In [1]:
```
# Answer 1:

import random

# Initialize dictionaries to store the occurrences for 10,000 rolls and 10 rolls
occurrences_10000 = {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0}
occurrences_10 = {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0}

# Simulate rolling the dice 10,000 times
num_rolls_10000 = 10000
for _ in range(num_rolls_10000):
    roll_result = random.randint(1, 6)
    occurrences_10000[roll_result] += 1

# Simulate rolling the dice 10 times
num_rolls_10 = 10
for _ in range(num_rolls_10):
    roll_result = random.randint(1, 6)
    occurrences_10[roll_result] += 1

# Print the dictionaries containing the occurrences
print("Occurrences for 10,000 rolls:", occurrences_10000)
print("Occurrences for 10 rolls:", occurrences_10)
```

```
Occurrences for 10,000 rolls: {1: 1641, 2: 1743, 3: 1692, 4: 1664, 5: 1627, 6: 1633}
Occurrences for 10 rolls: {1: 2, 2: 2, 3: 0, 4: 3, 5: 2, 6: 1}
```

In [2]:
```
#Answer 2:

# Input two sentences from the user
sentence1 = input("sentence 1: ").lower()
sentence2 = input("sentence 2: ").lower()

# Initialize lists to store characters in each sentence
char_list_sentence1 = []
char_list_sentence2 = []

# Iterate through all characters in the first sentence and add them to the list
for char in sentence1:
    char_list_sentence1.append(char)

# Iterate through all characters in the second sentence and add them to the list
for char in sentence2:
    char_list_sentence2.append(char)

# Initialize a list to store common characters in the order they appear
common_chars = []

# Iterate through the characters in sentence1
for char in char_list_sentence1:
    if char in char_list_sentence2 and char not in common_chars:
        common_chars.append(char)

# If space is in common_chars, count it only once and add it to the result list as "sp
if ' ' in common_chars:
    common_chars.remove(' ')
    common_chars.append('space')
```

```
# Create a formatted list of common characters if there are common characters
if common_chars:
    common_chars_str = ', '.join(common_chars[:-1]) + f', and {common_chars[-1]}' if l
    print(f"Then the output is: {len(common_chars)}. (they are: {common_chars_str})")
else:
    print("There are no common characters.")
```

```
sentence 1: I like Python programming
sentence 2: I like java
Then the output is: 6. (they are: i, l, k, e, a, and space)
```

In [5]:

```
#Answer 3:

import re

# Input sentence and delimiters from the user
sentence = input("sentence: ")
delimiter_input = input("delimiters (separated by spaces e.g., , / ): ")

# Split the delimiter input into a list
delimiters = delimiter_input.split()

# Create a pattern for splitting using regular expressions
delimiter_pattern = '|'.join(map(re.escape, delimiters))

# Split the sentence using the delimiter pattern
parts = re.split(delimiter_pattern, sentence)

# Remove any empty strings from the result
parts = [part.strip() for part in parts if part.strip()]

# Print the result
print(parts)
```

```
sentence: I like computer programming, including Python, Java, and C/C++
delimiters (separated by spaces e.g., , / ): , /
['I like computer programming', 'including Python', 'Java', 'and C', 'C++']
```

In [4]:

```
#Answer 4:

d = {'x': 7, 'y': 2, 'a': 3, 'm': 2}

while True:
    choice = input("Please select operation: (1: sort by key, 2: sort by value)")

    if choice == '1':
        sorted_d = dict(sorted(d.items()))
        for key, value in sorted_d.items():
            print(f"{key} , {value}")
    elif choice == '2':
        sorted_d = dict(sorted(d.items(), key=lambda item: item[1]))
        for key, value in sorted_d.items():
            print(f"{key} , {value}")
    else:
        print("Invalid choice. Please select 1 or 2.")

    another = input("Do you want to perform another operation? (yes/no): ")
    if another.lower() != 'yes':
        break
```

```
Please select operation: (1: sort by key, 2: sort by value)1
a , 3
m , 2
x , 7
y , 2
Do you want to perform another operation? (yes/no): yes
Please select operation: (1: sort by key, 2: sort by value)2
y , 2
m , 2
a , 3
x , 7
Do you want to perform another operation? (yes/no): no
```

In [ ]: