

80 Carlos

Aluno \_\_\_\_\_ Código: \_\_\_\_\_

Cada questão abaixo vale 2,0 pontos. Você pode resolver quantas quiser dentro do tempo de 80 minutos.

**Assunto da prova:** comandos de atribuição, entrada de dados (input), saída (print), condicional (if) e funções.

1. Pesquisando no Google, temos que "A velocidade média é uma grandeza caracterizada pela variação da posição de um corpo em determinado tempo. Com base na divisão do espaço percorrido pelo tempo que demorou para percorrer, é possível encontrar a velocidade do corpo:  $V_m = \Delta S / \Delta t$ ." Faça um programa que pergunta ao usuário quanto tempo demorou a viagem dele até os Lençóis Maranhenses, que é um percurso de 260 km. Uma vez que possua o tempo e a distância, informe ao usuário a velocidade média que foi empregada na viagem em km/h.
2. Você está programando o jogo de dados Yahtzee. A tarefa que foi designada para você é a de verificar se todos os cinco valores lançados em cinco dados comuns de 6 faces são iguais. Quando os cinco valores são iguais, chamamos isso de Yahtzee, para isso deve implementar a função **yahtzee** (veja cabeçalho abaixo), que recebe como parâmetros os valores inteiros de 1 a 6 que foram tirados em cada um dos cinco dados. A função yahtzee deve retornar True quando esses cinco valores forem iguais, ou False em caso contrário.  
`def yahtzee(d1:int, d2:int, d3:int, d4:int, d5:int) -> bool:`
3. Você está programando o jogo de dados Yahtzee e sua nova tarefa agora é identificar um fullhouse. A categoria fullhouse ocorre quando, no resultado dos cinco dados, temos uma trinca (três valores iguais) e mais um par (dois valores iguais que são diferentes da trinca). A função fullhouse, a seguir, deve retornar True quando tivermos uma trinca e um par (conforme definido) ou False, em caso contrário. *Exemplos:* {1, 2, 1, 2, 1} é um fullhouse; {4, 4, 4, 4, 4} não é um fullhouse.  
`def fullhouse(d1:int, d2:int, d3:int, d4:int, d5:int) -> bool:`
4. Você está programando o jogos de dados Yahtzee e recebeu a tarefa de somar a ocorrência de um determinado valor entre os resultados de cinco dados. Para isso, deve implementar a função **sumCount** (veja cabeçalho abaixo), que retorna a soma de todos os resultados **d1, d2, d3, d4, d5** que são exatamente iguais ao valor **value** passado como parâmetro. *Exemplos:* `sumCount(3, 5, 2, 3, 1, 3)` retorna 6 pois temos dois 3 entre os resultados {5, 2, 3, 1, 3}; por outro lado, `sumCount(1, 6, 5, 4, 3, 2)` retorna 0, pois não temos o 1 entre os resultados {6, 5, 4, 3, 2}.  
`def sumCount(value:int, d1:int, d2:int, d3:int, d4:int, d5:int) -> int:`
5. Crie um programa de computador que pede para o usuário informar quatro inteiros em qualquer ordem. O programa deve verificar se a soma do menor e maior valor entre esses inteiros é maior, igual ou maior que os dois outros inteiros informados. Perceba que podemos ter valores iguais entre os inteiros informados pelo usuário. *Exemplos:* Para os valores {19, 1, 8, 12}, o programa deve imprimir que é igual, pois a soma do menor 1 com o maior 19 é igual à soma dos outros dois (8+12); por outro lado, para os valores {3, 3, 78, 109}, o programa deve imprimir que é maior, pois a soma 3+109 é maior que 3+78.
6. Só três pessoas compareceram na festa. O nome delas (**name1, name2, name3**) está sendo informado para a função **party** a seguir. Um programador deveria ter criado uma função que verifica se "Maria" e "João", os dois, foram para a festa juntos. Acontece que a função está com dois bugs. Dê dois exemplos de strings (**name1, name2, name3**) que provocam cada um dos dois referidos bugs. Depois disso, reescreva o código, corrigindo-o.

```
def party(name1: str, name2: str, name3: str) -> bool:
    if name1=="Maria":
        return name2=="João" or name3=="João":
    elif name2=="Maria":
        return name1 or name3=="João"
    else:
        return name1=="João" or name2=="João"
```

Força, Padawan! Sorte é para os despreparados.

tempo = float(input('em horas, qual a tempo levou o carro viagem?'))

distancia = 260 # km

(20)

velocidade\_media = distancia / tempo

print('a velocidade média na sua viagem foi de: {velocidade\_media} # km/h')

5) numero1 = int(input('diga quatro números inteiros, um por vez:')) # faltou o else da primeira if, mas iria repetir a estrutura até ~~for~~ eliminar os bugs.

(10) numero2 = int(input())

numero3 = int(input())

numero4 = int(input())

if numero1 > numero2 and numero1 > numero3 and numero1 > numero4:

maior\_numero = numero1

if numero2 > numero3 and numero4 > numero3:

maior\_numero = numero3

Soma = maior\_numero + menor\_numero → falta variável

Soma2 = numero2 + numero4

if soma > soma2:

print("soma > soma2")

elif soma == soma2:

print("soma == soma2")

elif soma < soma2:

print("soma < soma2")

→ Problems se há dois ou mais maiores iguais.

⇒ Poss fazer assim, faltam mais ifs

2) def Yatzee(d1: int, d2: int, d3: int, d4: int, d5: int) → bool

(20) if d1 == d2 and d2 == d3 and d3 == d4 and d4 == d5:

return True

else:

return False

3) def Yatzee

def fullYatzee(d1: int, d2: int, d3: int, d4: int, d5: int) → bool:

(10) if d1 == d2 and d2 == d3:

if d4 == d5 and d1 != d4:

return True

else:

return False

if d1 == d2 and d2 == d4:

if d3 == d5 and d1 != d3:

return True

else:

return False

if d1 == d2 and d2 == d5:

if d3 == d4 and d1 != d3:

return True

else:

return False

if d1 == d3 and d3 == d4:

if d2 == d5 and d1 != d2:

return True

else:

# três números coincidem, o quarto difere dos outros números, onde que por sua vez deve coincidir com o quinto, onde essa estrutura na função inteira.

Poss fazer desse jeito, precisa ter todos os casos, são 5x4 ifs.

(10)



3) (cont...)

```
if d1 == d3 and d3 == d5:  
    if d2 == d4 and d1 != d2:  
        return True  
    else:  
        return False
```

```
if d1 == d4 and d4 == d5:  
    if d2 == d3 and d1 != d2:  
        return True  
    else:  
        return False
```

6) def party(name1: str, name2: str, name3: str) -> bool:

(20)

```
if name1 == "maria":  
    return name2 == "joão" or name3 == "joão"  
elif name2 == "maria":  
    return name1 == "joão" or name3 == "joão"  
elif name3 == "maria":  
    return name1 == "joão" or name2 == "joão"
```

# a filiação anterior devia uma chamada a str false (joão, maria, maria) ou chamada maria false name2  
joão não seria dada como name1.

4) def rum\_calandr (nome1: int, d1: int, d2: int, d3: int, d4: int, d5: int) -> int: # não dá tempo.

X