

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 amarelos (A) iniciam em 201.
- As numerações dos cartões verdes (V) inicial em 1.



Elabore um programa em Python que:

- A. Deve-se implementar uma **Lista Encadeada Simples** em que: [EXIGÊNCIA DE CÓDIGO 1 de 7];
 - a. O Nodo representa um cartão numerado contendo: **número**, **cor** e um ponteiro para o **próximo**;
 - b. A lista contém um ponteiro para a cabeça da lista (**head**);
- B. Deve-se implementar a função **inserirSemPrioridade(nodo)** em que: [EXIGÊNCIA DE CÓDIGO 2 de 7];
 - a. Deve-se andar pela lista a partir da cabeça (**head**) e inserir o nodo no **final da lista**.
- C. Deve-se implementar a função **inserirComPrioridade(nodo)** em que: [EXIGÊNCIA DE CÓDIGO 3 de 7];
 - a. 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**.
 - b. O nodo inserido deve **sempre** estar antes de todos os nodos com cor “V”.
- D. Deve-se implementar a função **inserir()** em que: [EXIGÊNCIA DE CÓDIGO 4 de 7];
 - a. Deve-se solicitar ao usuário a cor (“A” ou “V”) e o número (inteiro).
 - b. Deve-se criar um nodo com a cor e o número fornecidos pelo usuário.
 - c. Se a lista estiver vazia, a cabeça (**head**) da lista deve apontar para o nodo criado.
 - d. Senão, se a cor do nodo for “V”, deve-se chamar a função **inserirSemPrioridade(nodo)**.
 - e. Senão, se a cor do nodo for “A”, deve-se chamar a função **inserirComPriordade(nodo)**.
- E. Deve-se implementar a função **imprimirListaEspera()** em que: [EXIGÊNCIA DE CÓDIGO 5 de 7];
 - a. Deve-se imprimir todos os cartões e seus respectivos números a partir do primeiro até o último da lista.
- F. Deve-se implementar a função **atenderPaciente()** em que: [EXIGÊNCIA DE CÓDIGO 6 de 7];
 - a. Deve-se remover o primeiro paciente da fila e imprimir uma mensagem chamando o paciente para atendimento informando o número do seu cartão.
- G. Deve-se implementar um menu para utilização do sistema em que: [EXIGÊNCIA DE CÓDIGO 7 de 7];
 - a. Deve-se apresentar as opções (1 – adicionar paciente a fila, 2 – mostrar pacientes na fila, 3 – chamar paciente, 4 – sair)
 - b. Se escolhida a opção 1, chamar a função **inserir()**.
 - c. Se escolhida a opção 2, chamar a função **imprimirListaEspera()**.
 - d. Se escolhida a opção 3, chamar a função **atenderPaciente()**.
 - e. Se escolhida a opção 4, encerrar o programa.
 - f. Se escolhida uma opção diferente as opções disponíveis, voltar ao item G.a.



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

- H. 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];
- I. 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];
- J. 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];



EXEMPLO DE SAÍDA DE CONSOLE:



```
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): V
Informe o número do cartão: 1
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): V
Informe o número do cartão: 2
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): V
Informe o número do cartão: 3
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): A
Informe o número do cartão: 201
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
```





```
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): A
Informe o número do cartão: 202
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): V
Informe o número do cartão: 4
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): V
Informe o número do cartão: 5
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): A
Informe o número do cartão: 203
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
```





```
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): A
Informe o número do cartão: 204
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>1
Informe a cor do cartão (A/V): A
Informe o número do cartão: 205
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
```

Figura 1: Exemplo de saída de console que o aluno deve fazer. Em que se insere 10 pacientes (5 com cartão verde e 5 com cartão amarelo) conforme [EXIGÊNCIA DE SAÍDA DE CONSOLE 1 de 3];



```
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>2
Lista -> [A,201] [A,202] [A,203] [A,204] [A,205] [V,1] [V,2] [V,3] [V,4] [V,5]
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
```

Figura 2: Exemplo de saída de console que o aluno deve fazer. Em que mostra a lista de pacientes, conforme [EXIGÊNCIA DE SAÍDA DE CONSOLE 2 de 3];



```
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>3
Atendendo o paciente cartão cor A e número 201
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>3
Atendendo o paciente cartão cor A e número 202
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>2
Lista -> [A,203] [A,204] [A,205] [V,1] [V,2] [V,3] [V,4] [V,5]
1 - Adicionar paciente a fila
2 - Mostrar pacientes na fila
3 - Chamar paciente
4 - sair
>>4
```

