In [7]:
```
"Topic:- String Based Assignment Problem"
"Question:-1"
def reverse_string(s):# using def function here
        return s[::-1] # [::-1] notation reverse the string by starting from the last character and moving backwards wi
input_string=input("enter the string")
reversed_string=reverse_string(input_string)
print(reversed_string)
```

```
enter the stringDeepak
kapeeD
```

In [13]:
```
"Question:-2"
def is_palindrome(s): # the is_palindrome function takes as input and return true if the string is a palindrome and fal
    return s==s[::-1]# it uses slicing ([::-1]) to reverse the string and then comapre it with the real ,original strin
input_string=input("enter the string")
if is_palindrome(input_string):
    print("The string is a palindrome")
else:
    print("The string is not a palindrome")
```

```
enter the stringnitin
The string is a palindrome
```

In [14]:
```
"Question:-3"
input_string=input("enter a string")#here input()fucntion is used to prompt the user to enter a string.and the user inp
uppercase_string=input_string.upper()# here we use uppercase function
print("Uppercase string",uppercase_string)
```

```
enter a sttringdeepaksingh
Uppercase string DEEPAKSINGH
```

In [15]:
```
"Question:-4"
input_string=input("Enter a string")#here input()fucntion is used to prompt the user to enter a string.and the user inp
lowercase_string=input_string.lower()# here we use lowercase function
print("Lowercase",lowercase_string)
```

```
Enter a stringDEEPAKSINGH
Lowercase deepaksingh
```

In [17]:
```
"Question:-5"
input_string=input("enter a string")
count=0
for letter in input_string:
    if letter in ('a','e','i','o','u','A','E','I','O','U'):
        count=count+1
```

```
print(count)
```

```
enter a stringApple
2
```

In [18]:
```
"Question:-6"
input_string=input("Enter a string")
count=0
for letter in input_string:
    if letter in("bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ"):
        count=count+1
print(count)
```

```
Enter a stringApple
3
```

In [22]:
```
"Question:-7"
def remove_whitespace(input_string):
    return input_string.replace(" ", "")

input_string = input("Enter the string: ")
no_whitespace_string = remove_whitespace(input_string)
print("String without white space:", no_whitespace_string)
```

```
Enter the string: Remove all whitespace from a string
String without white space: Removeallwhitespacefromastring
```

In [23]:
```
"Question:-8"
input_string = input("Enter the string ")

length = 0
for char in input_string:
    length += 1

print("Length of the string", length)
```

```
Enter the string:
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }systemmmm hang kr diya team inida ne
Length of the string: 36
```

In [24]:
```python
"Question:-9"
input_string = input("Enter the string ")
specific_word = input("Enter the specific word to check for ")

if specific_word in input_string:
    print(f"The string contains the word '{specific_word}'.")
else:
    print(f"The string does not contain the word '{specific_word}'.")
```

```
Enter the string
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }India win in semifinals #
Enter the specific word to check for #
The string contains the word '#'.
```

In [25]:
```python
"Question:10"
input_string = input("Enter the string ")
old_word = input("Enter the word to replace ")
new_word = input("Enter the replacement word ")

updated_string = input_string.replace(old_word, new_word)

print("Updated string", updated_string)
```

```
Enter the string
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }Worldcup winner england
Enter the word to replace
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }worldcup winners India in2023
Enter the replacement word Winners are inida
Updated string Worldcup winner england
```

In [1]:
```python
"Question:-11"
input_string = input("Enter the string ")
search_word = input("Enter the word to count ")
word_list = input_string.split()
count = 0
for word in word_list:
    if word == search_word:
        count += 1
print(f"The word '{search_word}' appears {count} times in the string.")
```

```
Enter the string Deepak singh
Enter the word to count e
The word 'e' appears 0 times in the string.
```

In [4]:
```python
"Question:-12"
input_string = input("Enter the string ")
search_word = input("Enter the word to find")
index = input_string.find(search_word)
if index != -1:
    print(f"The first occurrence of '{search_word}' starts at index {index}.")
""
```

```
Enter the string Deepak is king
Enter the word to findD
The first occurrence of 'D' starts at index 0.
```

In [5]:
```python
"Question:-13"
input_string=input("enter the string")
search_word=input("enter the word to find")
index=input_string.find(search_word)
if index !=-1:
    print(f"The last occurrence of'{search_word}' starts at index{index}.")
```

```
enter the stringDeepak is king
enter the word to findg
The last occurrence of'g' starts at index13.
```

In [7]:
```python
"Question:-14"
input_string=input("enter the string")
words_of_list=input_string.split()
print("list of word",words_of_list)
```

```
enter the stringsplit a string into a list of words
list of word ['split', 'a', 'string', 'into', 'a', 'list', 'of', 'words']
```

In [8]:
```python
"Question:-15"
input_list = ["Happy", "chhhath", "Novemeber"]
#input_string = input[]"Enter the string "] # here user can give a string at his own
output_string = ' '.join(input_list)# Join the list of words into a string with spaces between them
print("input)list", output_string)
```

```
Resulting string: Happy chhhath Novemeber
```

In [10]:
```python
"Question:-16"
input_string = input("Enter the string: ")
output_string = input_string.replace(' ', '_')# Replace spaces with underscores
print("Resulting string", output_string)
```

```
Enter the string:
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }iqra Parveen
Resulting string: iqra_Parveen
```

In [12]:
```
"Question:-17"
input_string = input("Enter the string: ")
specific_word = input("Enter the specific word or phrase to check for: ")
if input_string.startswith(specific_word):
    print(f"The string starts with '{specific_word}'.")
else:
    print(f"The string does not start with '{specific_word}'.")
```

```
Enter the string:
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }# computer science with python
Enter the specific word or phrase to check for: #
The string starts with '#'.
```

In [13]:
```
"Question:=18"
input_string = input("Enter the string ")
specific_word = input("Enter the specific word or phrase to check for at the end: ")
if input_string.endswith(specific_word):
    print(f"The string ends with '{specific_word}'.")
else:
    print(f"The string does not end with '{specific_word}'.")
```

```
Enter the string: Computer science with python@
Enter the specific word or phrase to check for at the end: @
The string ends with '@'.
```

In [14]:
```
"Question:-19"
input_string = input("Enter the string")
title_case_string = input_string.title()# Convert the string to title case
print("Title case string", title_case_string)
```

```
Enter the string
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }"hello world"
Title case string "Hello World"
```

In [17]:
```python
"Question :-20"
input_string = input("Enter the string ")
word_list = input_string.split()# Split the string into words
longest_word_in_string = max(word_list, key=len)# Find the longest word
print("Longest word in the string", longest_word_in_string)
```

```
Enter the string "Longest Word in string"
Longest word in the string "Longest
```

In [19]:
```python
"Question:-21"
input_string = input("Enter the string ")
word_list = input_string.split()# Split the string into words
shortest_word_in_string = min(word_list, key=len)# Find the shortest word
print("shortest word in the string", shortest_word_in_string)
```

```
Enter the string shortest word in the string
shortest word in the string in
```

In [20]:
```python
"Question:-22"
input_string = input("Enter the string")
reversed_string = input_string[::-1]# Reverse the order of characters
print("Reversed string", reversed_string)
```

```
Enter the string:PYTHON LOVE HAI
Reversed string  IAH EVOL NOHTYP
```

In [22]:
```python
"Question:-23"
input_string = input("Enter the string ")
if input_string.isalnum():#isalnum() method returns True if all characters in the string are alphanumeric and False
    print("The string is alphanumeric")
else:
    print("The string is not alphanumeric")
```

```
Enter the string The string is alphanumeric
The string is not alphanumeric
```

In [23]:
```
"Question:24"
input_string=input("enter the string")
digits_list = [char for char in input_string if char.isdigit()]# Extract all digits from the string
extracted_digits = ''.join(digits_list)# Convert the list of digits to a string
print("Extracted digits", extracted_digits)
```

```
enter the stringDecision Tree Classifier
Extracted digits
```

In [25]:
```
"Question:25"
input_string = input("Enter the string ")
alphabets_list = [char for char in input_string if char.isalpha()]# Extract all alphabets from the string
extracted_alphabets = ''.join(alphabets_list)# Convert the list of alphabets to a string
print("Extracted Alphabets", extracted_alphabets)
```

```
Enter the string " Convert the list of alphabets to a string"
Extracted Alphabets Convertthelistofalphabetstoastring
```

In [27]:
```
"Question:-26"
input_string = input("Enter the string")
count = 0
for letter in input_string:
    if letter.isupper():
        count += 1
print("Number of uppercase letters", count)
```

```
Enter the stringMACHINE learning
Number of uppercase letters 7
```

In [28]:
```
"Question:-27"
input_string=input("enter the string")
count=0
for letter in input_string:
    if letter.islower():
        count+=1
print("Number of lowercase letters",count)
```

```
enter the string
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }MAchine learning
Number of lowercase letters 13
```

In [32]:
```
"Question:-28"
input_string = input("Enter the string ")
output_string = input_string.swapcase()
print("Resulting string", output_string)
```

```
Enter the string Machine Learning
Resulting string mACHINE lEARNING
```

In [33]:
```
"Question:-29"
input_string = input("Enter the string: ")
word_to_remove = input("Enter the word to remove: ")
modified_string = input_string.replace(word_to_remove, '')# Replace the word with an empty string to remove it
print("Modified string:", modified_string)
```

```
Enter the string:
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }Machine Learning Smj Ku nhi aa rahi ha #
Enter the word to remove: #
Modified string: Machine Learning Smj Ku nhi aa rahi ha
```

In [34]:
```
"Question:-30"
email_address = input("Enter an email address ")
if '@' in email_address and '.' in email_address and email_address.index('@') < email_address.rindex('.'):
    print("The email address is valid.")
else:
    print("The email address is not valid.")
```

```
Enter an email address Dpsingh1518@gmail.com
The email address is valid.
```

In [35]:
```
"Question:-31"
email_address = input("Enter an email address ")
if '@' in email_address:
    username = email_address.split('@')[0]
    print("Username", username)
else:
    print("Invalid email address.")
```

```
Enter an email address: itsdeepaksingh00@gmail.com
Username: itsdeepaksingh00
```

In [38]:
```
"Question:-32"
email_address = input("enter an email address")
if '@' in email_address:
    domain = email_address.split('@')[1]
    print("Domain",domain)
else:
    print("Inavlid email address")
```

```
enter an email addressitsdeepaksingh00@gmail.com
Domain gmail.com
```

In [39]:
```
"Question:-33"
input_string = input("Enter the string ")
modified_string = ' '.join(input_string.split())
print("Modified string", modified_string)
```

```
Enter the string:
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }Machine learning
Modified string: Machine learning
```

In [40]:
```
"Question:-34"
input_url_string = input("Enter the URL ")
valid_schemes = ["http", "https", ]
if any(input_url_string.startswith(scheme) for scheme in valid_schemes) and "." in input_url_string:
    print("The URL is valid.")
else:
    print("The URL is not valid.")
```

```
```

```
Enter the URL: http://localhost:8889/notebooks/14th_November_String_based_practice_questions.ipynb
The URL is valid.
```

In [46]:
```
"Question:-35"
url_string = input("Enter the URL: ")
protocol_end = url_string.find("://")
if protocol_end != -1:
    protocol = url_string[:protocol_end]
    print("Protocol:", protocol)
else:
    print("Invalid URL. No protocol found.")
```

```
Enter the URL: http://localhost:8889/notebooks/14th_November_String_based_practice_questions.ipynb
Protocol: http
```

In [47]:
```
"Question:-36"
input_string = input("Enter the string: ")
char_frequency = {}
for char in input_string:
    char_frequency[char] = char_frequency.get(char, 0) + 1
print("Character frequencies:")
for char, count in char_frequency.items():
    print(f"'{char}': {count}")
```

```
Enter the string: "Pankaj sir is best teacher"
Character frequencies:
'"': 2
'P': 1
'a': 3
'n': 1
'k': 1
'j': 1
' ': 4
's': 3
'i': 2
'r': 2
'b': 1
'e': 3
't': 2
'c': 1
'h': 1
```

In [48]:
```
"Question:-37"
input_string = input("Enter the string ")
no_punctuation_string = ''.join(char for char in input_string if char.isalnum() or char.isspace())
print("String without punctuation", no_punctuation_string)
```

```
Enter the string: http://localhost:8889/notebooks/14th_November_String_based_practice_questions.ipynb
String without punctuation: httplocalhost8889notebooks14thNovemberStringbasedpracticequestionsipynb
```

In [49]:
```
"Quewstion:-38"
input_string = input("enter the string")
if all(char.isdigit() for char in input_string):
    print("The string contains only digits")
else:
    print("The string does not contain only digits")
```

```
enter the stringFull stack data science pro
The string does not contain only digits.
```

In [52]:
```
"Question:-39"
input_string = input("Enter the string")
if all(char.isalpha() for char in input_string):
    print("The string contains only alphabets.")
else:
    print("The string does not contain only alphabets.")
```

```
Enter the stringfull stack data science pro
The string does not contain only alphabets.
```

In [54]:
```
"Question:-40"
input_string = input("enter the string")
char_list =list(input_string)
print("The character",char_list)
```

```
enter the string
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }Full stack data science pro
The character ['F', 'u', 'l', 'l', ' ', 's', 't', 'a', 'c', 'k', ' ', 'd', 'a', 't', 'a', ' ', 's', 'c', 'i', 'e', 'n',
'c', 'e', ' ', 'p', 'r', 'o']
```

