

QUESTÃO 1 de 4 - Conteúdo até Aula 03

Enunciado: Imagina-se que você é um dos programadores responsáveis pela construção de app para uma empresa X que vende Planos de Saúde. Uma das estratégias dessa empresa X é cobrar um valor diferente com base na idade do cliente, conforme a **listagem abaixo**:

- Se a idade for **maior ou igual** que **0** e **menor** que **19**, o valor será de **100%** do **valor base** do plano (100 / 100);
- Se a idade for **maior ou igual** que **19** e **menor** que **29**, o valor será de **150%** do **valor base** do plano (150 / 100);
- Se a idade for **maior ou igual** que **29** e **menor** que **39**, o valor será de **225%** do **valor base** do plano (225 / 100);
- Se a idade for **maior ou igual** que **39** e **menor** que **49**, o valor será de **240%** do **valor base** do plano (240 / 100);
- Se a idade for **maior ou igual** que **49** e **menor** que **59**, o valor será de **350%** do **valor base** do plano (350 / 100);
- Se a idade for **maior ou igual** que **59**, o valor será de **600%** do **valor base** do plano (600 / 100);

O valor mensal do plano é calculado da seguinte maneira:

$$\text{valorMensal} = \text{valorBase} * \text{porcentagem}$$

Exemplo: Se o **valorBase** informado for **100.00** e a **idade** for **45** anos (**240%** segundo a tabela acima)

$$\text{valorMensal} = 100.00 * \left(\frac{240}{100}\right) = R\$ 240.00$$

Elabore um programa em Python que:

- A. Deve-se implementar o **print** com o seu **nome completo** (somente print, não usar input aqui).
Por exemplo: **print(“Sistema desenvolvido por Bruno KostiuK”) [EXIGÊNCIA DE CÓDIGO 1 de 6];**
- B. Deve-se implementar o input do **valorBase** do plano e da **idade** do cliente **[EXIGÊNCIA DE CÓDIGO 2 de 6];**
- C. Deve-se implementar as regras de valores **conforme a enunciado acima** (obs.: atente-se as condições de menor, igual e maior) **[EXIGÊNCIA DE CÓDIGO 3 de 6];**
- D. Deve-se implementar o **valorMensal** **[EXIGÊNCIA DE CÓDIGO 4 de 6];**
- E. Deve-se implementar as estruturas **if, elif e else (todas elas)** **[EXIGÊNCIA DE CÓDIGO 5 de 6];**
- F. Deve-se inserir comentários **relevantes** no código **[EXIGÊNCIA DE CÓDIGO 6 de 6];**
- G. Deve-se apresentar na saída de console uma mensagem com seu nome completo **[EXIGÊNCIA DE SAÍDA DE CONSOLE 1 de 2];**
- H. Deve-se apresentar na saída de console a utilização do sistema informando uma **idade maior ou igual a 29 anos**, apresentando na saída de console o **valorMensal** do plano **[EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 2];**

EXEMPLO DE SAÍDA DE CONSOLE:

```
➡ Bem vindo ao Sistema do Bruno KostiuK
Informe o valor Base do plano: R$ 134.05
Informe a idade do cliente: 34
O valor mensal do plano é de: R$ 301.61
```

Figura 1.1: Exemplo de saída de console que o aluno deve fazer. Em que se perguntar o valorBase do plano (pode ser qualquer valor) e a idade (maior ou igual a 29 anos **[EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 2]**), e é apresentado o valorMensal.

Apresentação de *Código da Questão 1:*

```
"""
Calcula o valor mensal do plano de acordo com a idade do cliente.

Parâmetros:
- idade (int): Idade do cliente
- valorBase (float): Valor base do plano

Retorno:
- float: Valor mensal atualizado com base na idade do cliente
"""

def valorMensal(idade, valorBase):
    if idade >= 0 and idade < 19:
        return valorBase
    elif idade >= 19 and idade < 29:
        return (valorBase * 150) / 100
    elif idade >= 29 and idade < 39:
        return (valorBase * 225) / 100
    elif idade >= 39 and idade < 49:
        return (valorBase * 240) / 100
    elif idade >= 49 and idade < 59:
        return (valorBase * 350) / 100
    else:
        return (valorBase * 600) / 100

# Exibe mensagem de boas-vindas ao sistema
print('Bem vindo ao sistema do WANDERSON TEIXEIRA SOUSA')

while True: # Loop para solicitar as entradas até que sejam válidas
    try:
        valorBase = float(input('Informe o valor base do plano: ')) # Solicita ao usuário o valor base do plano e converte para float
        while valorBase < 0: # Verifica se o valor base é negativo, e solicita novamente caso seja
            print('Valor base inválido. O valor deve ser positivo.')
            valorBase = float(input('Informe o valor base do plano: '))

        idade = int(input('Informe a idade do cliente: ')) # Solicita ao usuário a idade do cliente e converte para inteiro
        while idade < 0: # Verifica se a idade é negativa, e solicita novamente caso seja
            print('Idade inválida, digite novamente.')
            idade = int(input('Informe a idade do cliente: '))

        # Calcula e exibe o valor mensal do plano com base na idade e valor base
        print(f'O valor mensal do plano é de: R$ {valorMensal(idade, valorBase):.2f}')
        break # Encerra o loop quando as entradas são válidas

    # Captura exceções de entrada inválida e solicita ao usuário que insira valores numéricos
    except ValueError:
        print('Entrada inválida. Por favor, insira um número válido.')
```

Apresentação de ***Saída do Console da Questão 1:***

```
PS C:\Users\Wanderson\Documents\GitHub\UNINTER\Lógica de Programação e Algoritmos> & "C:/Program Files/Python312/python.exe" "c:/Users/Wanderson/Documents/GitHub/UNINTER/Lógica de Programação e Algoritmos/trabalho.py"
Bem vindo ao sistema do WANDERSON TEIXEIRA SOUSA
Informe o valor base do plano: 134.05
Informe a idade do cliente: 34
O valor mensal do plano é de: R$ 301.61
PS C:\Users\Wanderson\Documents\GitHub\UNINTER\Lógica de Programação e Algoritmos>
```

QUESTÃO 2 de 4 - Conteúdo até aula 04

Enunciado: Você e sua equipe de programadores foram contratados para desenvolver um app de vendas para uma Pizzaria que vende sabores de Pizzas Doces e Pizzas Salgadas. Você ficou com a parte de desenvolver a interface do cliente para retirada do produto.

A Loja possui seguinte relação:

- Tamanho **P**: Pizza Salgada (**PS**) custa 30 reais e a Pizza Doce (**PD**) custa 34 reais;
- Tamanho **M**: Pizza Salgada (**PS**) custa 45 reais e a Pizza Doce (**PD**) custa 48 reais;
- Tamanho **G**: Pizza Salgada (**PS**) custa 60 reais e a Pizza Doce (**PD**) custa 66 reais;

Elabore um programa em Python que:

- A. Deve-se implementar o **print** com o seu **nome completo** (somente print, não usar input aqui).
Por exemplo: **print(“Bem-vindos a Pizzaria do Bruno Kostiuk”)**
Além do seu nome completo, deve-se implementar um **print com um Menu** para o cliente. [EXIGÊNCIA DE CÓDIGO 1 de 8];
- B. Deve-se implementar o input do **sabor** (PS/PD) e o print “Sabor inválido. Tente novamente” se o usuário entra com valor diferente de PS e PD [EXIGÊNCIA DE CÓDIGO 2 de 8];
- C. Deve-se implementar o input do **tamanho** (P/M/G) e o print “Tamanho inválido. Tente novamente” se o usuário com entra valor diferente de P, M ou G [EXIGÊNCIA DE CÓDIGO 3 de 8];
- D. Deve-se implementar **if, elif e/ou else**, utilizando o modelo **aninhado** (aula 3 – Tema 4) com cada uma das combinações de **sabor** e **tamanho** [EXIGÊNCIA DE CÓDIGO 4 de 8];
- E. Deve-se implementar um **acumulador** para somar os valores dos pedidos (valor total do pedido) [EXIGÊNCIA DE CÓDIGO 5 de 8];
- F. Deve-se implementar o input com a pergunta: “Deseja pedir mais alguma coisa?”. Se sim **repetir a partir do item B**, senão encerrar o programa executar o print do **acumulador** [EXIGÊNCIA DE CÓDIGO 6 de 8];
- G. Deve-se implementar as estruturas de **while, break, continue (todas elas)** [EXIGÊNCIA DE CÓDIGO 7 de 8];
- H. Deve-se inserir comentários **relevantes** no código [EXIGÊNCIA DE CÓDIGO 8 de 8];
- I. Deve-se apresentar na saída de console uma mensagem com o seu nome completo e o menu para o cliente conhecer as opções [EXIGÊNCIA DE SAÍDA DE CONSOLE 1 de 4];
- J. Deve-se apresentar na saída de console um pedido em que o usuário errou o **sabor** [EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 4];
- K. Deve-se apresentar na saída de console um pedido em que o usuário errou o **tamanho** [EXIGÊNCIA DE SAÍDA DE CONSOLE 3 de 4];
- L. Deve-se apresentar na saída de console um pedido com duas opções sabores diferentes e com tamanhos diferentes [EXIGÊNCIA DE SAÍDA DE CONSOLE 4 de 4];

EXEMPLO DE SAÍDA DE CONSOLE:

```
➡ ----- Bem-vindo a Pizzaria do Bruno Kostiuk -----
-----Cardápio-----
-----
---| Tamanho |      Pizza Salgada(PS) |      Pizza Doce(PD) |---
---|   P   |      R$ 30.00         |      R$ 34.00         |---
---|   M   |      R$ 45.00         |      R$ 48.00         |---
---|   G   |      R$ 60.00         |      R$ 66.00         |---
-----

Entre com o sabor desejado (PS/PD): PG
Sabor inválido. Tente novamente

Entre com o sabor desejado (PS/PD): PS
Entre com o tamanho desejado (P/M/G): EXGG
Tamanho inválido. Tente novamente

Entre com o sabor desejado (PS/PD): PS
Entre com o tamanho desejado (P/M/G): G
Você pediu uma Pizza Salgada no tamanho G: R$ 60.00

Deseja mais alguma coisa? (S/N): S
Entre com o sabor desejado (PS/PD): PD
Entre com o tamanho desejado (P/M/G): M
Você pediu uma Pizza Doce no tamanho M: R$ 48.00

Deseja mais alguma coisa? (S/N): N

O valor total a ser pago: R$ 108.00
```

Figura 2.1: Exemplo de saída de console que o aluno deve fazer. Em que se perguntar o sabor e o tamanho. Há uma tentativa de pedido que se errou o sabor e outra que se errou o tamanho. Há também um pedido com dois itens com sabores e tamanhos diferentes.

Apresentação de **Código da Questão 2:**

```
"""
Calcula o valor total do pedido de pizzas de acordo com o sabor e o tamanho escolhidos pelo cliente.
Funções:
- menu(): Exibe o cardápio com preços de pizzas salgadas e doces.
- pedir_pizza(tamanho, sabor, acumulador): Calcula o valor da pizza baseada no tamanho e sabor escolhidos pelo cliente e acumula o valor total do pedido.
Parâmetros da função pedir_pizza:
- tamanho (str): Tamanho da pizza ('P', 'M', ou 'G').
- sabor (str): Sabor da pizza ('PS' para pizza salgada, 'PD' para pizza doce).
- acumulador (float, opcional): Acumulador que armazena o valor total do pedido. Valor inicial é 0.
Retorno:
- float: Valor total atualizado após o pedido.
O programa permite que o cliente faça múltiplos pedidos, valida as entradas e calcula o valor final.
"""

# Função para exibir o menu da pizzeria e informações iniciais
def menu():
    print(10 * '-' + ' Bem-vindo a Pizzaria do Wanderson Teixeira ' + 10 * '-')
    print(64 * '-')
    print(27 * '-' + ' Cardápio ' + 27 * '-')
    print(64 * '-')
    print(3 * '-' + '| Tamanho | Pizza Salgada (PS) | Pizza Doce (PD) |' + 3 * '-')
    print(3 * '-' + '| P | R$ 30.00 | R$ 34.00 |' + 3 * '-')
    print(3 * '-' + '| M | R$ 45.00 | R$ 48.00 |' + 3 * '-')
    print(3 * '-' + '| G | R$ 60.00 | R$ 66.00 |' + 3 * '-')
    print(64 * '-')

# Função para processar o pedido de pizza
def pedir_pizza(tamanho, sabor, acumulador=0):
    # Verifica o sabor da pizza e atribui o preço de acordo com o tamanho
    if sabor == 'PS':
        if tamanho == 'P':
            preco = 30
        elif tamanho == 'M':
            preco = 45
        elif tamanho == 'G':
            preco = 60
        print(f'\nVocê pediu uma Pizza Salgada no tamanho {tamanho}: R$ {preco:.2f} reais')
    elif sabor == 'PD':
        if tamanho == 'P':
            preco = 34
        elif tamanho == 'M':
            preco = 48
        elif tamanho == 'G':
            preco = 66
        print(f'\nVocê pediu uma Pizza Doce no tamanho {tamanho}: R$ {preco:.2f} reais')

    # Soma o preço ao acumulador de total
    acumulador += preco
    return acumulador

# Exibe o menu
menu()
acumulador = 0 # Inicializa o acumulador de preço total

while True:
    try:
        # Solicita o sabor da pizza ao cliente
        sabor = input('\nEntre com o sabor desejado (PS/PD): ').upper()
        # Verifica se o sabor informado é inválido
        while sabor not in ['PS', 'PD']:
            print('Sabor inválido. Tente novamente.\n')
            sabor = input('Entre com o sabor desejado (PS/PD): ').upper()

        # Solicita o tamanho da pizza ao cliente
        tamanho = input('Entre com o tamanho desejado (P/M/G): ').upper()
        # Verifica se o tamanho informado é inválido
        while tamanho not in ['P', 'M', 'G']:
            print('Tamanho inválido. Tente novamente.\n')
            tamanho = input('Entre com o tamanho desejado (P/M/G): ').upper()

        # Atualiza o acumulador com o valor retornado pela função pedir_pizza
        acumulador = pedir_pizza(tamanho, sabor, acumulador)
        # Pergunta se o cliente deseja mais alguma coisa
        mais_pizza = input('\nDeseja mais alguma coisa? (S/N): ').upper()

        # Verifica se a resposta é inválida
        while mais_pizza not in ['S', 'N']:
            mais_pizza = input('\nDeseja mais alguma coisa? (S/N): ').upper()
        # Finaliza o programa caso o cliente não queira mais pizzas
        if mais_pizza == 'N':
            print(f'\n0 valor total a ser pago: R$ {acumulador:.2f}')
            break
    except ValueError:
        print('\nEntrada inválida. Por favor, tente novamente.')
```

Apresentação de *Saída do Console da Questão 2:*

```
PS C:\Users\Wanderson\Documents\GitHub\UNINTER\Lógica de Programação e Algoritmos> & "C:/Program Files/Python312/python.exe" "c:/Users/Wanderson/Documents/GitHub/UNINTER/Lógica de Programação e Algoritmos/trabalho2.py"
----- Bem-vindo a Pizzaria do Wanderson Teixeira -----
----- Cardápio -----
---| Tamanho | Pizza Salgada (PS) | Pizza Doce (PD) |---
---| P       | R$ 30.00           | R$ 34.00         |---
---| M       | R$ 45.00           | R$ 48.00         |---
---| G       | R$ 60.00           | R$ 66.00         |---
-----

Entre com o sabor desejado (PS/PD): PG
Sabor inválido. Tente novamente.

Entre com o sabor desejado (PS/PD): PS
Entre com o tamanho desejado (P/M/G): EXGG
Tamanho inválido. Tente novamente.

Entre com o tamanho desejado (P/M/G): G

Você pediu uma Pizza Salgada no tamanho G: R$ 60.00 reais

Deseja mais alguma coisa? (S/N): S

Entre com o sabor desejado (PS/PD): PD
Entre com o tamanho desejado (P/M/G): M

Você pediu uma Pizza Doce no tamanho M: R$ 48.00 reais

Deseja mais alguma coisa? (S/N): N

O valor total a ser pago: R$ 108.00
PS C:\Users\Wanderson\Documents\GitHub\UNINTER\Lógica de Programação e Algoritmos> |
```

QUESTÃO 3 de 4 - Conteúdo até aula 05

Enunciado: Você foi contratado para desenvolver um sistema de Venda de uma Empresa Y que vende toras de arvore para outras empresas que vendem madeira. Você ficou com a parte de desenvolver a interface com o cliente.

A Empresa Y opera as vendas da seguinte maneira:

- Tora de Pinho (PIN), o valor do metro cúbico (m³) é de cento e cinquenta reais e quarenta centavos;
 - Tora de Peroba (PER), o valor do metro cúbico (m³) é de cento e setenta reais e vinte centavos;
 - Tora de Mogno (MOG), o valor do metro cúbico (m³) é de cento e noventa reais e noventa centavos;
 - Tora de Ipê (IPE), o valor do metro cúbico (m³) é de duzentos e dez reais e dez centavos;
 - Tora de Imbuia (IMB), o valor do metro cúbico (m³) é de duzentos e vinte reais e setenta centavos;
-
- Se a quantidade (em m³) de toras for **menor** que 100 não há desconto na venda (0/100);
 - Se a quantidade (em m³) de toras for **igual ou maior** que 100 e **menor** que 500, o desconto será de 4% (4/100);
 - Se a quantidade (em m³) de toras for **igual ou maior** que 500 e **menor** que 1000, o desconto será de 9% (9/100);
 - Se a quantidade (em m³) de toras for **igual ou maior** que 1000 e **menor ou igual** que 2000, o desconto será de 16% (16/100);
 - Se a quantidade (em m³) de toras for **maior** que 2000, não é aceito pedidos com essa quantidade de toras;
-
- ♦ Para o **adicional** de transporte rodoviário (1) é cobrado um valor **extra** de 1000 reais;
 - ♦ Para o **adicional** de transporte ferroviário (2) é cobrado um valor **extra** de 2000 reais;
 - ♦ Para o **adicional** de transporte hidroviário (3) é cobrado um valor **extra** de 2500 reais;