Figura 3: Exemplo de saída de console que o aluno deve fazer. Em que ele chama dois pacientes para atendimento e em seguida mostra a lista de pacientes, conforme [EXIGÊNCIA DE SAÍDA DE CONSOLE 3 de 3];

Apresentação de *Código da Questão 1:*

SUBSTITUIR ESSE TEXTO QUE ESTÁ EM VERMELHO PELO SEU CÓDIGO DO EXERCÍCIO 1.
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!).

```
class Nodo:
    def __init__(self, numero, cor):
        self.numero = numero
        self.cor = cor
        self.proximo = None

class ListaEncadeada:
    def __init__(self):
        self.head = None

    def inserirSemPrioridade(self, nodo):
        if self.head is None:
            self.head = nodo
        else:
            atual = self.head
            while atual.proximo is not None:
                atual = atual.proximo
            atual.proximo = nodo

    def inserirComPrioridade(self, nodo):
        if self.head is None or self.head.cor == 'V':
            nodo.proximo = self.head
            self.head = nodo
        else:
            atual = self.head
            while atual.proximo is not None and atual.proximo.cor == 'A':
                atual = atual.proximo
            nodo.proximo = atual.proximo
            atual.proximo = nodo

    def inserir(self):
        cor = input("Digite a cor do cartão (A ou V): ").strip().upper()
        numero = int(input("Digite o número do cartão: "))
        nodo = Nodo(numero, cor)

        if self.head is None:
            self.head = nodo
        else:
            if cor == 'V':
                self.inserirSemPrioridade(nodo)
            else:
                self.inserirComPrioridade(nodo)

    def imprimirListaEspera(self):
        atual = self.head
        while atual is not None:
            print(f"Cartão {atual.numero} - Cor {atual.cor}")
            atual = atual.proximo

    def atenderPaciente(self):
        if self.head is None:
            print("Nenhum paciente na fila.")
        else:
            paciente = self.head
            self.head = self.head.proximo
            print(f"Chamando paciente com cartão {paciente.numero} para atendimento.")

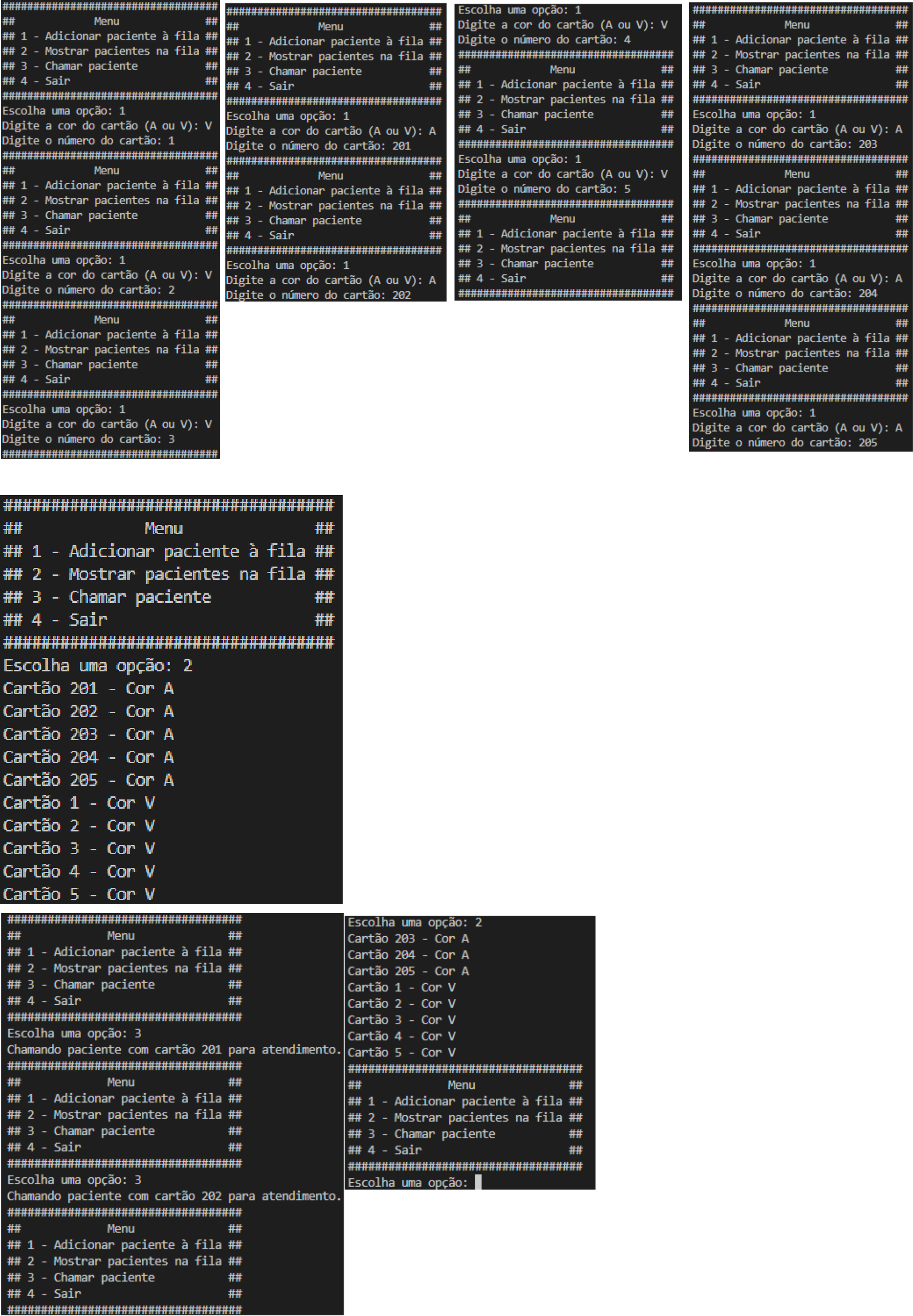
def menu():
    lista = ListaEncadeada()
    while True:
        print("#####")
        print("##          Menu          ##")
        print("## 1 - Adicionar paciente à fila ##")
        print("## 2 - Mostrar pacientes na fila ##")
        print("## 3 - Chamar paciente      ##")
        print("## 4 - Sair                 ##")
        print("#####")
        opcao = int(input("Escolha uma opção: "))

        if opcao == 1:
            lista.inserir()
        elif opcao == 2:
            lista.imprimirListaEspera()
        elif opcao == 3:
            lista.atenderPaciente()
        elif opcao == 4:
            break
        else:
            print("Opção inválida. Tente novamente.")

menu()
```


Apresentação de Saída do Console da Questão 1:

SUBSTITUIR ESSE TEXTO QUE ESTÁ EM LARANJA PELA A SAÍDA DO CONSOLE DO EXERCÍCIO 1
NÃO ESQUECER DE CUMPRIR AS EXIGÊNCIAS
SERÁ ACEITO SOMETE SAÍDAS DO CONSOLE NO FORMATO IMAGEM (NADA DE TEXTO AQUI! ZERA ESSA PARTE DA QUESTÃO!)



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:

$$posição = (CHAR1_{ASCII} + CHAR2_{ASCII}) MOD 10$$

Onde $CHAR1_{ASCII}$ e $CHAR2_{ASCII}$ são os valores ASCII da primeira e segunda letra, respectivamente (Tabela ASCII no final do documento).

Elabore um programa em Python que:

- A. Deve-se implementar a tabela Hash com 10 posições, onde inicialmente todas as posições possuem valor **None** [EXIGÊNCIA DE CÓDIGO 1 de 7];
- B. Deve-se implementar as **Listas Encadeadas Simples** em que: [EXIGÊNCIA DE CÓDIGO 2 de 7];
- a. O Nodo representa um Estado contendo: **sigla, nomeEstado** e um ponteiro para o **próximo**;
- b. As 10 posições da tabela hash, representam a cabeça de cada lista (**head**).
- C. 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];
- D. 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];
- E. Deve-se implementar a função hash, conforme enunciado. [EXIGÊNCIA DE CÓDIGO 5 de 7];
- F. 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, pode inserir no código mesmo) [EXIGÊNCIA DE CÓDIGO 6 de 7];
- G. 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):

- H. 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];
- I. 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];
- J. 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];

Caracteres de controle ASCII			ASCII caracteres imprimíveis			Caracteres ASCII estendidos										
00	NULL	(Null character)	32	space	64	@	96	`	128	Ç	160	á	192	Ł	224	Ó
01	SOH	(Start of Header)	33	!	65	A	97	a	129	ü	161	í	193	ł	225	ô
02	STX	(Start of Text)	34	"	66	B	98	b	130	é	162	ó	194	Ł	226	Ò
03	ETX	(End of Text)	35	#	67	C	99	c	131	â	163	ú	195	ł	227	Ô
04	EOT	(End of Trans.)	36	\$	68	D	100	d	132	ä	164	ñ	196	—	228	ö
05	ENQ	(Enquiry)	37	%	69	E	101	e	133	à	165	Ń	197	+	229	Ő
06	ACK	(Acknowledgement)	38	&	70	F	102	f	134	á	166	ª	198	ä	230	µ
07	BEL	(Bell)	39	'	71	G	103	g	135	ç	167	º	199	Å	231	þ
08	BS	(Backspace)	40	(72	H	104	h	136	ê	168	¿	200	Ł	232	ß
09	HT	(Horizontal Tab)	41)	73	I	105	i	137	ë	169	®	201	ƒ	233	Ù
10	LF	(Line feed)	42	*	74	J	106	j	138	è	170	¬	202	ƒ	234	Ú
11	VT	(Vertical Tab)	43	+	75	K	107	k	139	ï	171	½	203	ƒ	235	Û
12	FF	(Form feed)	44	,	76	L	108	l	140	î	172	¼	204	ƒ	236	Ý
13	CR	(Carriage return)	45	-	77	M	109	m	141	ì	173	ì	205	=	237	Ÿ
14	SO	(Shift Out)	46	.	78	N	110	n	142	Ä	174	«	206	ƒ	238	—
15	SI	(Shift In)	47	/	79	O	111	o	143	Å	175	»	207	¤	239	·
16	DLE	(Data link escape)	48	0	80	P	112	p	144	É	176	⌘	208	ø	240	≡
17	DC1	(Device control 1)	49	1	81	Q	113	q	145	æ	177	⌘	209	Ð	241	±
18	DC2	(Device control 2)	50	2	82	R	114	r	146	Æ	178	⌘	210	È	242	≡
19	DC3	(Device control 3)	51	3	83	S	115	s	147	ø	179	⌘	211	Ê	243	¼
20	DC4	(Device control 4)	52	4	84	T	116	t	148	ö	180	⌘	212	È	244	¶
21	NAK	(Negative acknowl.)	53	5	85	U	117	u	149	ò	181	À	213	ı	245	§
22	SYN	(Synchronous idle)	54	6	86	V	118	v	150	û	182	Á	214	ı	246	÷
23	ETB	(End of trans. block)	55	7	87	W	119	w	151	ù	183	Â	215	ı	247	˙
24	CAN	(Cancel)	56	8	88	X	120	x	152	ÿ	184	©	216	ı	248	°
25	EM	(End of medium)	57	9	89	Y	121	y	153	Û	185	ı	217	ı	249	˙
26	SUB	(Substitute)	58	:	90	Z	122	z	154	Ü	186	ı	218	ı	250	˙
27	ESC	(Escape)	59	;	91	[123	{	155	ø	187	ı	219	ı	251	˙
28	FS	(File separator)	60	<	92	\	124		156	£	188	ı	220	ı	252	˙
29	GS	(Group separator)	61	=	93]	125	}	157	Ø	189	ı	221	ı	253	˙
30	RS	(Record separator)	62	>	94	^	126	~	158	×	190	ı	222	ı	254	■
31	US	(Unit separator)	63	?	95	_			159	ƒ	191	ı	223	ı	255	nbsp
127	DEL	(Delete)														

EXEMPLO DE SAÍDA DE 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];

```
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 1: 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];