In [55]:
```
"Question:-41"
string1 = input("Enter the first string: ")# Input from the user
string2 = input("Enter the second string: ")
string1 = string1.replace(" ", "").lower()# Remove spaces and convert to lowercase for case-insensitive comparison
string2 = string2.replace(" ", "").lower()
if sorted(string1) == sorted(string2):
    print("The strings are anagrams.")
else:
    print("The strings are not anagrams.")
```

```
Enter the first string: Full stack data science pro
Enter the second string: India is winner of worldcup
The strings are not anagrams.
```

In [60]:
```
"Question:-42"
#plaintext = input("Enter the string to encode: ")# Input from the user
#shift = int(input("Enter the shift value: "))# Define the shift value (e.g., 3 for a basic Caesar cipher)
#def caesar_cipher(char, shift):# Function to encode a character using Caesar cipher
   # if char.isalpha():
       # start = ord('a') if char.islower() else ord('A')
       # return chr((ord(char) - start + shift) % 26 + start)"
 #   else:
  #      return char
#ciphertext = ''.join(caesar_cipher(char, shift) for char in plaintext)# Encode the plaintext using Caesar cipher
#print("Encoded string", ciphertext)


plaintext = input("Enter the string to encode ")# Input from the user
while True:# Get a valid shift value from the user
    try:
        shift = int(input("Enter the shift value "))
        break  # Exit the loop if conversion to int is successful
    except ValueError:
        print("Invalid input. Please enter a valid integer.")
def caesar_cipher(char, shift):# Function to encode a character using Caesar cipher
    if char.isalpha():
        start = ord('a') if char.islower() else ord('A')
        return chr((ord(char) - start + shift) % 26 + start)
    else:
        return char
ciphertext = ''.join(caesar_cipher(char, shift) for char in plaintext)# Encode the plaintext using Caesar cipher
print("Encoded string:", ciphertext)
```

```
Enter the string to encode
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }full stack data science pro
Enter the shift value
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }india is winner
Invalid input. Please enter a valid integer.
Enter the shift value 3
Encoded string: ixoo vwdfn gdwd vflhqfh sur
```

In [61]:

```python
"Question:-43"
ciphertext = input("Enter the string to decode: ")# Input from the user
while True:# Get a valid shift value from the user
    try:
        shift = int(input("Enter the shift value: "))
        break  # Exit the loop if conversion to int is successful
    except ValueError:
        print("Invalid input. Please enter a valid integer.")
def caesar_decipher(char, shift):# Function to decode a character using Caesar cipher
    if char.isalpha():
        start = ord('a') if char.islower() else ord('A')
        return chr((ord(char) - start - shift) % 26 + start)
    else:
        return char
decoded_text = ''.join(caesar_decipher(char, shift) for char in ciphertext)# Decode the ciphertext using Caesar cipher
print("Decoded string:", decoded_text)
```

```
Enter the string to decode: Full stack data science pro
Enter the shift value: deepak
Invalid input. Please enter a valid integer.
Enter the shift value: 1
Decoded string: Etkk rszbj czsz rbhdmbd oqn
```

In [62]:
```
"Question:-44"
input_string = input("Enter the string ")# Input from the user
cleaned_string = ''.join(char.lower() if char.isalpha() or char.isspace() else ' ' for char in input_string)# Remove pu
words = cleaned_string.split()# Split the string into words
most_frequent_word = ""# Initialize variables to store the most frequent word and its count
max_word_count = 0
for word in words:# Count the frequency of each word
    count = words.count(word)
    if count > max_word_count:
        max_word_count = count
        most_frequent_word = word
print("Most frequent word:", most_frequent_word)
```

```
Enter the string Stack data science pro
Most frequent word: stack
```

In [66]:
```
"Question:-45"
input_string = input("Enter the string ")
unique_words = []
for word in words:
    if word not in unique_words:
        unique_words.append(word)
print("Unique words", unique_words)
```

```
Enter the string Full stack @ data  science
Unique words ['full', 'stack', 'data', 'science']
```

In [67]:
```
"Question:-46"
word = "deepak"
vowels = "aeiou"
count = 0
prev_char_was_vowel = False
for char in word.lower():
    if char in vowels:
        if not prev_char_was_vowel:
            count += 1
        prev_char_was_vowel = True
    else:
        prev_char_was_vowel = False
print(f"The word '{word}' has {count} syllables.")
```

```
The word 'deepak' has 2 syllables.
```

In [71]:
```
"Question:-47"
input_string = input("Enter the string ")
```

```
special_characters = input("Enter the specific word or phrase to check for at the end")
if special_characters in input_string:
    print(f"The string contains '{special_characters}'.")
else:
    print(f"The string does not contain '{special_characters}'.")
```

```
Enter the string Full stack
Enter the specific word or phrase to check for at the end@
The string does not contain '@'.
```

In [73]:
```
"Question:-48"
input_string = input("Enter the string")
n = int(input("Enter the index of the word to remove"))
words = input_string.split()
if 1 <= n <= len(words):
    del words[n - 1]
    result_string = ' '.join(words)
    print(f"String after removing the {n}th word {result_string}")
else:
    print("Invalid value of n. Please provide a valid index.")
```

```
Enter the stringfull stack data
Enter the index of the word to remove2
String after removing the 2th word full data
```

In [74]:
```
"question:-49"
input_string = input("Enter the string")
n = int(input("Enter the position to insert the word"))
new_word = input("Enter the word to insert")
words = input_string.split()
if 0 <= n <= len(words):
    words.insert(n, new_word)
    result_string = ' '.join(words)
    print(f"String after inserting '{new_word}' at position {n}: {result_string}")
else:
    print("Invalid position. Please provide a valid index.")
```

```
Enter the string
  @media print {
    .ms-editor-squiggles-container {
      display:none !important;
    }
  }
  .ms-editor-squiggles-container {
    all: initial;
  }Full stack data
Enter the position to insert the word2
Enter the word to insertdeepak
String after inserting 'deepak' at position 2: Full stack deepak data
```

In [75]:
```
"Question:-50"
csv_string = """1,Deepak,kumar
2,ram,Siya
3,Shiv,raja
"""
csv_list = [row.split(',') for row in csv_string.split('\n') if row]
for row in csv_list:
    print(row)
```

```
['1', 'Deepak', 'kumar']
['2', 'ram', 'Siya']
['3', 'Shiv', 'raja']
```

In [ ]:
```
# LIST BASED PRACTICE PROBLEM
```

In [15]:
```
"QUESTION:-1"
input_list =list(range(1,11))
print(input_list)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[15], line 2
      1 "QUESTION:-1"
----> 2 input_list =list(range(1,11))
      3 print(input_list)

TypeError: 'list' object is not callable
```

In [14]:
```
"Question:-2"
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
count = 0
for _ in my_list:
```

```
    count += 1
print(f"The length of the list is: {count}")
```

```
The length of the list is: 10
```

In [18]:
```
"Question:-3"
input_list = [1, 2, 3, 4, 5,6,7]
input_list.append(8)
print(input_list)
```

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

In [21]:
```
"Question:-4"
input_list=[1,2,3,4,5]
input_list.insert(1,6)
print(input_list)
```

```
[1, 6, 2, 3, 4, 5]
```

In [22]:
```
"Question:5"
input_list=[1,2,3,4]
input_list.remove(1)
print(input_list)
```

```
[2, 3, 4]
```

In [24]:
```
"Question:6"
input_list=[1,2,3,4]
input_list.pop(2)
print(input_list)
```

```
[1, 2, 4]
```

In [27]:
```
"Question:-7"
input_list = [1, 2, 3, 4, 5,6]
if 5 in input_list:
    print("5 exists in the list")
else:
    print("5 does not exist in the list")
```

```
5 exists in the list
```

In [28]:
```
"Question:-8"
input_list = [1, 2, 3, 4, 5, 3, 6]
element = 5
index = input_list.index(element)
print(f"The index of the first occurrence of {element} is: {index}")
```

```
The index of the first occurrence of 5 is: 4
```

In [29]:
```python
"Question:-9"
input_list = [1, 2, 3, 4, 2, 5, 2, 6]
element = 2
count = input_list.count(element)
print(f"The element {element} occurs {count} times in the list")
```

```
The element 2 occurs 3 times in the list
```

In [32]:
```python
"Question:-10"
input_list = [1, 2, 3, 4, 5, 6]
input_list.reverse()  # Reverse the list in-place
output_list = input_list[::-1]
print("output_list", output_list)
```

```
output_list: [1, 2, 3, 4, 5, 6]
```

In [36]:
```python
"Question:-11"
input_list = [5, 2, 8, 1, 3]
sorted_list = sorted(input_list)
print("Original list", input_list)
print("Sorted list in ascending order", sorted_list)
```

```
Original list: [5, 2, 8, 1, 3]
Sorted list in ascending order: [1, 2, 3, 5, 8]
```

In [37]:
```python
"Question:-12"
input_list=[5,2,8,1,3,]
sorted_list =sorted(input_list)
print("original list",input_list)
print("sorted list in descending order",sorted_list)
```

```
original list [5, 2, 8, 1, 3]
sorted list in descending order [1, 2, 3, 5, 8]
```

In [38]:
```python
"Question:-13"
even_numbers = [num for num in range(1, 21) if num % 2 == 0]
print(even_numbers)
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

In [41]:
```python
"question:-14"
odd_numbers = [num for num in range(1, 21) if num % 2 != 0]
print(odd_numbers)
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
```

In [44]:
```
"Queston:-15"
input_list = [1, 2, 3, 4, 5]
sum_of_elements = sum(input_list)
print("Sum of all elements", sum_of_elements)
```

```
Sum of all elements: 15
```

In [49]:
```
"question:-16"
input_list = [1, 2, 3, 4, 5]
max_value = max(input_list)
print("The maximum value of the list", max_value)
```

```
The maximum value of the list 5
```

In [51]:
```
"Question:-17"
input_list=[10,20,30,40,50,60]
min_values = min(input_list)
print("The miniume value of the list ", min_values)
```

```
The miniume value of the list  10
```

In [53]:
```
"Question:-18"
squares_list = [num ** 2 for num in range(1, 11)]
print(squares_list)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

In [55]:
```
"Question:-19"
random_numbers = [int(input("Enter a number: ")) for _ in range(5)]
print(random_numbers)
```

```
Enter a number: 1
Enter a number: 2
Enter a number: 5
Enter a number: 4
Enter a number: 6
[1, 2, 5, 4, 6]
```

In [6]:
```
"Question:-20"
input_list = [1, 2, 3, 3, 4, 4, 5]
unique_list = list(set(input_list))
print(unique_list)
```

```
[1, 2, 3, 4, 5]
```

In [4]:
```
"Question:-21"
list1 = [1, 2, 3, 4, 5]
list2 = [5, 6, 7, 8, 9]

common_elements = [element for element in list1 if element in list2]
print(common_elements)
```

```
[5]
```

In [3]:
```
"Question:-22"
list1 = [10, 15, 20, 25, 30, 35, 40]
list2 = [25, 40, 35]

difference = [element for element in list1 if element not in list2]
print(difference)
```

```
[10, 15, 20, 30]
```

In [75]:
```
"Question:-23"
input_list1 = [1, 2, 3]
input_list2 = [4, 5, 6]
merged_list = input_list1 + input_list2# Merge lists using the + operator
print("Merged list", merged_list)# Print the merged list
```

```
Merged list: [1, 2, 3, 4, 5, 6]
```

In [77]:
```
"Question:-24"
input_list=[1,2,3,4]
result = [num * 2 for num in input_list]
print(" input_ List",input_list)
print("Result", result)
```

```
 input_ List [1, 2, 3, 4]
Result: [2, 4, 6, 8]
```

In [78]:
```
"Question:-25"
input_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]# Sample list of numbers
output_list = [num for num in input_list if num % 2 != 0]# Using list comprehension to filter out even numbers
print("Input List:", input_list)
print("Output_List (Odd Numbers):", output_list)
```

```
Input List: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Output_List (Odd Numbers): [1, 3, 5, 7, 9]
```

In [79]:
```
"Question:-26"
string_list = ["1", "2", "3", "4", "5"]
```

```
integer_list = [int(num) for num in string_list] #Using list comprehension to convert strings to integers
print("Original String List:", string_list)
print("Integer List:", integer_list)
```

```
Original String List: ['1', '2', '3', '4', '5']
Integer List: [1, 2, 3, 4, 5]
```

In [80]:
```
"Question:-27"
integer_list = [1, 2, 3, 4, 5]
string_list = [str(num) for num in integer_list]# Using list comprehension to convert integers to strings
print("Original Integer List:", integer_list)
print("String List:", string_list)
```

```
Original Integer List: [1, 2, 3, 4, 5]
String List: ['1', '2', '3', '4', '5']
```

In [1]:
```
"question:-28"
nested_list = [[1, 2, 3], [4, 5, 6], [7], [8, 9]]
flat_list = []
for sublist in nested_list:
    for element in sublist:
        flat_list.append(element)
print(flat_list)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

In [2]:
```
"Question:-29"
fibonacci_numbers = [0, 1]  # initialize the list with the first two Fibonacci numbers
while len(fibonacci_numbers) < 10:  # repeat until the list has 10 numbers
    next_number = fibonacci_numbers[-1] + fibonacci_numbers[-2]  # calculate the next Fibonacci number
    fibonacci_numbers.append(next_number)  # add the next Fibonacci number to the list
print(fibonacci_numbers)
```

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

In [7]:
```
"Question:-30"
input_list = [1, 2, 3, 4, 5]
is_sorted = True
for i in range(len(my_list) - 1):# here we using loop (for)
    if my_list[i] > my_list[i + 1]:
        is_sorted = False
        break
print(is_sorted)  # Output: True
my_list = [5, 4, 3, 2, 1]
is_sorted = True
for i in range(len(my_list) - 1):
```

```
    if my_list[i] > my_list[i + 1]:
        is_sorted = False
        break

print(is_sorted)  # Output: False
```

```
True
False
```

In [10]:
```
"Question:-31"
input_list = [1, 2, 3, 4, 5]
n = 3  # Number of positions to rotate left
n = n % len(input_list)# Ensure n is within the range of the list length
rotated_list = input_list[n:] + input_list[:n]# Perform left rotation without using a function or libraries
print(rotated_list)
```

```
[4, 5, 1, 2, 3]
```

In [11]:
```
"Question:-32"
input_list = [1, 2, 3, 4, 5]
n = 4  # Number of positions to rotate right
n = n % len(my_list)# Ensure n is within the range of the list length
rotated_list = my_list[-n:] + my_list[:-n]# Perform right rotation without using a function or libraries
print(rotated_list)
```

```
[4, 3, 2, 1, 5]
```

In [12]:
```
"Question:-33"
primes = []
for num in range(2, 51):
    is_prime = True
    for i in range(2, int(num ** 0.5) + 1):
        if num % i == 0:
            is_prime = False
            break
    if is_prime:
        primes.append(num)
print(primes)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
```

In [13]:
```
"Question-34"
input_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
n = 2  # Size of each chunk
chunks = [input_list[i:i + n] for i in range(0, len(input_list), n)]# Split the list into chunks of size 'n'
print(chunks)
```

```
[[1, 2], [3, 4], [5, 6], [7, 8], [9, 10]]
```

In [14]:
```
"Question:-35"
input_list = [10, 5, 8, 12, 7, 3, 15]
largest = second_largest = float('-inf')# Find the second-largest number
for num in input_list:
    if num > largest:
        second_largest = largest
        largest = num
    elif num > second_largest and num != largest:
        second_largest = num
print("Second Largest Number", second_largest)
```

```
Second Largest Number 12
```

In [15]:
```
"Question:-36"
input_list = [1, 2, 3, 4, 5]
input_list = [x ** 2 for x in input_list]# Replace every element with its square
print(input_list)
```

```
[1, 4, 9, 16, 25]
```

In [17]:
```
"Question:-37"
input_list = ['apple', 'banana', 'orange', 'grape']
input_dict = {element: index for index, element in enumerate(input_list)}# Convert the list to a dictionary
print(input_dict)
```

```
{'apple': 0, 'banana': 1, 'orange': 2, 'grape': 3}
```

In [18]:
```
"Question:-38"
input_list = [1, 2, 3, 4, 5]
length = len(input_list)# Shuffle the elements of the list randomly
for i in range(length - 1, 0, -1):# Perform swaps to shuffle the list
    j = (i * 7 + 3) % length# Generate a pseudo-random index between 0 and i (inclusive
    input_list[i], input_list[j] = input_list[j], input_list[i]# Swap the elements at indices i and j
print(input_list)
```

```
[5, 1, 3, 2, 4]
```

In [19]:
```
"Question:-39"
fibonacci_numbers = [0, 1]  # initialize the list with the first two Fibonacci numbers
while len(fibonacci_numbers) < 10:  # repeat until the list has 10 numbers
    next_number = fibonacci_numbers[-1] + fibonacci_numbers[-2]  # calculate the next Fibonacci number
    fibonacci_numbers.append(next_number)  # add the next Fibonacci number to the list
print(fibonacci_numbers)
```

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

In [20]:
```
"Question:-40"
list1 = [1, 2, 3, 4, 5]
list2 = [5, 6, 7, 8, 9]
has_common_element = any(x in list2 for x in list1)# Check if the two lists have at least one element in common
if has_common_element:
    print("The lists have at least one common element.")
else:
    print("The lists have no common elements.")
```

```
The lists have at least one common element.
```

In [23]:
```
"question:-41"
input_list = [1, 2, 3, 4, 5]
input_list = []# Remove all elements from the list
print(input_list)
```

```
[]
```

In [28]:
```
"Question:-42"
input_list = [1, -2, 3, -4, 5, -6]
input_list = [max(0, x) for x in input_list]# Replace negative numbers with 0
print(input_list)
```

```
[1, 0, 3, 0, 5, 0]
```

In [29]:
```
'Question:-43'
list_string = "apple,orange,banana,grape"
word_list = list_string.split(',')# Convert the string into a list of words using a comma as the delimiter
print(word_list)
```

```
['apple', 'orange', 'banana', 'grape']
```

In [30]:
```
"Question:-44"
word_list = ['Hello,', 'this', 'is', 'a', 'sample', 'string.']
my_string = ' '.join(word_list)# Convert the list of words into a string
print(my_string)
```

```
Hello, this is a sample string.
```

In [31]:
```
"Question:-45"
n = 10  # Change this to the desired number of powers of 2
powers_of_2 = [2 ** i for i in range(n)]# Create a list of the first n powers of 2
print(powers_of_2)
```

```
[1, 2, 4, 8, 16, 32, 64, 128, 256, 512]
```

In [1]:
```
"Question:-46"
string_list = ['apple', 'banana', 'kiwi', 'strawberry', 'blueberry']
longest_string = max(string_list, key=len)# Find the longest string in the list
print("Longest String", longest_string)
```

```
Longest String strawberry
```

In [2]:
```
"Question:-47"
input_list=['apple', 'banana', 'kiwi', 'strawberry', 'blueberry']
shortest_string = min(input_list,key=len)#find the shortest string in a list
print("shortest_string",shortest_string)
```

```
shortest_string kiwi
```

In [4]:
```
"Question:-48"
def generate_triangular_numbers(n):
    triangular_numbers = []
    for i in range(1, n + 1):
        # Using the formula for triangular numbers: T_n = n * (n + 1) / 2
        triangular_numbers.append((i * (i + 1)) // 2)
    return triangular_numbers
n = 20# Example: Generate the first 20 triangular numbers
triangular_numbers_list = generate_triangular_numbers(n)
print("The first {} triangular numbers are {}".format(n, triangular_numbers_list))
```

```
The first 20 triangular numbers are: [1, 3, 6, 10, 15, 21, 28, 36, 45, 55, 66, 78, 91, 105, 120, 136, 153, 171, 190, 21
0]
```

In [5]:
```
"Question:-49"
def is_subsequence(subsequence, sequence):
    # Find the index of each element in the subsequence in the sequence
    indices = [sequence.index(element) for element in subsequence if element in sequence]

    # Check if the indices are in increasing order
    return indices == sorted(indices)

# Example:
sequence = [1, 2, 3, 4, 5, 6, 7]
subsequence = [2, 4, 6]

if is_subsequence(subsequence, sequence):
    print("{} is a subsequence of {}".format(subsequence, sequence))
else:
    print("{} is not a subsequence of {}".format(subsequence, sequence))
```

```
[2, 4, 6] is a subsequence of [1, 2, 3, 4, 5, 6, 7]
```

In [6]:
```
"question:-50"
def swap_elements_by_indices(lst, index1, index2):
    if 0 <= index1 < len(lst) and 0 <= index2 < len(lst):  # Ensure the indices are within the bounds of the list
        temp = lst[index1] # Swap the elements using a temporary variable
        lst[index1] = lst[index2]
        lst[index2] = temp
    else:
        print("Error: Indices out of bounds.")
my_list = [1, 2, 3, 4, 5]# Example:
print("Original List:", my_list)
swap_elements_by_indices(my_list, 1, 3)# Swap elements at indices 1 and 3

print("List after swapping elements at indices 1 and 3:", my_list)
```

```
Original List: [1, 2, 3, 4, 5]
List after swapping elements at indices 1 and 3: [1, 4, 3, 2, 5]
```

In [ ]:
```
# TUPLES BASED PRACTICE PROBLEM
```

In [8]:
```
"question:-1"
my_tuples=(1,2,3,4,5)
```

In [10]:
```
"Question:-2"
my_tuples=(1,2,3,4,5)
third_element=my_tuples[2]
print(third_element)
```

```
3
```

In [11]:
```
"Question:-3"
my_tuple = (1, 2, 3, 4, 5)
length = 0# Initialize a counter variable
for _ in my_tuple:# Iterate through the tuple and count elements
    length += 1
print("Length of the tuple:", length)# Print the length
```

```
Length of the tuple: 5
```

In [12]:
```
"Question:-4"
my_tuple=(1, 2, 3, 4, 5,1)
count_occurrences=my_tuple
print(count_occurrences)
```

```
(1, 2, 3, 4, 5, 1)
```

In [14]:
```
"Question:-5"
my_tuple=(1,2,3,4,5)
index_occurrences=my_tuple.index(1)
print(index_occurrences)
```

```
0
```

In [16]:
```
"Question:-6"
my_tuple=(1,2,3,4,5)
element_exists=3
if element_exists in my_tuple:
    print("Element exists in my tuple")
else:
    print("Element is not exists in my tuples")
```

```
Element exists in my tuple
```

In [18]:
```
"Question:-7"
my_tuple=(1,2,3,4,5)
output_list=list(my_tuple)
print(output_list)
```

```
[1, 2, 3, 4, 5]
```

In [19]:
```
"Question:8"
my_list=(1,2,3,4,5)
output_tuple=tuple(my_list)
print(output_list)
```

```
[1, 2, 3, 4, 5]
```

In [20]:
```
"Question:-9"
my_tuple = (1, 2, 3, 4, 5)
var1, var2, var3, var4, var5 = my_tuple
print("Unpacked variables:", var1, var2, var3, var4, var5)
```

```
Unpacked variables: 1 2 3 4 5
```

In [21]:
```
"Question:-10"
even_numbers_tuple = tuple(x for x in range(1, 11) if x % 2 == 0)
print("Tuple of even numbers from 1 to 10", even_numbers_tuple)
```

```
Tuple of even numbers from 1 to 10 (2, 4, 6, 8, 10)
```

In [22]:
```
"Question:-11"
odd_numbers_tuple = tuple(x for x in range(1, 11) if x % 2 != 0)
print("Tuple of odd number from 1 to 10 ",odd_numbers_tuple)
```

```
Tuple of odd number from 1 to 10  (1, 3, 5, 7, 9)
```

In [2]:
```
"Question:-12"
tuple1 = (1, 2, 3)
tuple2 = (4, 5, 6)
concatenated_tuple = tuple1 + tuple2
print(concatenated_tuple)
```

```
(1, 2, 3, 4, 5, 6)
```

In [3]:
```
"Question:-13"
input_tuple =(1,2,3)
n=3
output_tuple=input_tuple*3
print(output_tuple)
```

```
(1, 2, 3, 1, 2, 3, 1, 2, 3)
```

In [4]:
```
"Question:-14"
my_tuple = ()
if not my_tuple:
    print("The tuple is empty.")
else:
    print("The tuple is not empty.")
```

```
The tuple is empty.
```

In [5]:
```
"Question:-15"
nested_tuple = ((1, 2, 3), ('a', 'b', 'c'), (True, False))
print(nested_tuple)
```

```
((1, 2, 3), ('a', 'b', 'c'), (True, False))
```

In [6]:
```
"Queston:-16"
nested_tuple = ((1, 2, 3), ('a', 'b', 'c'), (True, False))
first_element = nested_tuple[0]
print(first_element)
```

```
(1, 2, 3)
```

In [7]:
```
"Question:-17"
single_element_tuple = (7,)
```

```
print(single_element_tuple)
```

```
(7,)
```

In [8]:
```
"Question:-18"
tuple1 = (1, 2, 3)
tuple2 = (1, 2, 4)
if tuple1 == tuple2:
    print("Tuples are equal.")
else:
    print("Tuples are not equal.")
```

```
Tuples are not equal.
```

In [ ]:
```
"Question:-19"
my_tuple = (1, 2, 3, 4, 5)
del my_tuple
# we cannot delete a tuple becasue tuple are imutable and if we trying to delet it will shows error
```

In [9]:
```
"Question:-20"
my_tuple = (1, 2, 3, 4, 5)
sliced_tuple = my_tuple[1:4]# Slice from index 1 to index 4 (exclusive)
print(sliced_tuple)
```

```
(2, 3, 4)
```

In [10]:
```
"Question:-21"
my_tuple = (10, 5, 8, 15, 3)
max_value = max(my_tuple)
print(max_value)
```

```
Maximum value in the tuple 15
```

In [11]:
```
"Question:-22"
my_tuple=(10,5,8,15,3)
min_value=min(my_tuple)
print(min_value)
```

```
3
```

In [12]:
```
"Question:-23"
my_string = "Deepak_singh"
char_tuple = tuple(my_string)
print(char_tuple)
```

```
('D', 'e', 'e', 'p', 'a', 'k', '_', 's', 'i', 'n', 'g', 'h')
```

In [13]:
```
"Question:-24"
char_tuple = ('D', 'e', 'e', 'p', 'a', 'k', '_', 's', 'i', 'n', 'g', 'h')
resulting_string = ''.join(char_tuple)
print(resulting_string)
```