O valor final da conta é calculado da seguinte maneira:

total = ((**tipoMadeira * qtdToras**)*(**1-desconto**)) + **transporte**

Elabore um programa em Python que:

- A. Deve-se implementar o **print** com o seu **nome completo** (somente print, não usar input aqui).
Por exemplo: **print(“Bem-vindos a Madeireira do Lenhador Bruno Kostiuk”)** [EXIGÊNCIA DE CÓDIGO 1 de 7];
- B. Deve-se implementar a função **escolha_tipo()** que **não** recebe parâmetros e que: [EXIGÊNCIA DE CÓDIGO 2 de 7];
 - a. Pergunta o **tipo de madeira** desejado;
 - b. **Retorna o VALOR do tipo de madeira** com base na escolha do usuário (use **return**);
 - c. Repete a pergunta do item **B.a** se digitar uma opção diferente de: PIN/PER/MOG/IPE/IMB;
- C. Deve-se implementar a função **qtd_toras()** que **não** recebe parâmetros e que: [EXIGÊNCIA DE CÓDIGO 3 de 7];
 - a. Pergunta a **quantidade de toras**;
 - b. **Retorna** (use **return**) a **quantidade de toras E o valor do desconto** (os dois valores) seguindo a regra do enunciado;
 - c. Repete a pergunta do item **C.a** se digitar um valor acima de 2000 ou valor não numérico (use try/except para não numérico)
- D. Deve-se implementar a função **transporte()** que **não** recebe parâmetros e que: [EXIGÊNCIA DE CÓDIGO 4 de 7];
 - a. Pergunta pelo serviço **adicional de transporte**;
 - b. **Retorna** (use **return**) o **valor** de apenas uma das **opções de transporte**;
 - c. Repetir a pergunta item **D.a** se digitar uma opção diferente de: 1/2/3;
- E. Deve-se implementar o total a pagar no código principal (**main**), ou seja, não pode estar dentro de função, conforme o enunciado [EXIGÊNCIA DE CÓDIGO 5 de 7];
- F. Deve-se implementar **try/except** [EXIGÊNCIA DE CÓDIGO 6 de 7];
- G. Deve-se inserir comentários relevantes no código [EXIGÊNCIA DE CÓDIGO 7 de 7];
- H. Deve-se apresentar na saída de console uma mensagem com o seu nome completo [EXIGÊNCIA DE SAÍDA DE CONSOLE 1 de 4];
- I. Deve-se apresentar na saída de console um pedido no qual o usuário errou a opção de tipo de madeira [EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 4];
- J. Deve-se apresentar na saída de console um pedido no qual o usuário digitou um valor que ultrapasse a quantidade máxima de toras aceitas (2000) [EXIGÊNCIA DE SAÍDA DE CONSOLE 3 de 4];
- K. Deve-se apresentar na saída de console um pedido com opção de tipo de madeira, quantidade de toras e transporte válidos [EXIGÊNCIA DE SAÍDA DE CONSOLE 4 de 4];

EXEMPLO DE SAÍDA DE CONSOLE:



Bem vindo a Madeireira do Lenhador Bruno Kostiuk

Nome completo

Entre com o Tipo de Madeira desejado

PIN - Tora de Pinho

PER - Tora de Peroba

MOG - Tora de Mogno

IPE - Tora de Ipê

IMB - Tora de Imbuia

Errou o tipo de Madeira

>>TÁBUA

Escolha inválida, entre com o modelo novamente

Entre com o Tipo de Madeira desejado

PIN - Tora de Pinho

PER - Tora de Peroba

MOG - Tora de Mogno

IPE - Tora de Ipê

IMB - Tora de Imbuia

Errou a quantidade de toras

>>IPE

Entre com a quantidade de toras (m³): 500000

Não aceitamos pedidos com essa quantidade de toras.

Por favor, entre com a quantidade novamente.

Entre com a quantidade de toras (m³): 500

Escolha o tipo de Transporte:

1 - Transporte Rodoviário - R\$ 1000.00

2 - Transporte Ferroviário - R\$ 2000.00

3 - Transporte Hidroviário - R\$ 2500.00

>>3

Total: R\$ 98095.50

Pedido com tipo de tora, quantidade de tora e transporte válidos

Figura 3.1: Exemplo de saída de console que o aluno deve fazer. Em que se pergunta pelo tipo de tora e se erra opção inicialmente, e que se passa a quantidade de toras acima do aceito. Na sequência, o usuário digitou um tipo de tora, quantidade de toras e transporte válidos.

Apresentação de Código da Questão 3:

```
"""
Calcula o valor total da compra de toras de madeira, considerando o tipo de madeira, a quantidade de toras, o desconto proporcional à quantidade, e o custo do transporte.

Parâmetros:
    valor_madeira (str): O tipo de madeira escolhido, que pode ser:
        - 'PIN': Pinho
        - 'PER': Peroba
        - 'MOG': Mogno
        - 'IPE': Ipê
        - 'IMB': Imbuia
    quantidade (float): A quantidade de toras em metros cúbicos (m³). O valor deve estar entre 0 e 2000.
    desconto (float): Percentual de desconto aplicado com base na quantidade de toras:
        - 0% para menos de 100 m³
        - 4% para entre 100 e 499 m³
        - 9% para entre 500 e 999 m³
        - 16% para 1000 m³ ou mais
    valor_transporte (float): O custo do transporte, que pode ser:
        - 1000.00 para transporte rodoviário
        - 2000.00 para transporte ferroviário
        - 2500.00 para transporte hidroviário

Retorno: str: O valor total a pagar, formatado com duas casas decimais (ex: "R$ 12345.67").
"""

def escolha_tipo():
    # Dicionário com o preço por tipo de madeira
    madeiras_validas = {'PIN': 150.40, 'PER': 170.20, 'MOG': 190.90, 'IPE': 210.10, 'IMB': 220.70}
    while True:
        # Solicita o tipo de madeira ao usuário
        madeira = input('\nEntre com o Tipo de Madeira desejado\nPIN - Tora de Pinho\nPER - Tora de Peroba\nMOG - Tora de Mogno\nIPE - Tora de Ipê\nIMB - Tora de Imbuia\nQual código: ').upper()
        if madeira in madeiras_validas:
            return madeiras_validas[madeira]
        else:
            print('Escolha inválida, entre com o modelo novamente.')

def qtd_toras():
    while True:
        try:
            # Solicita a quantidade de toras ao usuário
            quantidade = float(input('\nEntre com a quantidade de toras (m³): ')) # Entrada de quantidade de madeira
            if quantidade < 0 or quantidade > 2000:
                # Verifica se a quantidade está dentro do limite permitido
                print('Não aceitamos pedidos com essa quantidade de toras. Entre novamente.')
            # Condição de desconto
            elif quantidade < 100:
                return quantidade, 0
            elif quantidade < 500:
                return quantidade, 4
            elif quantidade < 1000:
                return quantidade, 9
            else:
                return quantidade, 16
        except ValueError:
            # Tratamento para entradas inválidas
            print('Quantidade inválida. Por favor, insira um número.')

def transporte():
    while True:
        try:
            # Solicita o tipo de transporte ao usuário
            opcao = int(input('\nEscolha o tipo de Transporte:\n1- Transporte Rodoviário - R$ 1000.00\n2 - Transporte Ferroviário - R$ 2000.00\n3 - Transporte Hidroviário - R$ 2500.00\nQual
transporte (1/2/3)? '))
            # Retorna valor para transporte rodoviário
            if opcao == 1:
                return 1000
            elif opcao == 2:
                return 2000
            elif opcao == 3:
                return 2500
            else:
                print('Opção inválida.')
        except ValueError:
            # Tratamento para entradas inválidas
            print('Escolha inválida. Insira um número.')

print("Bem-vindos a Madeireira do Wanderson Teixeira Sousa") # Saudação inicial
def main():
    # Executa as funções para coletar as informações necessárias
    valor_madeira = escolha_tipo() # Obtém o valor da madeira escolhida
    quantidade, desconto = qtd_toras() # Obtém a quantidade de toras e o desconto
    valor_transporte = transporte() # Obtém o valor do transporte

    # Calcula o valor total, aplicando o desconto e somando o custo do transporte
    total = (valor_madeira * quantidade * (1 - desconto / 100)) + valor_transporte
    print(f'Total a pagar: R$ {total:.2f}')

# Executa a função principal
main()
```

Apresentação de Saída do Console da Questão 3:

```
C:\Windows\system32\cmd.e: X + v

C:\Users\Wanderson\Documents\GitHub\UNINTER\Lógica de Programação e Algoritmos>python trabalho3.py
Bem-vindos a Madeireira do Wanderson Teixeira Sousa

Entre com o Tipo de Madeira desejado
PIN - Tora de Pinho
PER - Tora de Peroba
MOG - Tora de Mogno
IPE - Tora de Ipê
IMB - Tora de Imbuia
Qual código: TÁBUA
Escolha inválida, entre com o modelo novamente.

Entre com o Tipo de Madeira desejado
PIN - Tora de Pinho
PER - Tora de Peroba
MOG - Tora de Mogno
IPE - Tora de Ipê
IMB - Tora de Imbuia
Qual código: IPE

Entre com a quantidade de toras (m³): 500000
Não aceitamos pedidos com essa quantidade de toras. Entre novamente.

Entre com a quantidade de toras (m³): 500

Escolha o tipo de Transporte:
1- Transporte Rodoviário - R$ 1000.00
2 - Transporte Ferroviário - R$ 2000.00
3 - Transporte Hidroviário - R$ 2500.00
Qual transporte (1/2/3)? 3
Total a pagar: R$ 98095.50

C:\Users\Wanderson\Documents\GitHub\UNINTER\Lógica de Programação e Algoritmos>
```

QUESTÃO 4 de 4 - Conteúdo até aula 06

Enunciado: Você e sua equipe de programadores foram contratados por uma pequena empresa para desenvolver um software de gerenciamento de Contatos Comerciais. Este software deve ter o seguinte menu e opções:

- 1) Cadastrar Contato
- 2) Consultar Contato
 - 1. Consultar Todos
 - 2. Consultar por Id
 - 3. Consultar por Atividade
 - 4. Retornar ao menu
- 3) Remover Contato
- 4) Encerrar Programa

Elabore um programa em Python que:

- A. Deve-se implementar o **print** com o seu **nome completo** (somente print, não usar input aqui).
Por exemplo: **print(“Bem vindos a lista de contatos do Bruno Kostiuk”)** [EXIGÊNCIA DE CÓDIGO 1 de 8];
- B. Deve-se implementar uma lista com o nome de **lista_contatos** e a variável **id_global** com valor inicial igual ao número de seu RU [EXIGÊNCIA DE CÓDIGO 2 de 8];
- C. Deve-se implementar uma função chamada **cadastrar_contato(id)** que recebe **apenas id** como parâmetro e que: [EXIGÊNCIA DE CÓDIGO 3 de 8];
 - a. Pergunta **nome, atividade, telefone** do contato;
 - b. Armazena o **id** (este é fornecido via parâmetro da função), **nome, atividade, telefone** dentro de um dicionário;
 - c. **Copiar** o dicionário para dentro da **lista_contatos** (utilizar o **copy**);
- D. Deve-se implementar uma função chamada **consultar_contatos()** que **não** recebe parâmetros e que: [EXIGÊNCIA DE CÓDIGO 4 de 8];
 - a. Deve-se perguntar qual opção deseja (1. Consultar Todos / 2. Consultar por Id / 3. Consultar por Setor / 4. Retornar ao menu):
 - i. Se Consultar Todos, apresentar todos os contatos com todos os seus dados cadastrados;
 - ii. Se Consultar por Id, solicitar ao usuário que informe um id, e apresentar o contato **específico** (apenas 1) com todos os seus dados cadastrados;
 - iii. Se Consultar por Atividade, solicitar ao usuário que informe a atividade, e apresentar o(s) contato(s) que exercem aquela atividade com todos os seus dados cadastrados;
 - iv. Se Retornar ao menu, deve-se **retornar** ao menu principal (return);
 - v. Se Entrar com um valor diferente de 1, 2, 3 ou 4, printar “Opção inválida” e repetir a pergunta **D.a.**
 - vi. Enquanto o usuário não escolher a opção 4, o menu consultar contatos deve se repetir.
- E. Deve-se implementar uma função chamada **remover_contato()** em que: [EXIGÊNCIA DE CÓDIGO 5 de 8];
 - a. Deve-se pergunta pelo **id** do contato a ser removido;
 - b. Remover o contato da **lista_contatos**;
 - c. Se o id fornecido não for de um contato da lista, printar “**Id inválido**” e repetir a pergunta **E.a.**
- F. Deve-se implementar uma estrutura de menu no código principal (**main**), ou seja, **não pode estar dentro de função**, em que: [EXIGÊNCIA DE CÓDIGO 6 de 8];
 - a. Deve-se pergunta qual opção deseja (1. Cadastrar Contato / 2. Consultar Contato / 3. Remover Contato / 4. Encerrar Programa):
 - i. Se Cadastrar Contato, **incrementar** em um **id_global** e **em seguida**, chamar a função **cadastrar_contato (id_global)**;
 - ii. Se Consultar Contato, chamar função **consultar_contato ()**;
 - iii. Se Remover Contato, chamar função **remover_ contato ()**;
 - iv. Se Encerrar Programa, sair do menu (e com isso acabar a execução do código);
 - v. Se Entrar com um valor diferente de 1, 2, 3 ou 4, printar “Opção inválida” e repetir a pergunta **F.a.**
 - vi. Enquanto o usuário não escolher a opção 4, o menu deve se repetir.
- G. Deve-se implementar uma **lista de dicionários** (uma lista contento dicionários dentro) [EXIGÊNCIA DE CÓDIGO 7 de 8];
- H. Deve-se inserir comentários **relevantes** no código [EXIGÊNCIA DE CÓDIGO 8 de 8];
- I. Deve-se apresentar na saída de console um cadastro do **seu contato** da seguinte forma: para **nome** informe seu **nome completo** (não usar apelidos ou abreviações), para **atividade** informar como **estudante**, e para **telefone** informe sua **RU**. [EXIGÊNCIA DE SAÍDA DE CONSOLE 1 de 6];
- J. Deve-se apresentar na saída de console um cadastro de **mais 2** contatos com mesmo tipo de atividade (por exemplo: marceneiro, padeiro, pintor, pedreiro) [EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 6];
- K. Deve-se apresentar na saída de console uma consulta de todos os contatos [EXIGÊNCIA DE SAÍDA DE CONSOLE 3 de 6];
- L. Deve-se apresentar na saída de console uma consulta por código (id) de um dos contados [EXIGÊNCIA DE SAÍDA DE CONSOLE 4 de 6];
- M. Deve-se apresentar na saída de console uma consulta por atividade em que **2** contatos exerçam a mesma atividade [EXIGÊNCIA DE SAÍDA DE CONSOLE 5 de 6];
- N. Deve-se apresentar na saída de console uma remoção de um dos contatos e em seguida de uma consulta de todos os contatos, provando que o contato foi removido [EXIGÊNCIA DE SAÍDA DE CONSOLE 6 de 6];

EXEMPLO DE SAÍDA DE CONSOLE:

```
➡ Bem vindo a Lista de Contatos do Bruno Kostiuk Nome completo
-----
----- MENU PRINCIPAL -----
Escolha a opção desejada:
1 - Cadastrar Contato
2 - Consultar Contato(s)
3 - Remover Contato
4 - Sair
>>1
-----
----- MENU CADASTRAR CONTATO -----
Id do Contato: 4297914
Por favor entre com o nome do Contato: Bruno Kostiuk
Por favor entre com a Atividade do contato: Estudante
Por favor entre com o telefone do contato: 4297913
-----
```

Figura 4.1: Exemplo de saída de console que o aluno deve fazer. Apresenta o print com seu nome completo e é realizado o cadastro do primeiro contato, note que o ID do contato não inicia em 1, pois ele deve iniciar com o seu RU (caso o RU informado não seja o seu, irá receber zero em toda questão). O primeiro contato deve ser cadastrado com SEU NOME COMPLETO, em Atividade informe Estudante e em Contato informe o SEU RU.

```
➡ -----
----- MENU PRINCIPAL -----
Escolha a opção desejada:
1 - Cadastrar Contato
2 - Consultar Contato(s)
3 - Remover Contato
4 - Sair
>>1
-----
----- MENU CADASTRAR CONTATO -----
Id do Contato: 4297915
Por favor entre com o nome do Contato: Tamy
Por favor entre com a Atividade do contato: Professor
Por favor entre com o telefone do contato: 99998888
-----
----- MENU PRINCIPAL -----
Escolha a opção desejada:
1 - Cadastrar Contato
2 - Consultar Contato(s)
3 - Remover Contato
4 - Sair
>>1
-----
----- MENU CADASTRAR CONTATO -----
Id do Contato: 4297916
Por favor entre com o nome do Contato: Osmar
Por favor entre com a Atividade do contato: Professor
Por favor entre com o telefone do contato: 88889999
-----
```

Figura 4.2: Exemplo de saída de console que o aluno deve fazer. São cadastrados mais dois contatos com mesmo tipo de Atividade.


```
-----  
----- MENU CONSULTAR CONTATOS -----  
Escolha a opção desejada:  
1 - Consultar Todos os Contatos  
2 - Consultar Contato por id  
3 - Consultar Contato(s) por Atividade  
4 - Retornar  
>>1  
-----  
id: 4297914  
nome: Bruno KostiuK  
atividade: Estudante  
telefone: 4297913  
  
id: 4297915  
nome: Tamy  
atividade: Professor  
telefone: 99998888  
  
id: 4297916  
nome: Osmar  
atividade: Professor  
telefone: 88889999  
-----
```