```
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 1: 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!).

```
class Nodo:
    def __init__(self, sigla, nomeEstado):
        self.sigla = sigla
        self.nomeEstado = nomeEstado
        self.proximo = None

class TabelaHash:
    def __init__(self):
        self.tabela = [self.criarNodoNone() for _ in range(10)]

    def criarNodoNone(self):
        return Nodo("None", "None")

    def hash(self, sigla):
        if sigla == "DF":
            return 7
        e1 = ord(sigla[0])
        e2 = ord(sigla[1])
        return (e1 + e2) % 10

    def inserir(self, sigla, nomeEstado):
        indice = self.hash(sigla)
        novo_nodo = Nodo(sigla, nomeEstado)
        atual = self.tabela[indice]
        if atual.sigla == "None":
            self.tabela[indice] = novo_nodo
            novo_nodo.proximo = self.criarNodoNone()
        else:
            while atual.proximo and atual.proximo.sigla != "None":
                atual = atual.proximo
            novo_nodo.proximo = atual.proximo
            atual.proximo = novo_nodo

    def imprimirTabela(self):
        for i in range(len(self.tabela)):
            print(f"Posição {i}: ", end="")
            atual = self.tabela[i]
            while atual is not None:
                print(f"{atual.sigla} ", end="")
                atual = atual.proximo
            print()

def menu():
    tabela_hash = TabelaHash()
    estados = {
        "AC": "Acre", "AL": "Alagoas", "AP": "Amapá", "AM": "Amazonas", "BA": "Bahia",
        "CE": "Ceará", "DF": "Distrito Federal", "ES": "Espírito Santo", "GO": "Goiás",
        "MA": "Maranhão", "MT": "Mato Grosso", "MS": "Mato Grosso do Sul", "MG": "Minas Gerais",
        "PA": "Pará", "PB": "Paraíba", "PR": "Paraná", "PE": "Pernambuco", "PI": "Piauí",
        "RJ": "Rio de Janeiro", "RN": "Rio Grande do Norte", "RS": "Rio Grande do Sul",
        "RO": "Rondônia", "RR": "Roraima", "SC": "Santa Catarina", "SP": "São Paulo",
        "SE": "Sergipe", "TO": "Tocantins", "MN": "Matheus Nakade"
    }

    while True:

        print("#####")
        print("##          Menu          ##")
        print("##1 - Mostrar tabela hash  ##")
        print("##2 - Inserir estado       ##")
        print("##3 - Sair                 ##")
        print("#####")
        opcao = int(input("Escolha uma opção: "))

        if opcao == 1:
            tabela_hash.imprimirTabela()
        elif opcao == 2:
            sigla = input("Digite a sigla do estado (2 letras): ").upper()
            if sigla in estados:
                nomeEstado = estados[sigla]
                tabela_hash.inserir(sigla, nomeEstado)
                print(f"Estado {nomeEstado} ({sigla}) inserido na tabela hash.")
            else:
                print("Sigla inválida. Tente novamente.")
        elif opcao == 3:
            break
        else:
            print("Opção inválida. Tente novamente.")

menu()
```


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!)

```
#####
##          Menu          ##
##1 - Mostrar tabela hash  ##
##2 - Inserir estado      ##
##3 - Sair                 ##
#####
Escolha uma opção: 1
Posição 0: None
Posição 1: None
Posição 2: None
Posição 3: None
Posição 4: None
Posição 5: None
Posição 6: None
Posição 7: None
Posição 8: None
Posição 9: None
```

```
#####
##          Menu          ##
##1 - Mostrar tabela hash  ##
##2 - Inserir estado      ##
##3 - Sair                 ##
#####
Escolha uma opção: 1
Posição 0: GO MS RN SC None
Posição 1: AL BA MT RO None
Posição 2: AC AM ES MA PR SE None
Posição 3: PI SP TO None
Posição 4: RR None
Posição 5: AP PA RS None
Posição 6: CE PB RJ None
Posição 7: DF None
Posição 8: MG None
Posição 9: PE None
```

```
#####
##          Menu          ##
##1 - Mostrar tabela hash  ##
##2 - Inserir estado      ##
##3 - Sair                 ##
#####
Escolha uma opção: 1
Posição 0: GO MS RN SC None
Posição 1: AL BA MT RO None
Posição 2: AC AM ES MA PR SE None
Posição 3: PI SP TO None
Posição 4: RR None
Posição 5: AP PA RS MN None
Posição 6: CE PB RJ None
Posição 7: DF None
Posição 8: MG None
Posição 9: PE None
```