```
Deepak_singh
```

In [14]:
```
"Question:-25"
mixed_tuple = (1, 'Hello', 3.14, True, [5, 6, 7])
print(mixed_tuple)
```

```
(1, 'Hello', 3.14, True, [5, 6, 7])
```

In [15]:
```
"Question:-26"
tuple1 = (1, 2, 3)
tuple2 = (1, 2, 3)
if tuple1 == tuple2:
    print("Tuples are identical.")
else:
    print("Tuples are not identical.")
```

```
Tuples are identical.
```

In [16]:
```
"Question:-27"
my_tuple = (4, 1, 7, 3, 9)
sorted_tuple = tuple(sorted(my_tuple))
print(sorted_tuple)
```

```
(1, 3, 4, 7, 9)
```

In [17]:
```
"Question:-28"
tuple_of_integers = (1, 2, 3, 4, 5)
tuple_of_strings = tuple(str(i) for i in tuple_of_integers)
print(tuple_of_strings)
```

```
('1', '2', '3', '4', '5')
```

In [18]:
```
'Question:-29'
tuple_of_strings = ('10', '20', '30', '40', '50')
tuple_of_integers = tuple(int(s) for s in tuple_of_strings)
print(tuple_of_integers)
```

```
(10, 20, 30, 40, 50)
```

In [19]:
```
"Question:-30"
tuple1 = (1, 2, 3)
```

```
tuple2 = ('a', 'b', 'c')
merged_tuple = tuple1 + tuple2
print(merged_tuple)
```

```
(1, 2, 3, 'a', 'b', 'c')
```

In [20]:
```
"Question:-31"
nested_tuple = ((1, 2, 3), ('a', 'b', 'c'), (True, False))
flattened_tuple = tuple(item for sublist in nested_tuple for item in sublist)
print(flattened_tuple)
```

```
(1, 2, 3, 'a', 'b', 'c', True, False)
```

In [21]:
```
"Question:-32"
prime_numbers = (2, 3, 5, 7, 11)
print(prime_numbers)
```

```
(2, 3, 5, 7, 11)
```

In [22]:
```
"Question:-33"
def is_palindrome_tuple(input_tuple):
    return input_tuple == input_tuple[::-1]
palindrome_tuple = (1, 2, 3, 4, 3, 2, 1)
non_palindrome_tuple = (1, 2, 3, 4, 5)
print(is_palindrome_tuple(palindrome_tuple))
print(is_palindrome_tuple(non_palindrome_tuple))
```

```
True
False
```

In [23]:
```
"Question:-34"
squares_tuple = tuple(x ** 2 for x in range(1, 6))
print(squares_tuple)
```

```
(1, 4, 9, 16, 25)
```

In [24]:
```
"Question:-35"
original_tuple = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
filtered_tuple = tuple(x for x in original_tuple if x % 2 != 0)
print(filtered_tuple)
```

```
(1, 3, 5, 7, 9)
```

In [25]:
```
"Question:-36"
original_tuple = (1, 2, 3, 4, 5)
multiplied_tuple = tuple(x * 2 for x in original_tuple)
print(multiplied_tuple)
```

```
(2, 4, 6, 8, 10)
```

In [26]:
```python
"Question:-37"
import random
random_tuple = tuple(random.uniform(0, 1) for _ in range(5))
print(random_tuple)
```

```
(0.6089494821689807, 0.5124330586817425, 0.5238184465843724, 0.20481973257312047, 0.16028318992370516)
```

In [27]:
```python
"Question:-38"
my_tuple = (1, 2, 3, 4, 5)
is_sorted = my_tuple == tuple(sorted(my_tuple))
if is_sorted:
    print("The tuple is sorted.")
else:
    print("The tuple is not sorted.")
```

```
The tuple is sorted.
```

In [28]:
```python
"Question:-39"
def rotate_left_tuple(my_tuple, n):
    n = n % len(my_tuple)  # Ensure n is within the range of the tuple length
    rotated_tuple = my_tuple[n:] + my_tuple[:n]
    return rotated_tuple
original_tuple = (1, 2, 3, 4, 5)
rotated_tuple = rotate_left_tuple(original_tuple, 2)
print("Original Tuple", original_tuple)
print("Rotated Tuple", rotated_tuple)
```

```
Original Tuple: (1, 2, 3, 4, 5)
Rotated Tuple: (3, 4, 5, 1, 2)
```

In [29]:
```python
"Question:-40"
def rotate_right_tuple(my_tuple, n):
    n = n % len(my_tuple)
    return my_tuple[-n:] + my_tuple[:-n]
original_tuple = (1, 2, 3, 4, 5)
rotated_tuple = rotate_right_tuple(original_tuple, 2)

print("Original Tuple", original_tuple)
print("Rotated Tuple", rotated_tuple)
```

```
Original Tuple: (1, 2, 3, 4, 5)
Rotated Tuple: (3, 4, 5, 1, 2)
```

In [31]:
```
"Question:-41"
fibonacci_sequence = [0, 1]
while len(fibonacci_sequence) < 5:
    next_number = fibonacci_sequence[-1] + fibonacci_sequence[-2]
    fibonacci_sequence.append(next_number)
first_5_fibonacci = tuple(fibonacci_sequence)
print(first_5_fibonacci)
```

```
(0, 1, 1, 2, 3)
```

In [33]:
```
"Question:-42"
element1 = input("Enter the first element ")
element2 = input("Enter the second element ")
element3 = input("Enter the third element")
user_tuple = (element1, element2, element3)
print("User Tuple", user_tuple)
```

```
Enter the first element 1
Enter the second element 2
Enter the third element3
User Tuple ('1', '2', '3')
```

In [34]:
```
"Question:-43"
def swap_elements_in_tuple(input_tuple, index1, index2):
    tuple_list = list(input_tuple)
    tuple_list[index1], tuple_list[index2] = tuple_list[index2], tuple_list[index1]
    swapped_tuple = tuple(tuple_list)
    return swapped_tuple
my_tuple = (1, 2, 3, 4)
swapped_tuple = swap_elements_in_tuple(my_tuple, 0, 1)
print("Original Tuple", my_tuple)
print("Swapped Tuple", swapped_tuple)
```

```
Original Tuple: (1, 2, 3, 4)
Swapped Tuple: (2, 1, 3, 4)
```

In [35]:
```
"Question:-44"
def reverse_tuple(input_tuple):
    reversed_tuple = input_tuple[::-1]
    return reversed_tuple
my_tuple = (1, 2, 3, 4)
reversed_tuple = reverse_tuple(my_tuple)
print("Original Tuple", my_tuple)
print("Reversed Tuple", reversed_tuple)
```

