Group Anagrams using Python

Anagrams are words formed by rearranging the letters of another word, For example, car and arc, cat and act, etc. Grouping anagrams is one of the popular questions in coding interviews. So if you want to learn how to solve the problem of grouping anagrams, this article is for you. In this article, I will take you through a tutorial on how to group anagrams using Python.

Validate Anagrams using Python

An Anagram is a word or phrase that forms a different word or phrase when the letters of a word are rearranged. For example, the words “despair” and “praised” are anagrams. In this article, I’ll walk you through how to validate anagrams using Python.

The validation of anagram words is one of the favourite questions in coding interviews. The idea is to write an algorithm to check if the input word creates a meaningful word when rearranged. So to validate an anagram using Python, we need to input two words and check if word1 in any case matches word2 after rearranging the words.

For example, the words “cinema” and “Iceman”, let’s say that the word1 here is “cinema”, so we need to write an algorithm to check whether we can make the word “Iceman” after rearranging the letters of the word “cinema”. Below is how we can validate anagrams using Python:

In [1]:

```
def anagram(word1, word2):
    word1 = word1.lower()
    word2 = word2.lower()
    return sorted(word1) == sorted(word2)

print(anagram("cinema", "iceman"))
print(anagram("cool", "loco"))
print(anagram("men", "women"))
```

```
True
True
False
```

In the above code, I started with writing a Python function as “anagram” which includes two parameters(word1, word2). Now while initializing the words, I converted them to lower case then I am checking if the word1 equals the word2 after sorting both the words.

In [ ]:

In [ ]:

In [ ]:

Group Anagrams using Python

Grouping anagrams is one of the popular questions in coding interviews. Here you will be given a list of words, and you have to write an algorithm to group all the words which are anagrams of each other. So below is how you can write a Python function to group anagrams:

In [2]:

```
from collections import defaultdict

def group_anagrams(a):
    dfdict = defaultdict(list)
    for i in a:
        sorted_i = " ".join(sorted(i))
        dfdict[sorted_i].append(i)
    return dfdict.values()
```

Now let’s test the function by creating a list of words containing anagrams and some other words:

In [3]:

```
words = ["tea", "eat", "bat", "ate", "arc", "car"]
print(group_anagrams(words))
```

```
dict_values([['tea', 'eat', 'ate'], ['bat'], ['arc', 'car']])
```

So this is how you can group anagrams using the Python programming language. It’s valuable to practice coding interviews to improve your programming logic and coding skills. You can practice many more coding interview questions and Python projects from here.

In [ ]:

Summary

Grouping anagrams is one of the popular questions in coding interviews. Here you have to write an algorithm to group all the words which are anagrams of each other. I hope you liked this article on grouping anagrams using Python. Feel free to ask valuable questions in the comments section below.

In [ ]: