



**CENTRO UNIVERSITÁRIO INTERNACIONAL UNINTER**  
**ESCOLA SUPERIOR POLITÉCNICA**  
**ENGENHARIA DE COMPUTAÇÃO**  
**ESTRUTURA DE DADOS**

**ATIVIDADE PRÁTICA**

**CARLOS VINÍCIUS DE JESUS SANTOS – RU: 3708623**  
**VINICIUS POZZOBON BORIN**

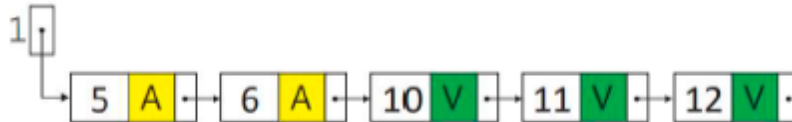
**ITABAIANA – SE**  
**2023**

## 1 EXERCÍCIO 1

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, o software desenvolvido tem como base o seguinte trecho de código-fonte:

```
1 class ElementoDaListaSimples:
2     def __init__(self, dado, cor):
3         self.dado = dado
4         self.cor = cor
5         self.proximo = None
6
7 class ListaEncadeadaSimples:
8     def __init__(self, nodos=None):
9         self.head = None
10        if nodos is not None:
11            nodo = ElementoDaListaSimples(dado=nodos.pop(0))
12            self.head = nodo
13            for elem in nodos:
14                nodo.proximo = ElementoDaListaSimples(dado=elem)
15                nodo = nodo.proximo
16
17    def inserirNoFinal(self, nodo):
18        if self.head is None:
19            self.head = nodo
20            return
21        nodo_atual = self.head
22        while nodo_atual.proximo != None:
23            nodo_atual = nodo_atual.proximo
24        nodo_atual.proximo = nodo
25        return
26
27    def inserir(self, dado, cor):
28        nodo = ElementoDaListaSimples(dado, cor)
29        if self.head is None:
30            self.head = nodo
31            return
32        else:
33            if nodo.cor == "V":
34                self.inserirNoFinal(nodo)
35            else:
36                self.inserirPrioridade(nodo)
37        return
38
```

Na linha 27, a função `inserir` recebe o número e a cor do cartão entregue ao paciente na triagem. Pacientes com cartão verde são inseridos no final da fila pela função `inserirNoFinal` (linhas 17-25). Pacientes com cartão amarelo têm prioridade no atendimento e são inseridos no início da fila, em ordem de chegada, pela função `inserirPrioridade`. Portanto, se são entregues os cartões 10-V, 11-V, 5-A, 12-V e 6-A, nessa ordem, a fila deve ficar assim organizada:



Considerando o processo de triagem descrito e o trecho de código-fonte apresentado, implemente a função `inserirPrioridade` conforme indicado (linha 36).

#### Solução do aluno:

```
class ElementoDaListaSimples:
    def __init__(self, dado, cor):
        self.dado = dado
        self.cor = cor
        self.proximo = None

class ListaEncadeadaSimples:
    def __init__(self, nodos=None):
        self.head = None
        if nodos is not None:
            # Cria o primeiro nó da lista a partir dos dados fornecidos
            nodo = ElementoDaListaSimples(dado=nodos.pop(0), cor="")
            self.head = nodo
            # Adiciona os demais nodos à lista, se houver mais dados
            for elem in nodos:
                nodo.proximo = ElementoDaListaSimples(dado=elem, cor="")
                nodo = nodo.proximo

    def inserirNoFinal(self, nodo):
        # Insere um nó no final da lista
        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

    def inserirPrioridade(self, nodo):
        # Insere um nó na posição correta para manter a ordem crescente
        if self.head is None or nodo.dado <= self.head.dado:
            # Inserção no início da lista
            nodo.proximo = self.head
```

```

        self.head = nodo
        return

    atual = self.head
    while atual.proximo is not None and atual.proximo.dado < nodo.dado:
        atual = atual.proximo

    nodo.proximo = atual.proximo
    atual.proximo = nodo

def inserir(self, dado, cor):
    # Insere um novo nó na lista com base na cor
    nodo = ElementoDaListaSimples(dado, cor)
    if self.head is None:
        self.head = nodo
        return
    else:
        if nodo.cor == "V":
            self.inserirNoFinal(nodo)
        else:
            self.inserirPrioridade(nodo)
        return

def imprimir_lista(self):
    # Imprime os elementos da lista
    print('Carlos Vinícius de Jesus Santos')

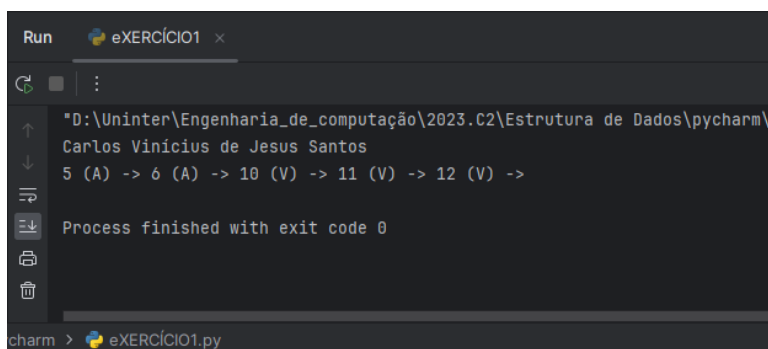
    atual = self.head
    while atual:
        print(f"{atual.dado} ({atual.cor})", end=" -> ")
        atual = atual.proximo
    print()

# Exemplo de uso
lista = ListaEncadeadaSimples()
lista.inserir(10, "V") #Inserir paciente com cartão verde no final
lista.inserir(11, "V") #Inserir paciente com cartão verde no final
lista.inserir(5, "A") #Inserir paciente com cartão amarelo no inicio
lista.inserir(12, "V") #Inserir paciente com cartão verde no final
lista.inserir(6, "A") #Inserir paciente com cartão amarelo no inicio

lista.imprimir_lista()

