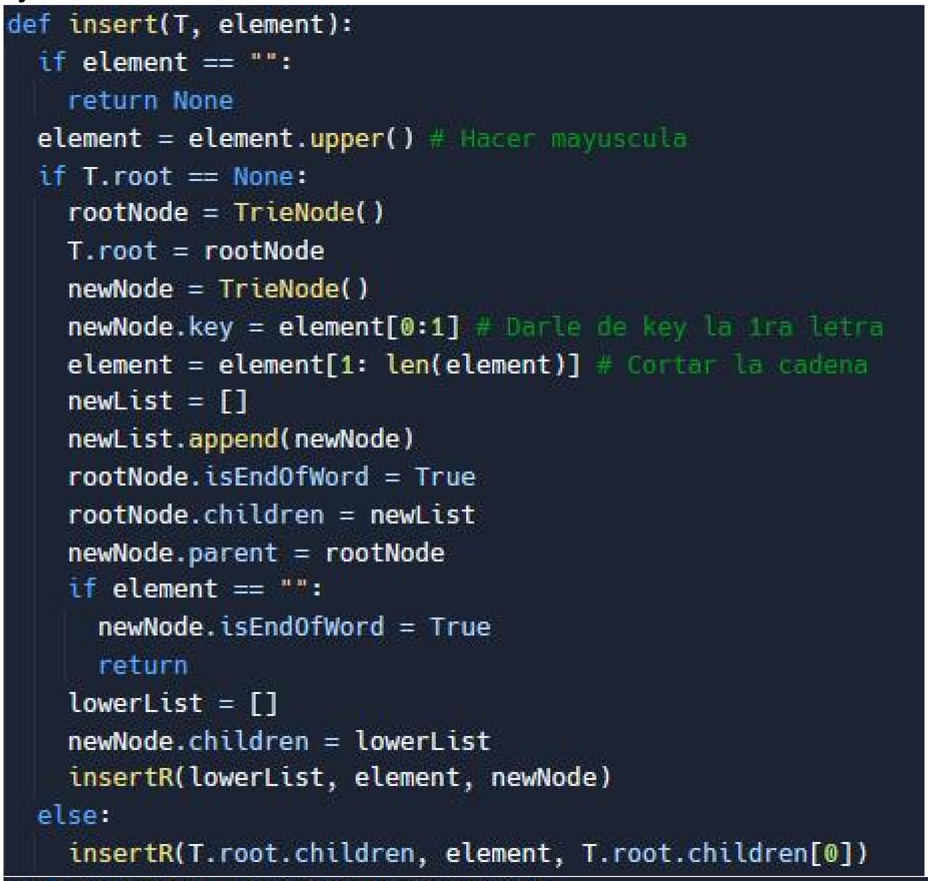
Martins Ezequiel 14000

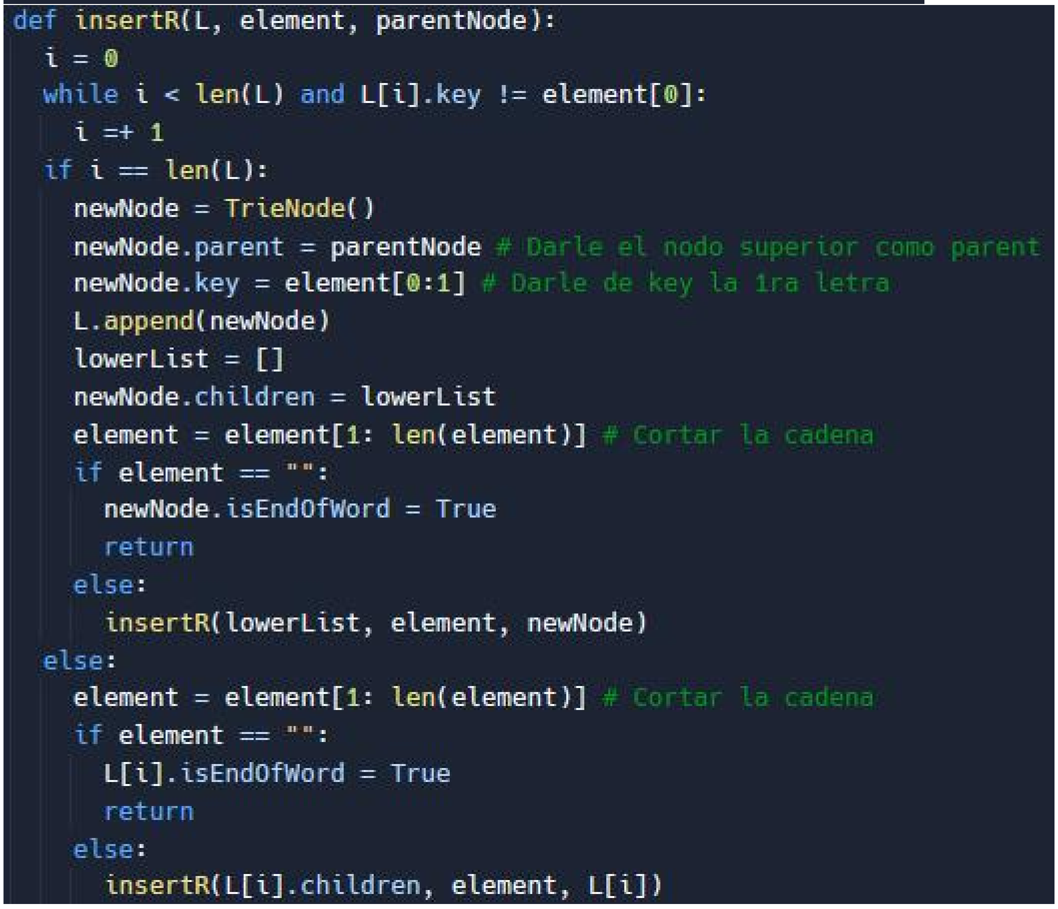
Trabajo Práctico Arboles AVL (U2)

https://replit.com/@EzeMarts/tp-trie#trie.py

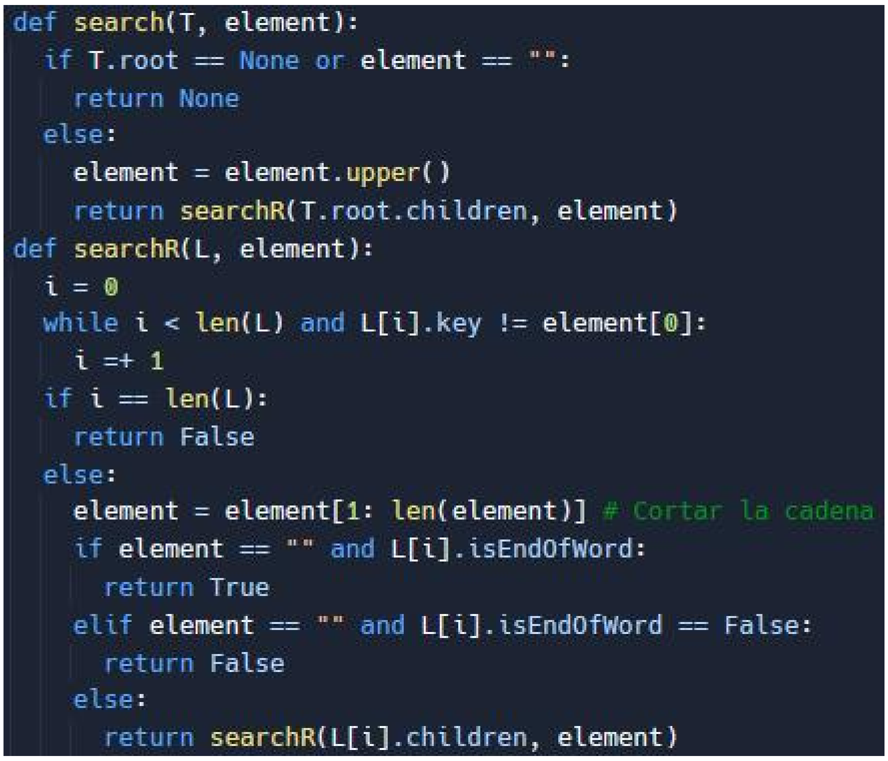
Ejercicio 1



```python
def insert(T, element):
  if element == "":
    return None
  element = element.upper() # Hacer mayuscula
  if T.root == None:
    rootNode = TrieNode()
    T.root = rootNode
    newNode = TrieNode()
    newNode.key = element[0:1] # Darle de key la 1ra letra
    element = element[1: len(element)] # Cortar la cadena
    newList = []
    newList.append(newNode)
    rootNode.isEndOfWord = True
    rootNode.children = newList
    newNode.parent = rootNode
    if element == "":
      newNode.isEndOfWord = True
      return
    lowerList = []
    newNode.children = lowerList
    insertR(lowerList, element, newNode)
  else:
    insertR(T.root.children, element, T.root.children[0])
```



```python
def insertR(L, element, parentNode):
  i = 0
  while i < len(L) and L[i].key != element[0]:
    i =+ 1
  if i == len(L):
    newNode = TrieNode()
    newNode.parent = parentNode # Darle el nodo superior como parent
    newNode.key = element[0:1] # Darle de key la 1ra letra
    L.append(newNode)
    lowerList = []
    newNode.children = lowerList
    element = element[1: len(element)] # Cortar la cadena
    if element == "":
      newNode.isEndOfWord = True
      return
    else:
      insertR(lowerList, element, newNode)
  else:
    element = element[1: len(element)] # Cortar la cadena
    if element == "":
      L[i].isEndOfWord = True
      return
    else:
      insertR(L[i].children, element, L[i])
```



```python
def search(T, element):
  if T.root == None or element == "":
    return None
  else:
    element = element.upper()
    return searchR(T.root.children, element)
def searchR(L, element):
  i = 0
  while i < len(L) and L[i].key != element[0]:
    i =+ 1
  if i == len(L):
    return False
  else:
    element = element[1: len(element)] # Cortar la cadena
    if element == "" and L[i].isEndOfWord:
      return True
    elif element == "" and L[i].isEndOfWord == False:
      return False
    else:
      return searchR(L[i].children, element)
```

Ejercicio 2

Ejercicio 3

Ejercicio 4

Ejercicio 5

Ejercicio 6

Ejercicio 7