Figura 4.3: Exemplo de saída de console que o aluno deve fazer. Em que se consulta Todos os contatos cadastrados.

```
Escolha a opção desejada:  
1 - Consultar Todos os Contatos  
2 - Consultar Contato por id  
3 - Consultar Contato(s) por Atividade  
4 - Retornar  
>>2  
Digite o id do contato: 4297914  
-----  
id: 4297914  
nome: Bruno KostiuK  
atividade: Estudante  
telefone: 4297913  
  
-----  
  
-----  
----- MENU CONSULTAR CONTATOS -----  
Escolha a opção desejada:  
1 - Consultar Todos os Contatos  
2 - Consultar Contato por id  
3 - Consultar Contato(s) por Atividade  
4 - Retornar  
>>3  
Digite a Atividade do(s) Contato(s): Professor  
-----  
id: 4297915  
nome: Tamy  
atividade: Professor  
telefone: 99998888  
  
id: 4297916  
nome: Osmar  
atividade: Professor  
telefone: 88889999  
-----
```

Figura 4.4: Exemplo de saída de console que o aluno deve fazer. Em que se consulta o contato com id número 4297914 e consulta pelo nome da Atividade (Professor).

```
➡ -----  
----- MENU PRINCIPAL -----  
Escolha a opção desejada:  
1 - Cadastrar Contato  
2 - Consultar Contato(s)  
3 - Remover Contato  
4 - Sair  
                                     Remove um contato  
>>3  
-----  
----- MENU REMOVER CONTATO -----  
Digite o id do contato a ser removido: 4297914  
Contato removido com sucesso!  
-----  
----- MENU PRINCIPAL -----  
Escolha a opção desejada:  
1 - Cadastrar Contato  
2 - Consultar Contato(s)  
3 - Remover Contato  
4 - Sair  
>>2  
-----  
----- MENU CONSULTAR CONTATOS -----  
Escolha a opção desejada:  
1 - Consultar Todos os Contatos  
2 - Consultar Contato por id  
3 - Consultar Contato(s) por Atividade  
4 - Retornar  
>>1  
-----  
id: 4297915  
nome: Tamy  
atividade: Professor  
telefone: 99998888  
                                     Realiza o consultar Todos mostrando  
                                     que o contato foi removido  
  
id: 4297916  
nome: Osmar  
atividade: Professor  
telefone: 88889999  
-----
```

Figura 4.5: Exemplo de saída de console que o aluno deve fazer. Em que se remove o contato de Id número 4297914 e depois se faz uma consulta de todos os contatos.

Apresentação de **Código da Questão 4:**

```
def valida_int(pergunta, min, max=None):
    """
    Função para validar a entrada do usuário, garantindo que seja um inteiro dentro de um intervalo.

    :param pergunta: A pergunta a ser exibida ao usuário.
    :param min: O valor mínimo permitido.
    :param max: O valor máximo permitido (opcional).
    :return: Retorna um número inteiro válido.
    """

    while True:
        try:
            x = int(input(pergunta))
            if x < min:
                print(f"Por favor, insira um valor maior ou igual a {min}.")
            elif max is not None and x > max:
                print(f"Por favor, insira um valor entre {min} e {max}.")
            else:
                return x # Retorna o valor válido
        except ValueError:
            print("Entrada inválida. Por favor, insira um número válido.")

# Lista para armazenar contatos
lista_contatos = []
id_global = 2412021 # Meu ID ALUNO (ID INICIAL PARA CADASTRO)

def cadastrar_contato(id):
    """
    Função para cadastrar um novo contato.

    :param id: O ID do novo contato.
    """

    # Pergunta informações do contato
    print(f'Id do Contato: {id_global}')
    nome = input('Por favor entre com o nome do Contato: ')
    atividade = input('Por favor entre com a Atividade do contato: ')
    telefone = input('Por favor entre com o telefone do contato: ')

    contato = {
        'id': id,
        'nome': nome,
        'atividade': atividade,
        'telefone': telefone
    }
    lista_contatos.append(contato.copy()) # Adiciona contato à lista
    print('Cadastro feito com sucesso...')

def consultar_contatos():
    """
    Função para consultar contatos cadastrados.
    """

    while True:
        print(20 * '-' + ' MENU CONSULTAR CONTATO ' + 20 * '-')
        item_escolha = valida_int('Escolha a opção desejada: \n1 - Consultar Todos os Contatos\n2 - Consultar Contato por id\n3 - Consultar Contato(s) por Atividade\n4 - Retornar\nQual escolha: ', 1, 4)

        if item_escolha == 1:
            listar_contatos()
        elif item_escolha == 2:
            id_contato = valida_int('Digite o Id do contato: ', 1)
            listar_contato_por_id(id_contato)
        elif item_escolha == 3:
            atividade_contato = input('Digite a Atividade do contato: ')
            listar_contatos_por_atividade(atividade_contato)
        elif item_escolha == 4:
            print('Retornando ao menu principal...\n')
            return # Retorna ao menu principal
        else:
            print('Opção inválida.')
```

```
def listar_contatos():
    """
    Função para listar todos os contatos cadastrados.
    """

    if not lista_contatos:
        print('Nenhum contato cadastrado.')
    else:
        for contato in lista_contatos:
            print(f"Id: {contato['id']}\nNome: {contato['nome']}\nAtividade: {contato['atividade']}\nTelefone: {contato['telefone']}\n")

def listar_contato_por_id(id_contato):
    """
    Função para listar um contato específico pelo ID.

    :param id_contato: O ID do contato a ser listado.
    """

    for contato in lista_contatos:
```

```
        if contato['id'] == id_contato:
            print(f'Id: {contato["id"]}\nNome: {contato["nome"]}\nAtividade: {contato["atividade"]}\nTelefone: {contato["telefone"]}\n')
            return
    print(f'Contato com Id {id_contato} não encontrado.')

def listar_contatos_por_atividade(atividade_contato):
    """
    Função para listar contatos com base na atividade.

    :param atividade_contato: A atividade que será usada para filtrar os contatos.
    """
    encontrou = False
    for contato in lista_contatos:
        if contato['atividade'] == atividade_contato:
            print(f'Id: {contato["id"]}\nNome: {contato["nome"]}\nAtividade: {contato["atividade"]}\nTelefone: {contato["telefone"]}\n')
            encontrou = True
    if not encontrou:
        print(f'Nenhum contato encontrado com a atividade "{atividade_contato}".')

def remover_contato():
    """
    Função para remover um contato da lista.
    """
    id_contato = valida_int('Digite o Id do contato que deseja remover: ', 1)
    for contato in lista_contatos:
        if contato['id'] == id_contato:
            lista_contatos.remove(contato)
            print(f'Contato com Id {id_contato} removido com sucesso.')
            return
    print(f'Contato com Id {id_contato} não encontrado.')

def menu():
    """
    Função para exibir o menu principal do programa.
    """
    print(64 * '-')
    print(24 * '-' + ' MENU PRINCIPAL ' + 24 * '-')
    print('Escolha a opção desejada:')
    print('1 - Cadastrar Contato')
    print('2 - Consultar Contato')
    print('3 - Remover Contato')
    print('4 - Sair')
    print(64 * '-')

# Início do programa
print('\nBem-vindo à Lista de Contatos do Wanderson Teixeira')
while True:
    menu() # Exibe o menu principal
    item_menu = valida_int('Escolha qual item do menu: ', 1, 4)
    print()
    if item_menu == 4:
        print('Encerrando o programa...')
        break # Encerra o programa
    elif item_menu == 1:
        print(20 * '-' + ' MENU CADASTRAR CONTATO ' + 20 * '-')
        id_global += 1 # Incrementa o ID global para o próximo contato
        cadastrar_contato(id_global) # Chama a função para cadastrar um novo contato

    elif item_menu == 2:
        consultar_contatos() # Chama a função para consultar contatos

    elif item_menu == 3:
        print(20 * '-' + ' MENU REMOVER CONTATO ' + 20 * '-')
        remover_contato() # Chama a função para remover um contato
```


