class No:

def \_\_init\_\_(self, key, dir, esq):

self.item = key

self.dir = dir

self.esq = esq

class Tree:

def \_\_init\_\_(self):

self.root = No(None,None,None)

self.root = None

def inserir(self, v):

novo = No(v,None,None) # cria um novo Nó

if self.root == None:

self.root = novo

else: # se nao for a raiz

atual = self.root

while True:

anterior = atual

if v <= atual.item: # ir para esquerda

atual = atual.esq

if atual == None:

anterior.esq = novo

return

# fim da condição ir a esquerda

else: # ir para direita

atual = atual.dir

if atual == None:

anterior.dir = novo

return

# fim da condição ir a direita

def buscar(self, chave):

if self.root == None:

return None # se arvore vazia

atual = self.root # começa a procurar desde raiz

while atual.item != chave: # enquanto nao encontrou

if chave < atual.item:

atual = atual.esq # caminha para esquerda

else:

atual = atual.dir # caminha para direita

if atual == None:

return None # encontrou uma folha -> sai

return atual # terminou o laço while e chegou aqui é pq encontrou item

# O sucessor é o Nó mais a esquerda da subarvore a direita do No que foi passado como parametro do metodo

def nosucessor(self, apaga): # O parametro é a referencia para o No que deseja-se apagar

paidosucessor = apaga

sucessor = apaga

atual = apaga.dir # vai para a subarvore a direita

while atual != None: # enquanto nao chegar no Nó mais a esquerda

paidosucessor = sucessor

sucessor = atual

atual = atual.esq # caminha para a esquerda

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# quando sair do while "sucessor" será o Nó mais a esquerda da subarvore a direita

# "paidosucessor" será o o pai de sucessor e "apaga" o Nó que deverá ser eliminado

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if sucessor != apaga.dir: # se sucessor nao é o filho a direita do Nó que deverá ser eliminado

paidosucessor.esq = sucessor.dir # pai herda os filhos do sucessor que sempre serão a direita

# lembrando que o sucessor nunca poderá ter filhos a esquerda, pois, ele sempre será o

# Nó mais a esquerda da subarvore a direita do Nó apaga.

# lembrando também que sucessor sempre será o filho a esquerda do pai

sucessor.dir = apaga.dir # guardando a referencia a direita do sucessor para

# quando ele assumir a posição correta na arvore

return sucessor