```

Imagem do código funcionando no seu computador:



```

Run eXERCÍCIO1 x
"D:\Uninter\Engenharia_de_computação\2023.C2\Estrutura de Dados\pycharm\
Carlos Vinícius de Jesus Santos
5 (A) -> 6 (A) -> 10 (V) -> 11 (V) -> 12 (V) ->
Process finished with exit code 0
charm > eXERCÍCIO1.py

```

## 2 EXERCÍCIO 2

Uma empresa trabalha na produção de concreto e terceiriza o serviço de transporte do produto. Os caminhoneiros telefonam para a empresa e registram seu interesse pelo trabalho. Todas as manhãs os caminhoneiros estacionam o caminhão no pátio da empresa e aguarda a sua vez. O atendimento segue o critério de ordem de chegada. Esse processo é, atualmente, controlado pela secretária, que utiliza sua agenda para gerenciar os motoristas diariamente. A empresa, que carrega, no máximo 10 caminhões por dia, pretende informatizar esse processo.

Implemente a solução desse problema, utilizando o conceito de fila.

Solução do aluno:

```
#Criação da lista de python (array) vazio e informando o tamanho máximo de
#10 unidades.
fila = []
tam = 10

# Laço infinito que permanece enquanto o(a) operador(a) não informar que
#quer sair.
while True:
    print('\n1 - Inserir caminhão na fila')
    print('2 - Despachar caminhão carregado')
    print('3 - Listar caminhões da fila')
    print('4 - Sair\n')

    op = input("Escolha uma opção: ")
    if op == '1':
        dado = input('Qual a placa do caminhão: ')
        dado = dado.upper()
        #Enquanto a fila de caminhões for menor a 10 unidades, poderá inserir.
        if len(fila) < 10:
            fila.append(dado)
        # Quando a fila for igual a 10 caminhões, não poderá mais inserir.
        else:
            print('\nFila cheia! Aguarde a saída de um caminhão.\n')
    elif op == '2':
        # Se a fila não estiver vazia, poderá ser retirado o elemento
        #localizado na posição 0, ou seja, o primeiro da fila.
        if len(fila) > 0:
            fila.pop(0)
        else:
            print('Fila vazia! Impossível remover. ')
    elif op == '3':
        for item in fila:
            print(item, end=' | ')
        print('\n')
    elif op == '4':
        print('Encerrando...')
        break
    else:
        print("\nOpção invalida!")
        print("Selecione outra opção! \n")
```

Imagem do código funcionando no seu computador:

```
Selecione outra opção!
```

- 1 - Inserir caminhão na fila
- 2 - Despachar caminhão carregado
- 3 - Listar caminhões da fila
- 4 - Sair

```
Escolha uma opção: 1
```

```
Qual a placa do caminhão: ght5678
```

```
Fila cheia! Aguarde a saída de um caminhão.
```

- 1 - Inserir caminhão na fila
- 2 - Despachar caminhão carregado
- 3 - Listar caminhões da fila
- 4 - Sair

```
Escolha uma opção: 3
```

```
ABC1234 | DEF5678 | GHI9012 | MN07890 | PQR2345 | STU6789 | VWX1234 | EF63456 | BCD9012 | GHJ3456 |
```