**ANO**

**2025**



**ATIVIDADE PRÁTICA**

ESTRUTURA DE DADOS /

PROGRAMAÇÃO III

**Emanuel Rosa Zolet. RU: 4557826**

**Prof. Me. Bruno Kostiuk**

QUESTÃO 1 de 2 – Lista Encadeada

**Enunciado:** Com a finalidade de melhorar o atendimento e priorizar os casos mais urgentes, a direção de um hospital criou um sistema de triagem em que um profissional da saúde classifica a ordem de atendimento com base numa avaliação prévia do paciente, entregando-lhe um cartão numerado verde (V) ou amarelo (A), que define o menor ou maior grau de urgência da ocorrência, respectivamente. Para informatizar esse processo, a direção do hospital contratou você para desenvolver uma fila de chamada seguindo as seguintes regras:

* Pacientes com cartão numerado amarelo (A) são chamados antes dos pacientes com cartão numerado verde (V)
* Entre os pacientes com cartão numerado amarelo (A), os que tem numeração menor são atendidos antes.
* Entre os pacientes com cartão numerado verde (V), os que tem numeração menor são atendidos antes.
* As numerações dos cartões verdes (V) iniciam em 1.
* As numerações dos cartões amarelos (A) iniciam em 201.

Elabore um programa em Python que:

1. Deve-se implementar uma **Lista Encadeada Simples** em que**:** [EXIGÊNCIA DE CÓDIGO 1 de 7];
   1. O Nodo representa um cartão numerado contendo: **número**, **cor** e um ponteiro para o **próximo**;
   2. A lista é não circular, ou seja, seu último elemento aponta para nulo;
2. Deve-se implementar a função **inserirSemPrioridade(nodo)** em que**:** [EXIGÊNCIA DE CÓDIGO 2 de 7];
   1. Deve-se andar pela lista a partir da cabeça (***head***) e inserir o nodo no **final da lista**.
3. Deve-se implementar a função **inserirComPrioridade(nodo)** em que**:** [EXIGÊNCIA DE CÓDIGO 3 de 7];
   1. Deve-se andar pela lista a partir da cabeça (***head***) e inserir o nodo **após todos os nodos com cor “A” que estão na lista**.
   2. O nodo inserido deve **sempre** estar antes de todos os nodos com cor “V”.
4. Deve-se implementar a função **inserir()** em que**:** [EXIGÊNCIA DE CÓDIGO 4 de 7];
   1. Deve-se solicitar ao usuário a cor (“A” ou “V”).
   2. A partir da cor, o número (inteiro) do paciente deve ser **atribuído automaticamente** seguindo a ordem numérica. Por exemplo: o primeiro paciente “V” será o 1, o segundo 2, e assim por diante.
   3. Deve-se criar um nodo com a cor e o número atribuído ao paciente.
   4. Se a lista estiver vazia, a cabeça (***head***) da lista deve apontar para o nodo criado. Senão, se a cor do nodo for “V”, deve-se chamar a função **inserirSemPrioridade(nodo).** Senão, se a cor do nodo for “A”, deve-se chamar a função **inserirComPriordade(nodo).**
5. Deve-se implementar a função **imprimirListaEspera()** em que: [EXIGÊNCIA DE CÓDIGO 5 de 7];
   1. Deve-se imprimir todos os cartões e seus respectivos números a partir do primeiro até o último da lista.
6. Deve-se implementar a função **atenderPaciente()** em que: [EXIGÊNCIA DE CÓDIGO 6 de 7];
   1. Deve-se remover o primeiro paciente da fila e imprimir uma mensagem chamando o paciente para atendimento informando o número do seu cartão.
7. Deve-se implementar um menu para utilização do sistema em que: [EXIGÊNCIA DE CÓDIGO 7 de 7];
   1. Deve-se apresentar as opções (1 – adicionar paciente a fila, 2 – mostrar pacientes na fila, 3 – chamar paciente, 4 – sair)
   2. Se escolhida a opção 1, chamar a função **inserir().**
   3. Se escolhida a opção 2, chamar a função **imprimirListaEspera().**
   4. Se escolhida a opção 3, chamar a função **atenderPaciente().**
   5. Se escolhida a opção 4, encerrar o programa.
   6. Se escolhida uma opção diferente as opções disponíveis, volte para o menu.

Para testar o software, execute os seguintes passos e apresente a saída do console conforme exemplo de saída de console (próxima página):

1. Deve-se testar o sistema inserindo três (3) pacientes com cartão de cor “V”, dois (2) pacientes com cartão de cor “A”, dois (2) pacientes com cartão “V” e três (3) pacientes com cartão de cor “A”, **nessa respectiva ordem**. [EXIGÊNCIA DE SAÍDA DE CONSOLE 1 de 3];
2. Deve-se apresentar na saída de console a impressão da lista de espera (opção 2 do menu principal). [EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 3];
3. Deve-se apresentar na saída de console o atendimento de dois (2) pacientes (opção 3 do menu principal) e em seguida mostrar a lista de espera (opção 2 do menu principal). [EXIGÊNCIA DE SAÍDA DE CONSOLE 3 de 3];

Apresentação de **Código da Questão 1**:

# Inicialização da lista encadeada simples

class Cartao:

    # Exigência de código 1

    def \_\_init\_\_(*self*, *cor*):

*self*.cor = *cor*

*self*.numero = None

*self*.proximo = None

class ListaCartoes:

    def \_\_init\_\_(*self*):

*self*.head = None

    def inserirSemPrioridade(*self*, *nodo*):

        # Exigência de código 2

        if *self*.head is None:

*self*.head = *nodo*

            return

        nodo\_atual = *self*.head

        while nodo\_atual.proximo is not None:

            nodo\_atual = nodo\_atual.proximo

        nodo\_atual.proximo = *nodo*

    def inserirComPrioridade(*self*, *nodo*):

        # Exigência de código 3

        if *self*.head is None:

*self*.head = *nodo*

            return

        anterior = None

        atual = *self*.head

        # Avança enquanto a cor for "A"

        while atual is not None and atual.cor == "A":

            anterior = atual

            atual = atual.proximo

        # Insere o nodo antes do primeiro "V" ou após os "A"

        if anterior is None:

            # Insere no início

*nodo*.proximo = *self*.head

*self*.head = *nodo*

        else:

*nodo*.proximo = atual

            anterior.proximo = *nodo*

# Exemplo de uso:

lista = ListaCartoes()

def inserir():

    # Exigência de código 4

    cor = input("Digite a cor do cartão (A ou V): ").strip().upper()

    if cor not in ["A", "V"]:

        print("Cor inválida. Tente novamente.")

        return

    if not hasattr(inserir, "contador\_v"):

        inserir.contador\_v = 1

    if not hasattr(inserir, "contador\_a"):

        inserir.contador\_a = 201

    novo\_cartao = Cartao(cor)

    if cor == "V":

        novo\_cartao.numero = inserir.contador\_v

        inserir.contador\_v += 1

        lista.inserirSemPrioridade(novo\_cartao)

    else:

        novo\_cartao.numero = inserir.contador\_a

        inserir.contador\_a += 1

        lista.inserirComPrioridade(novo\_cartao)

def imprimirListaEspera():

    # Exigência de código 5

    print("Lista de espera:")

    if lista.head is None:

        print("Lista de espera vazia.")

        return

    atual = lista.head

    while atual is not None:

        print(f"Cartão {atual.numero} - Cor: {atual.cor}")

        atual = atual.proximo

def atenderPaciente():

    # Exigência de código 6

    print("Atendendo paciente...")

    if lista.head is None:

        print("Nenhum paciente na lista de espera.")

        return

    atendido = lista.head

    lista.head = atendido.proximo

    print(f"Atendendo paciente com cartão {atendido.numero} - Cor: {atendido.cor}")

    del atendido  # Libera o nodo atendido da memória

# MAIN

while True:

    # Exigência de código 7

    print("\nMenu:")

    print("1 - Adicionar paciente à fila")

    print("2 - Mostrar pacientes na fila")

    print("3 - Atender paciente")

    print("4 - Sair")

    opcao = input("Escolha uma opção: ").strip()

    if opcao == "1":

        inserir()

    elif opcao == "2":

        imprimirListaEspera()

    elif opcao == "3":

        atenderPaciente()

    elif opcao == "4":

        print("Saindo do programa.")

