Exercises

1. Write a Python program to find the common characters in two given strings.
2. Write a Python program to check if two given strings are anagrams of each other.
3. Write a Python program to find the second most frequent character in a given string.
4. Write a Python program to find the first non-repeating character in a given string.
5. Write a Python program to find the length of the longest substring without repeating characters in a given string.
6. Write a Python program to count the number of vowels in a given string.
7. Write a Python program to count the number of consonants in a given string.
8. Write a Python program to remove all the duplicate characters from a given string.
9. Write a Python program to find the smallest window in a given string containing all characters of another string.
10. Write a Python program to check if a given string contains only digits.

Exercises and Solution

1. Write a Python program to find the common characters in two given strings.

```
string1 = "hello"
string2 = "world"
common_chars = set(string1) & set(string2)
print(f"The common characters are {', '.join(common_chars)}")
```

2. Write a Python program to check if two given strings are anagrams of each other.

```
string1 = "silent"
string2 = "listen"
if sorted(string1) == sorted(string2):
    print("The strings are anagrams")
else:
    print("The strings are not anagrams")
```

3. Write a Python program to find the second most frequent character in a given string.

```
string = "hello world"
char_count = {}
for char in string:
    if char in char_count:
        char_count[char] += 1
    else:
        char_count[char] = 1
sorted_char_count = sorted(char_count.items(), key=lambda x: x[1], reverse=True)
second_most_frequent_char = sorted_char_count[1][0]
print(f"The second most frequent character is '{second_most_frequent_char}'")
```

4. Write a Python program to find the first non-repeating character in a given string.

```
string = "hello world"
char_count = {}
for char in string:
    if char in char_count:
        char_count[char] += 1
    else:
        char_count[char] = 1
for char in string:
    if char_count[char] == 1:
        print(f"The first non-repeating character is '{char}'")
        break
```

5. Write a Python program to find the length of the longest substring without repeating characters in a given string.

```
string = "abcabcbb"
max_length = 0
start = 0
char_index_map = {}
for i in range(len(string)):
    if string[i] in char_index_map and start <= char_index_map[string[i]]:
```

```
        start = char_index_map[string[i]] + 1
    else:
        max_length = max(max_length, i - start + 1)
    char_index_map[string[i]] = i
print(f"The length of the longest substring without repeating characters is {max_length}")
```

6. Write a Python program to count the number of vowels in a given string.

```
string = "hello world"
vowels = "aeiou"
vowel_count = 0
for char in string:
    if char.lower() in vowels:
        vowel_count += 1
print(f"There are {vowel_count} vowels in the string")
```

7. Write a Python program to count the number of consonants in a given string.

```
string = "hello world"
vowels = "aeiou"
consonant_count = 0
for char in string:
    if char.isalpha() and char.lower() not in vowels:
        consonant_count += 1
print(f"There are {consonant_count} consonants in the string")
```

8. Write a Python program to remove all the duplicate characters from a given string.

```
string = "hello world"
unique_chars = []
for char in string:
    if char not in unique_chars:
        unique_chars.append(char)
unique_string = "".join(unique_chars)
print(unique_string)
```

9. Write a Python program to find the smallest window in a given string containing all characters of another string.

```
string = "this is a test string"
pattern = "tist"
# initialize variables
start = 0
end = 0
min_window = None
pattern_dict = {}
for char in pattern:
    if char in pattern_dict:
        pattern_dict[char] += 1
    else:
        pattern_dict[char] = 1
count = len(pattern_dict)
# find smallest window
while end < len(string):
    if string[end] in pattern_dict:
        pattern_dict[string[end]] -= 1
        if pattern_dict[string[end]] == 0:
            count -= 1
    end += 1
    while count == 0:
        if min_window is None or end - start < len(min_window):
            min_window = string[start:end]
        if string[start] in pattern_dict:
            pattern_dict[string[start]] += 1
            if pattern_dict[string[start]] > 0:
                count += 1
        start += 1
print(f"The smallest window containing all characters of the pattern is '{min_window}'")
```

10. Write a Python program to check if a given string contains only digits.

```
sentence = "this is a test sentence for testing the word count program"

# split the sentence into words
words = sentence.split()

# create a dictionary to store word counts
word_counts = {}
for word in words:
    if word in word_counts:
        word_counts[word] += 1
    else:
        word_counts[word] = 1

# print the word counts
for word, count in word_counts.items():
    print(f"'{word}': {count}")
```