviada para algum lugar.

O conteúdo parecia estar criptografado utilizando um método primitivo semelhante à cifra de Cesar.

Agora vocês precisam de um programa capaz de traduzir esse conteúdo a partir de uma chave.

Desafio 16: Código Romano II

Escreva uma função que recebe uma string e um número e retorne o resultado da aplicação da Cifra de Cesar para descriptografar o seu conteúdo, ou seja, que retroceda cada letra pelo número passado seguindo a ordem alfabética.

Testes 16: Código Romano II

- Entrada: ('Vguvg', 2)

Saída: 'Teste'

- Entrada: ('BCDYZAbcdyza', 27)

Saída: 'ABCXYZabcxz'

- Entrada: ('Qaiik', 60)

Saída: 'Isaac'

- Entrada: ('Amozmlw', 8)

Saída: 'Segredo'

Strings Arrays

17. Relatório de Missão

Ao concluir uma missão em nome da Federação o capitão do esquadrão deve redigir o seu relatório como parte do protocolo padrão.

Para auxiliar neste processo, um robô assistente revisa o texto e pode sugerir correções.

A fim de agilizar esse processo, foi requisitado que você ajudasse na construção de uma interface para que o robô insira as correções diretamente no arquivo do relatório.

Desafio 17: Relatório de Missão

Escreva uma função que recebe três argumentos, uma frase, uma palavra e um array de índices. A função deve retornar a frase com a palavra inserida em cada uma das posições especificadas pelo array de índices.

- A função não deve funcionar em índices que não estejam no alcance da frase.
- A função deve retornar a mesma frase caso o array de índices esteja vazio.

Testes 17: Relatório de Missão

- Entrada: ('capaz utilizar as cápsulas emergência', 'de ', [6, 27])
Saída: 'capaz de utilizar as cápsulas de emergência'
- Entrada: ('Nós decidimos apesar das chances serem baixas que enviaríamos um sinal para [...]', '', [13, 45])
Saída: 'Nós decidimos, apesar das chances serem baixas, que enviaríamos um sinal para [...]'
- Entrada: ('Hello', 'world', [6])
Saída: 'Hello'
- Entrada: ('Isso é um teste', 'não', [])
Saída: 'Isso é um teste'

Classes Matemática

18. Módulo de Treinamento

Na colônia estabelecida no planeta primitivo Romulus está sendo criada uma escola a fim de fornecer educação básica para os colonos.

Para auxiliar durante as aulas de matemática, foi solicitado que uma equipe construísse um programa de computador capaz de explicar e resolver diversos tipos de problemas e conceitos.

Como parte da equipe, você foi encarregado de construir o módulo que trata de equações até o segundo grau, portanto precisará construir um programa capaz de resolver equações e descrever os passos da resolução.

É obrigatório o uso de classes.

Desafio 18: Módulo de Treinamento:

Escreva uma classe Equation que seja capaz de lidar com uma equações até o segundo grau.

Os objetos podem ser instanciados com valores padrões para $a = 0$, $b = 0$ e $c = 0$.

Ela deve ter um método para retornar suas raízes na forma de um array vazio, de um ou de dois elementos.

O método também deve descrever os passos para a resolução da equação.

Se a , b e c forem todos iguais a 0 ela não deve calcular as raízes e deve retornar uma mensagem de erro.

Testes 18: Módulo de Treinamento

- Entrada: (0, 2, 6)

Saída:

1. $2x + 6 = 0$

2. $2x = -6$

3. $x = -6 / 2$

4. $x = -3$

[-3]

- Entrada: ()

Saída: 'Erro! Nenhum parâmetro informado.'

- Entrada: (0, 0, 5)

Saída:

1. Parâmetros insuficientes.

Nenhuma raiz real.

[]

- Entrada: (1, -4, -5)

Saída:

1. $\Delta = -4^2 - 4 * 1 * -5$

2. $\Delta = 36$

3. $x' = (-(-4) + \sqrt{36}) / 2 * 1$

4. $x'' = (-(-4) - \sqrt{36}) / 2 * 1$

5. $x' = 4 + \sqrt{36} / 2$

6. $x'' = 4 - \sqrt{36} / 2$

7. $x' = 4 + 6 / 2$

8. $x'' = 4 - 6 / 2$

9. $x' = 5$

10. $x'' = -1$

[5, -1]

Matemática

Recursão

19. Cálculos de Viagens Espaciais II

A equipe de desenvolvimento do sistema de navegação das naves da Federação precisa novamente da sua ajuda para implementar uma funcionalidade.

Ela consiste em calcular a persistência multiplicativa de um determinado número.

A persistência multiplicativa é a quantidade de vezes que se precisa substituir um número pelo produto de seus algarismos até chegar a um número de um único dígito.

Por exemplo: $539 = 3$, pois $5 \times 3 \times 9 = 135$, $1 \times 3 \times 5 = 15$ e $1 \times 5 = 5$, ou seja, 3 multiplicações.

Desafio 19: Cálculos de Viagens Espaciais II

Escreva uma função que recebe um número e retorna a sua persistência multiplicativa, que é a quantidade de vezes que é necessário multiplicar os seus algarismos para se chegar em um número de um único algarismo.

Testes 19: Cálculos de Viagens Espaciais II

- Entrada: (539)

Saída: 3

- Entrada: (999)

Saída: 4

- Entrada: (7)

Saída: 0

- Entrada: (7169)

Saída: 5

Recursão Arrays

20. Quebrando a Senha

Em uma missão no planeta Darnas a equipe de reconhecimento encontrou diversos dispositivos que parecem ter informações cruciais para a missão, porém os dispositivos estão bloqueados por senhas.

Você e seu colega estão criando um procedimento para descobrir as senhas utilizando força bruta.

Você ficou encarregado de criar a função que irá calcular todas as senhas possíveis a partir da lista de caracteres que a compõe.

Desafio 20: Quebrando a Senha

Escreva uma função que receba um array de opções e retorne um array bidimensional de todas as senhas possíveis utilizando todos os elementos passados.

Faça isso utilizando recursão.

Testes 20: Quebrando a Senha

- Entrada: (["X", "s", "-", "#"])

Saída: [

```
[ 'X', 's', '-', '#' ], [ 's', 'X', '-', '#' ],
[ 's', '-', 'X', '#' ], [ 's', '-', '#', 'X' ],
[ 'X', '-', 's', '#' ], [ '-' , 'X', 's', '#' ],
[ '-' , 's', 'X', '#' ], [ '-' , 's', '#', 'X' ],
[ 'X', '-', '#', 's' ], [ '-' , 'X', '#', 's' ],
[ '-' , '#', 'X', 's' ], [ '-' , '#', 's', 'X' ],
[ 'X', 's', '#', '-' ], [ 's', 'X', '#', '-' ],
[ 's', '#', 'X', '-' ], [ 's', '#', '-', 'X' ],
[ 'X', '#', 's', '-' ], [ '#' , 'X', 's', '-' ],
[ '#' , 's', 'X', '-' ], [ '#' , 's', '-', 'X' ],
[ 'X', '#', '-', 's' ], [ '#' , 'X', '-', 's' ],
[ '#' , '-', 'X', 's' ], [ '#' , '-' , 's', 'X' ]
```

]

- Entrada: (["1", "2", "3"])

Saída: [['1', '2', '3'], ['2', '1', '3'], ['2', '3', '1'], ['1', '3', '2'],
['3', '1', '2'], ['3', '2', '1']]

- Entrada: ([])

Saída: [[]]

Matemática Classes Estruturas Básicas

21. Sistema de Localização

A estação espacial Olympus, que está nas etapas finais de sua construção, possui um sistema de rastreamento que deve ser capaz de localizar coordenadas próximas a partir da sua própria posição.

Para isso, você foi incumbido da tarefa de construir parte desse sistema.

Você deve construir uma classe capaz de armazenar um ponto de 3 coordenadas numéricas e também detectar o setor em que elas se encontram.

Além disso, também deve ser capaz de calcular a distância entre o ponto e a estação espacial.

Desafio 21: Sistema de Localização

Escreva uma classe que seja capaz de armazenar 3 coordenadas, determinar o setor em que se encontram suas coordenadas e sua distância em relação à estação espacial (coordenada [0, 0, 0]).

Distribuição dos setores:

- Alfa: [positivo, positivo, positivo]
- Beta: [positivo, positivo, negativo]
- Gama: [positivo, negativo, positivo]
- Delta: [positivo, negativo, negativo]
- Épsilon: [negativo, positivo, positivo]
- Zeta: [negativo, positivo, negativo]
- Sigma: [negativo, negativo, positivo]
- Ômega: [negativo, negativo, negativo]

Considere 0 como positivo para garantir que um ponto estará apenas em um único setor.

Testes 21: Sistema de Localização

- Entrada: ([37, 42, 15])

Saída setor: Alfa

Saída distância: 57.94825277780168

- Entrada: ([144, 49, 0])

Saída setor: Alfa

Saída distância: 152.10851389715174

- Entrada: ([-37, 0, 0])

Saída setor: Épsilon

Saída distância: 37

- Entrada: ([-19, -80, -32])

Saída setor: Delta

Saída distância: 88.23264701911646

22. Prisão Intergalática

A estação espacial Tartarus é utilizada pela Federação como prisão para criminosos de toda a galáxia e foi construída utilizando a mais avançada tecnologia para evitar qualquer possibilidade de fuga.

Um dos recursos utilizados é o monitoramento constante de cada prisioneiro, o que deve ser feito com cuidado redobrado quando estes são liberados de suas celas.

Sempre que isso ocorre é feito um escaneamento de todos os prisioneiros, que são sempre identificados por números consecutivos e possuem um tempo limite para entrem ou saiam.

Ao fim desse tempo é gerada uma lista que deve conter os números de todos os prisioneiros, mesmo que em ordem aleatória.

Para garantir que todos entrem e saiam sem que nenhum fique para trás de forma automatizada foi requerido que você construa um código para realizar essa tarefa.

Para esse cenário o último número será um verificador que indicará o número total de prisioneiros, portanto nunca estará faltando.

Desafio 22: Prisão Intergalática

Escreva uma função que receba uma lista embaralhada de números únicos de 1 até n no formato "0001" (string com zeros à esquerda), verifique se há elementos faltando (onde n nunca estará faltando) e, caso hajam, retorne os elementos que faltam.

Testes 22: Prisão Intergalática

- Entrada: (['0020', '0002', '0013', '0004', '0001', '0016', '0015',
'0006', '0012', '0007', '0005', '0008', '0011', '0010'])

Saída: ['0003', '0009', '0014', '0017', '0018', '0019']

- Entrada: (['0020', '0009', '0002', '0013', '0004', '0017', '0001',
'0003', '0016', '0015', '0019', '0006', '0012', '0007', '0005', '0014',
'0008', '0011', '0010', '0018'])

Saída: []

- Entrada: (['0004', '0002', '0005', '0003'])

Saída: ['0001']

- Entrada: ([])

Saída: []



PARABÉNS!

Você acaba de concluir
8 desafios Medium e avançar
na sua jornada pela galáxia.

Sua nova atribuição:
Programador(a) JavaScript
das galáxias nível ouro.

onebitcode {🌐}

FOR AMAZING PROGRAMMERS

Tire um print, publique nos **stories do Instagram**
e marque @onebitcode