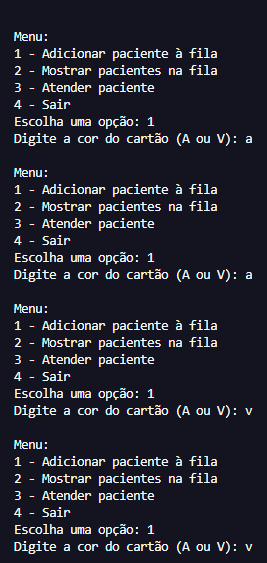
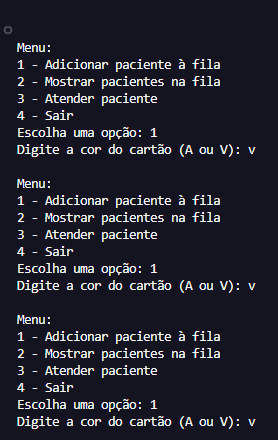
        break

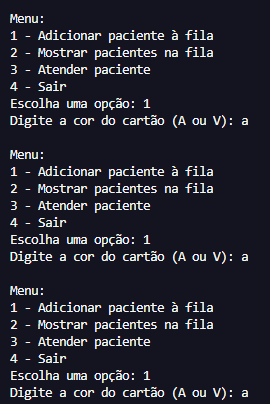
    else:

        print("Opção inválida. Tente novamente.")

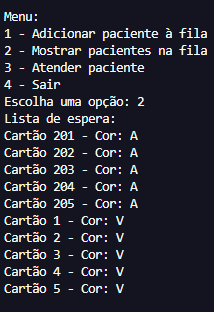
Apresentação de **Saída do Console da Questão 1**:

[EXIGÊNCIA DE SAÍDA DE CONSOLE 1 de 3];

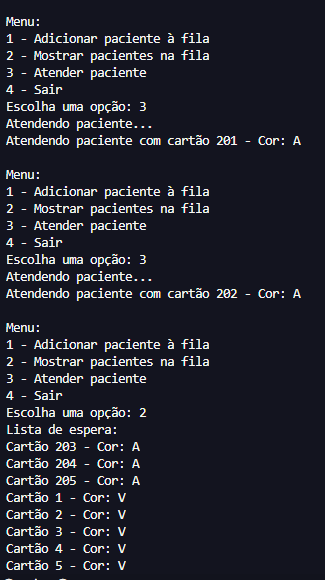
****

****

[EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 3];

****

[EXIGÊNCIA DE SAÍDA DE CONSOLE 3 de 3];



QUESTÃO 2 de 2 – Tabela Hash

**Enunciado:** Com o objetivo de criar um sistema novo de emplacamento de veículos, deputados em do Distrito Federal – DF, decidiram que o último número da placa dos veículos, irá representar o estado de registro dele. Para isso, sua equipe de desenvolvedores foi encarregada de desenvolver uma **Tabela Hash com endereçamento em cadeia de** **10 posições** (cada posição do vetor deve ser uma lista encadeada), representando os números de 0 a 9 que irão representar os 26 estados e o Distrito Federal (total 27).

A função hash deve seguir as seguintes regras:

* A entrada da função hash deve ser uma string com 2 letras, representando a sigla do estado e/ou distrito federal.
* Caso a sigla seja DF (Distrito Federal), por questões de superstição, os deputados solicitaram que o retorno da função seja 7 sempre.
* Caso contrário, a função deve retornar a posição com base no valor ASCII das duas letras e seguindo a seguinte regra:

Onde e são os valores ASCII da primeira e segunda letra, respectivamente (Tabela ASCII no final do documento).