Apresentação de Saída do Console da Questão 4:

Figura 4.1

```
Bem-vindo à Lista de Contatos do Wanderson Teixeira
-----
----- MENU PRINCIPAL -----
Escolha a opção desejada:
1 - Cadastrar Contato
2 - Consultar Contato
3 - Remover Contato
4 - Sair
-----

Escolha qual item do menu: 1

----- MENU CADASTRAR CONTATO -----
Id do Contato: 2412022
Por favor entre com o nome do Contato: Wanderson Teixeira Sousa
Por favor entre com a Atividade do contato: Estudante
Por favor entre com o telefone do contato: 2412022
Cadastro feito com sucesso...
-----
```

Figura 4.2

```
----- MENU PRINCIPAL -----
Escolha a opção desejada:
1 - Cadastrar Contato
2 - Consultar Contato
3 - Remover Contato
4 - Sair
-----

Escolha qual item do menu: 1

----- MENU CADASTRAR CONTATO -----
Id do Contato: 2412023
Por favor entre com o nome do Contato: Tamy
Por favor entre com a Atividade do contato: Professor
Por favor entre com o telefone do contato: 99998888
Cadastro feito com sucesso...
-----

----- MENU PRINCIPAL -----
Escolha a opção desejada:
1 - Cadastrar Contato
2 - Consultar Contato
3 - Remover Contato
4 - Sair
-----

Escolha qual item do menu: 1

----- MENU CADASTRAR CONTATO -----
Id do Contato: 2412024
Por favor entre com o nome do Contato: Osmar
Por favor entre com a Atividade do contato: Professor
Por favor entre com o telefone do contato: 88889999
Cadastro feito com sucesso...
-----
```

Figura 4.3

```
----- MENU CONSULTAR CONTATO -----
Escolha a opção desejada:
1 - Consultar Todos os Contatos
2 - Consultar Contato por id
3 - Consultar Contato(s) por Atividade
4 - Retornar
Qual escolha: 1
Id: 2412022
Nome: Wanderson Teixeira Sousa
Atividade: Estudante
Telefone: 2412022

Id: 2412023
Nome: Tamy
Atividade: Professor
Telefone: 99998888

Id: 2412024
Nome: Osmar
Atividade: Professor
Telefone: 88889999
```

Figura 4.4

```
----- MENU CONSULTAR CONTATO -----
Escolha a opção desejada:
1 - Consultar Todos os Contatos
2 - Consultar Contato por id
3 - Consultar Contato(s) por Atividade
4 - Retornar
Qual escolha: 2
Digite o Id do contato: 2412022
Id: 2412022
Nome: Wanderson Teixeira Sousa
Atividade: Estudante
Telefone: 2412022

----- MENU CONSULTAR CONTATO -----
Escolha a opção desejada:
1 - Consultar Todos os Contatos
2 - Consultar Contato por id
3 - Consultar Contato(s) por Atividade
4 - Retornar
Qual escolha: 3
Digite a Atividade do contato: Professor
Id: 2412023
Nome: Tamy
Atividade: Professor
Telefone: 99998888

Id: 2412024
Nome: Osmar
Atividade: Professor
Telefone: 88889999
```

Figura 4.5

```
----- MENU PRINCIPAL -----
Escolha a opção desejada:
1 - Cadastrar Contato
2 - Consultar Contato
3 - Remover Contato
4 - Sair

Escolha qual item do menu: 3

----- MENU REMOVER CONTATO -----
Digite o Id do contato que deseja remover: 2412022
Contato com Id 2412022 removido com sucesso.

----- MENU PRINCIPAL -----
Escolha a opção desejada:
1 - Cadastrar Contato
2 - Consultar Contato
3 - Remover Contato
4 - Sair

Escolha qual item do menu: 2

----- MENU CONSULTAR CONTATO -----
Escolha a opção desejada:
1 - Consultar Todos os Contatos
2 - Consultar Contato por id
3 - Consultar Contato(s) por Atividade
4 - Retornar
Qual escolha: 1
Id: 2412023
Nome: Tamy
Atividade: Professor
Telefone: 99998888

Id: 2412024
Nome: Osmar
Atividade: Professor
Telefone: 88889999
```