

**primeiro código usado:**

class PersonalArray:

SIZE = 5

insertPosition = 0

elements = [None] \* SIZE

# Função que serve para definir se o array está vazio ou não

def isEmpty(self):

return self.size() == 0

# Função que serve para retornar o número de elementos já armazenados no array

def size(self):

return self.insertPosition

# Função que serve para definir se precisamos de mais memória

def isMemoryFull(self):

return self.insertPosition == len(self.elements)

# Função para inserir na lista

def append(self, newElement):

if self.isMemoryFull():

self.updateMemory()

self.elements[self.insertPosition] = newElement

self.insertPosition += 1

# Função para aumentar a memória quando necessário

def updateMemory(self):

newArray = [None] \* (self.size() + self.SIZE)

for position in range(self.insertPosition):

newArray[position] = self.elements[position]

self.elements = newArray

newArray = None

# Função para limpar as posições

def clear(self):

for position in range(len(self.elements)):

self.elements[position] = None

self.elements = [None] \* self.SIZE

self.insertPosition = 0

# Função para remover o último elemento da lista "array"

def remove(self):

# Remove o último elemento usando o índice -1

self.elements[self.insertPosition - 1] = None

self.insertPosition -= 1 # Decrementa a posição de inserção

def remove\_at(self, position):

if not self.isEmpty() and 0 <= position < self.insertPosition:

# Reorganiza os elementos após a posição removida

for i in range(position, self.insertPosition - 1):

self.elements[i] = self.elements[i + 1]

self.elements[self.insertPosition - 1] = None

self.insertPosition -= 1 # Atualiza a posição de inserção

array = PersonalArray()

array.append("fusca")

array.append("kombi")

array.append("kwid")

array.append("Ferrari")

array.append("Hilux")

array.append("fusca")

array.append("kombi")

array.append("kwid")

array.append("Ferrari")

array.append("Hilux")

array.append("fusca")

array.append("kombi")

array.append("kwid")

array.append("Ferrari")

array.append("Hilux")

print("\nAntes de remover:")

print("Elementos:", array.elements)

print("Tamanho:", array.size())

#array.remove() #metodo para remover o ultimo da lista

#array.remove\_at(5) #metodo para remover uma posição específica

print("\nDepois de remover:")

print("Elementos:", array.elements)

print("Tamanho:", array.size())

print("\nCom o método clear:\n")

array.clear()

print("Elementos:", array.elements)

print("Tamanho:", array.size())

**codigo usado com a função de adicionar agora:**

class PersonalArray:

SIZE = 5

insertPosition = 0

elements = [None] \* SIZE

# Função que serve para definir se o array está vazio ou não

def isEmpty(self):

return self.size() == 0

# Função que serve para retornar o número de elementos já armazenados no array

def size(self):

return self.insertPosition

# Função que serve para definir se precisamos de mais memória

def isMemoryFull(self):

return self.insertPosition == len(self.elements)

# Função para inserir na lista

def append(self, newElement):

if self.isMemoryFull():

self.updateMemory()

self.elements[self.insertPosition] = newElement

self.insertPosition += 1

# Função para aumentar a memória quando necessário

def updateMemory(self):

newArray = [None] \* (self.size() + self.SIZE)

for position in range(self.insertPosition):

newArray[position] = self.elements[position]

self.elements = newArray

newArray = None

# Função para limpar as posições

def clear(self):

for position in range(len(self.elements)):

self.elements[position] = None

self.elements = [None] \* self.SIZE

self.insertPosition = 0

# Função para remover o último elemento da lista "array"

def remove(self):

# Remove o último elemento usando o índice -1

self.elements[self.insertPosition - 1] = None

self.insertPosition -= 1 # Decrementa a posição de inserção

def remove\_at(self, position):

if not self.isEmpty() and 0 <= position < self.insertPosition:

# Reorganiza os elementos após a posição removida

for i in range(position, self.insertPosition - 1):

self.elements[i] = self.elements[i + 1]

self.elements[self.insertPosition - 1] = None

self.insertPosition -= 1 # Atualiza a posição de inserção

# Função para inserir um elemento em uma posição específica

def insert\_at(self, position, newElement):

if position < 0 or position > self.insertPosition:

raise IndexError("Posição inválida para inserção.")

# Verifica se a memória está cheia e aumenta se necessário

if self.isMemoryFull():

self.updateMemory()

# Desloca os elementos para a direita para abrir espaço

for i in range(self.insertPosition, position, -1):

self.elements[i] = self.elements[i - 1]

# Insere o novo elemento na posição desejada

self.elements[position] = newElement

self.insertPosition += 1 # Atualiza a posição de inserção

array = PersonalArray()

array.append("fusca")

array.append("kombi")

array.append("kwid")

array.append("Ferrari")

array.append("Ferrari")

array.append("Hilux")

array.append("fusca")

array.append("kombi")

array.append("kwid")

array.append("Ferrari")

array.append("Hilux")

print("\n Lista antes:")

print("Elementos:", array.elements)

print("Tamanho:", array.size())

array.insert\_at(0, "UnoDaFirma")

array.insert\_at(1, "CarroDoOvo")

array.insert\_at(3, "jeep1")

array.insert\_at(3, "jeep2")

array.insert\_at(3, "jeep4")

array.insert\_at(3, "jeep5")

array.insert\_at(3, "jeep123")

array.insert\_at(2, "jeep3")

array.insert\_at(3, "jeep")

array.insert\_at(9, "jeep65")

array.insert\_at(21, "jeepIltimos")

array.insert\_at(3, "jeep235")

array.insert\_at(1, "jeep12")

array.insert\_at(3, "jeep")

print("\n Lista Agora com elementos adicionados")

print("Elementos:", array.elements)

print("Tamanho:", array.size())

#array.remove() #metodo para remover o ultimo da lista

#array.remove\_at(1) #metodo para remover uma posição específica

#print("\nDepois de remover:")

#print("Elementos:", array.elements)

#print("Tamanho:", array.size())

#print("\nCom o método clear:\n")

#array.clear()

#print("Elementos:", array.elements)

#print("Tamanho:", array.size())