Lab#2, NLP@CGU Spring 2023

This is due on 2023/03/13 15:30, commit to your github as a PDF (lab2.pdf) (File>Print>Save as PDF).

IMPORTANT: After copying this notebook to your Google Drive, please paste a link to it below. To get a publicly-accessible link, hit the *Share* button at the top right, then click "Get shareable link" and copy over the result. If you fail to do this, you will receive no credit for this lab!

LINK: paste your link here

https://colab.research.google.com/drive/1eVR9fQ8S6XyVi55lLpaqCFfJm-wivvOw?usp=sharing

---

Student ID: B0928024

Name: 莊靜修

＋ 程式碼 ＋ 文字

Question 1 (100 points)

Implementing Trie in Python.

Trie is a very useful data structure. It is commonly used to represent a dictionary for looking up words in a vocabulary.

For example, consider the task of implementing a search bar with auto-completion or query suggestion. When the user enters a query, the search bar will automatically suggests common queries starting with the characters input by the user.

按兩下 (或按 Enter 鍵) 即可編輯

```
# YOUR CODE HERE!
# IMPLEMENTIG TRIE IN PYTHON

class TrieNode:

    def __init__(self, char):
        self.char = char
        self.childs = []
        self.finished = False
        self.counter = 0

class Trie(object):

    def __init__(self):
        self.root = TrieNode("")

    def insert(self, word: str):
      node = self.root
      for c in word:
        founded = False
```

```
        founded = False
        for child in node.childs:
          if child.char == c:
            founded = True

            node = child
            break;
        if not founded:
          new_node = TrieNode(c)
          node.childs.append(new_node)
          node = new_node
      node.counter += 1
      node.finished = True

    def dfs(self, node, prefix):
      ans = []
      if (node.counter != 0):
            ans.append((prefix, node.counter))

      for child in node.childs:
          ans += self.dfs(child, prefix + child.char)

      return ans

    def query(self, x):
      node = self.root
      temp = ""

      for c in x:
          for n in node.childs:
              if (n.char == c):
                  node = n

      ans = self.dfs(node, x)
      ans = sorted(ans, key = lambda x: x[1], reverse = True)

      return ans



# # DO NOT MODIFY THE VARIABLES
obj = Trie()
obj.insert("長庚資工")
obj.insert("長大")
obj.insert("長庚")
obj.insert("長庚")
obj.insert("長庚大學")
obj.insert("長庚科技大學")

# # DO NOT MODIFY THE BELOW LINE!
# # THE RESULTS : [(words, count), (words, count)]
print(obj.query("長"))
```

```
# [('長庚', 2), ('長庚資工', 1), ('長庚大學', 1), ('長庚科技大學', 1), ('長大', 1)]

print(obj.query("長庚"))
# [('長庚', 2), ('長庚資工', 1), ('長庚大學', 1), ('長庚科技大學', 1)]
```

```
    [('長庚', 2), ('長庚資工', 1), ('長庚大學', 1), ('長庚科技大學', 1), ('長大', 1)]
    [('長庚', 2), ('長庚資工', 1), ('長庚大學', 1), ('長庚科技大學', 1)]
```