Elabore um programa em Python que:

1. Deve-se implementar a tabela Hash com 10 posições, onde inicialmente todas as posições possuem valor **None** [EXIGÊNCIA DE CÓDIGO 1 de 7];
2. Deve-se implementar as **Listas Encadeadas Simples** em que**:** [EXIGÊNCIA DE CÓDIGO 2 de 7];
   1. O Nodo representa um Estado contendo: **sigla**, **nomeEstado** e um ponteiro para o **próximo**;
   2. As 10 posições da tabela hash, representam a cabeça de cada lista (***head***).
3. Deve-se implementar a inserção no início da lista encadeada (cada elemento novo deve ser sempre **inserido no início da lista**) [EXIGÊNCIA DE CÓDIGO 3 de 7];
4. Deve-se implementar a impressão da tabela hash, onde devem ser impressas as **siglas** de todos os nodos que estão na tabela hash **separados por posição** [EXIGÊNCIA DE CÓDIGO 4 de 7];
5. Deve-se implementar a função hash, conforme enunciado. [EXIGÊNCIA DE CÓDIGO 5 de 7];
6. Deve-se implementar a inserção dos estados e distrito federal (**todos os 27 com nome e sigla**) na tabela hash utilizando a função hash (não precisa solicitar ao usuário a digitação via teclado, pode inserir no código mesmo de modo *hard code*) [EXIGÊNCIA DE CÓDIGO 6 de 7];
7. Deve-se inserir na Tabela, além dos estados e distrito federal, um estado fictício, sendo que esse estado tenha seu **nome completo** e como siglas, a primeira letra do seu nome e a primeira letra do seu último sobrenome. Exemplo: Bruno Kostiuk – BK. EXIGÊNCIA DE CÓDIGO 7 de 7];

Para testar o software, execute os seguintes passos e apresente a saída do console conforme exemplo de saída de console (próxima página):

1. Deve-se apresentar na saída de console, a impressão da tabela hash antes de inserir qualquer informação [EXIGÊNCIA DE SAÍDA DE CONSOLE 1 de 3];
2. Deve-se apresentar na saída de console, a impressão da tabela hash após inserir os 26 estados e o Distrito Federal - DF [EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 3];
3. Deve-se apresentar na saída de console, a impressão da tabela hash após inserir os 26 estados, Distrito Federal – DF e o estado fictício com seu nome completo. [EXIGÊNCIA DE SAÍDA DE CONSOLE 3 de 3];

A imagem apresenta a tabela ASCII disponível também em:

https://www.matematica.pt/util/resumos/tabela-ascii.php

EXEMPLO DE SAÍDA DE CONSOLE:

SOMENTE INFORMATIVO. PODE APAGAR AO ENTREGAR O TRABALHO.

A imagem apresenta a seguinte saída do console:

0: None
1: None
2: None
3: None
4: None
5: None
6: None
7: None
8: None
9: None

Figura 1: Exemplo de saída de console que o aluno deve fazer. Impressão da tabela hash antes de inserir qualquer informação, conforme [EXIGÊNCIA DE SAÍDA DE CONSOLE 1 de 3];

A imagem apresenta a seguinte saída do console:

0: SC->RN->MS->GO->None
1: RO->MT->BA->AL->None
2: SE->PR->MA->ES->AM->AC->None
3: TO->SP->PI->None
4: RR->None
5: RS->PA->AP->None
6: RJ->PB->CE->None
7: DF->None
8: MG->None
9: PE->None

Figura 2: Exemplo de saída de console que o aluno deve fazer. Impressão da tabela hash após inserir os 26 estados e o Distrito Federal - DF, conforme [EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 3];

A imagem apresenta a seguinte saída do console:

0: SC->RN->MS->GO->None
1: BK->RO->MT->BA->AL->None
2: SE->PR->MA->ES->AM->AC->None
3: TO->SP->PI->None
4: RR->None
5: RS->PA->AP->None
6: RJ->PB->CE->None
7: DF->None
8: MG->None
9: PE->None

Figura 3: Exemplo de saída de console que o aluno deve fazer. Impressão da tabela hash após inserir os 26 estados, Distrito Federal – DF e o estado fictício com seu nome completo (No caso foi inserido BK na posição 1), conforme [EXIGÊNCIA DE SAÍDA DE CONSOLE 3 de 3];

Apresentação de **Código da Questão 2**:

SUBSTITUIR ESSE TEXTO QUE ESTÁ EM VERMELHO PELO SEU CÓDIGO DO EXERCÍCIO 2.

NÃO ESQUECER DE CUMPRIR AS EXIGÊNCIAS DE CÓDIGO!!

O CÓDIGO DEVE ESTAR IDENTADO!!

**SERÃO ACEITOS SOMENTE CÓDIGOS NO FORMATO TEXTO (NADA DE IMAGEM NEM PRINT, ZERA A QUESTÃO!).**

Apresentação de **Saída do Console da Questão 2**:

SUBSTITUIR ESSE TEXTO QUE ESTÁ EM LARANJA PELA A SAÍDA DO CONSOLE DO EXERCÍCIO 2

NÃO ESQUECER DE CUMPRIR AS EXIGÊNCIAS

**SERÁ ACEITO SOMETE SAÍDAS DO CONSOLE NO FORMATO IMAGEM (NADA DE TEXTO AQUI! ZERA ESSA PARTE DA QUESTÃO!)**

**CRITÉRIOS DE AVALIAÇÃO**

SOMENTE INFORMATIVO. PODE APAGAR AO ENTREGAR O TRABALHO.

A avaliação se dará através de **critérios estritamente objetivos**:

* Dentro de cada questão, metade da nota se refere as exigências de implementação do código (EC), e a outra metade as exigências de saída (ES) de funcionamento do código:
* Cada questão vale metade da nota da atividade. Assim, calculamos a sua nota final na atividade prática com a fórmula abaixo:

Os critérios e suas respectivas pontuações estão na tabela a seguir:

|  |  |  |
| --- | --- | --- |
| **QUESTÃO 1** | | |
| **Exigências de Código (EC)** | | **Pts** |
| EC 1 | Implementar corretamente a Lista Encadeada Simples | 10 |
| EC 2 | Implementar corretamente a função/método *inserirSemPrioridade()* | 15 |
| EC 3 | Implementar corretamente a função/método *inserirComPrioridade()* | 30 |
| EC 4 | Implementar corretamente a função/método *inserir()* | 15 |
| EC 5 | Implementar corretamente a função/método *imprimirListaEspera()* | 10 |
| EC 6 | Implementar corretamente a função/método *atenderPaciente()* | 10 |
| EC 7 | Implementar corretamente o menu principal | 10 |
| **Total** | | **100** |
|  | | |
| **Exigências de Saída (ES)** | | **Pts** |
| ES 1 | Apresentar a inserção de 10 pacientes | 50 |
| ES 2 | Apresentar a impressão da lista de espera | 30 |
| ES 3 | Apresentar o atendimento de 2 pacientes | 20 |
| **Total** | | **100** |

|  |  |  |
| --- | --- | --- |
| **QUESTÃO 2** | | |
| **Exigências de Código (EC)** | | **Pts** |
| EC 1 | Implementar corretamente a Tabela Hash | 20 |
| EC 2 | Implementar corretamente a Lista Encadeada Simples | 10 |
| EC 3 | Implementar corretamente a inserção no inicio | 10 |
| EC 4 | Implementar corretamente a impressão da Tabela Hash | 20 |
| EC 5 | Implementar corretamente a função hash para os estados | 10 |
| EC 6 | Implementar corretamente a função para o Distrito Federal | 10 |
| EC 7 | Implementar corretamente a inserção dos estados na tabela | 10 |
| EC 8 | Implementar corretamente a inserção de um estado com seu nome | 10 |
| **Total** | | **100** |
|  | | | |
| **Exigências de Saída (ES)** | | **Pts** |
| ES 1 | Apresentar a impressão da Tabela Hash vazia | 30 |
| ES 2 | Apresentar a impressão da Tabela Hash com 26 estados + DF | 30 |
| ES 3 | Apresentar a impressão da Tabela Hash Completa com seu nome | 40 |
| **Total** | | **100** |