```
Original Tuple (1, 2, 3, 4)
Reversed Tuple (4, 3, 2, 1)
```

In [36]:
```python
"Question:-45"
def powers_of_two(n):
    return tuple(2 ** i for i in range(n))
n = 5
power_of_two_tuple = powers_of_two(n)
print("Powers of 2 Tuple:", power_of_two_tuple)
```

```
Powers of 2 Tuple: (1, 2, 4, 8, 16)
```

In [37]:
```python
"Question:-46"
my_tuple = ("apple", "banana", "kiwi", "strawberry", "orange")
if not my_tuple:
    print("The tuple is empty.")
else:
    longest_string = my_tuple[0]

    for string in my_tuple:
        if len(string) > len(longest_string):
            longest_string = string
    print("Longest String", longest_string)
```

```
Longest String strawberry
```

In [38]:
```python
"Question:-47"
my_tuple = ("apple", "banana", "kiwi", "strawberry", "orange")
if not my_tuple:
    print("The tuple is empty.")
else:
    shortest_string = my_tuple[0]
    for string in my_tuple:
        if len(string) < len(shortest_string):
            shortest_string = string
    print("Shortest String", shortest_string)
```

```
Shortest String kiwi
```

In [39]:
```python
"Question:-48"
n = 10
triangular_numbers_tuple = tuple((i * (i + 1)) // 2 for i in range(1, n + 1))
print(triangular_numbers_tuple)
```

```
(1, 3, 6, 10, 15, 21, 28, 36, 45, 55)
```

In [40]:
```
"Question:-49"
main_tuple = (1, 2, 3, 4, 5, 6, 7, 8)
sub_tuple = (3, 4, 5)
n = len(sub_tuple)
is_subsequence = any(main_tuple[i:i + n] == sub_tuple for i in range(len(main_tuple) - n + 1))
print(is_subsequence)
```

```
True
```

In [41]:
```
"Question:-50"
n = 10
alternating_tuple = tuple((i % 2) for i in range(n))
print(alternating_tuple)
```

```
(0, 1, 0, 1, 0, 1, 0, 1, 0, 1)
```

In [1]:
```
#SET BASED  PRACTICE PROBLEM
"Question:-1"
my_set={1,2,3,4,5}
```

In [3]:
```
"Question:-2"
my_set={1,2,3,4} # we give the values
output_set=my_set.copy()
output_set.add(5)
print(output_set)# here we print this
```

```
{1, 2, 3, 4, 5}
```

In [5]:
```
"Question:-3"
my_set={1,2,3,4,5}# enter the values
my_set.remove(5)# here we use remove keyword here to remove a element in my set
print(my_set)# here i print this
```

```
{1, 2, 3, 4}
```

In [7]:
```
"Question:-4"
my_set={1,2,3,4}# enter the values
element_exist=3
if element_exist in my_set:# using if else condition here
    print("The element is exist")
else:
    print("The element is not exist")
```

```
The element is exist
```

In [8]:
```
"Question:-5"
my_set = {1, 2, 3, 4, 5}
count = 0
for _ in my_set:
    count += 1
print("Length of the set", count)
```

```
Length of the set 5
```

In [11]:
```
"Question:-6"
my_set={1,2,3,4,5}
my_set.clear()
print(my_set)
```

```
set()
```

In [13]:
```
"Question:-7"
even_numbers_set = set(range(2, 11, 2))
print(even_numbers_set)
```

```
{2, 4, 6, 8, 10}
```

In [17]:
```
"Question:-8"
odd_numbers_set = set(range(1, 11,2))
print(odd_numbers_set)
```

```
{1, 3, 5, 7, 9}
```

In [18]:
```
"Question:-9"
set1 = {1, 2, 3, 4, 5}
set2 = {3, 4, 5, 6, 7}
union_result = set1.union(set2)
print("Union using method", union_result)
union_result_operator = set1 | set2
print("Union using | operator", union_result_operator)
```

```
Union using method {1, 2, 3, 4, 5, 6, 7}
Union using | operator {1, 2, 3, 4, 5, 6, 7}
```

In [19]:
```
"Question:-10"
set1 = {1, 2, 3, 4, 5}
set2 = {3, 4, 5, 6, 7}
intersection_result = set1.intersection(set2)
print("Intersection using method", intersection_result)
intersection_result_operator = set1 & set2
print("Intersection using & operator", intersection_result_operator)
```

```
Intersection using method {3, 4, 5}
Intersection using & operator {3, 4, 5}
```

In [20]:
```
"Question:-11"
set1={3,4,5}
set2={1,2,3}
output_difference=set1-set2
print(output_difference)
```

```
{4, 5}
```

In [21]:
```
"Question:-12"
set1 = {1, 2, 3, 4, 5}
set2 = {2, 4}
is_subset_method = set2.issubset(set1)
print("Is set2 a subset of set1 (using method)?", is_subset_method)
```

```
Is set2 a subset of set1 (using method)? True
```

In [22]:
```
"Question:-13"
set1 = {1, 2, 3, 4, 5}
set2 = {2, 4}
is_superset_method = set1.issuperset(set2)
print("Is set1 a superset of set2 (using method)?", is_superset_method)
```

```
Is set1 a superset of set2 (using method)? True
```

In [1]:
```
"Question:-14"
my_list = [1, 2, 3, 4, 5]# enter the values
my_set = set(my_list)# convert the lsit to set
print(my_set)
```

```
{1, 2, 3, 4, 5}
```

In [3]:
```
"Question:-15"
my_set=[1,2,3,4]# here we enter the value
my_set= list(my_set)# convert the set to list
print(my_set)
```

```
[1, 2, 3, 4]
```

In [6]:
```
"Question:-16"
my_set = {1, 2, 3, 4, 5}
removed_element = my_set.pop()# Remove a random element using pop()
print("Removed element", removed_element)
print("Updated set", my_set)
```

```
Removed element 1
Updated set {2, 3, 4, 5}
```

In [8]:
```python
"Question:-17"
my_set = {1, 2, 3, 4, 5}
popped_element = my_set.pop()
print("Popped element:", popped_element)
print("Updated set:", my_set)
```

```
Popped element: 1
Updated set: {2, 3, 4, 5}
```

In [10]:
```python
"Question:-18"
set1 = {1, 2, 3, 4, 5}
set2 = {6, 7, 8, 9, 10}
no_common_elements = set1.isdisjoint(set2)

if no_common_elements:
    print("The sets have no elements in common.")
else:
    print("The sets have elements in common.")
```

```
The sets have no elements in common.
```

In [11]:
```python
"Question:-19"
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}
symmetric_diff_result = set1.symmetric_difference(set2)
print("Symmetric Difference using method", symmetric_diff_result)
```

```
Symmetric Difference using method {1, 2, 3, 6, 7, 8}
```

In [12]:
```python
"Question:-20"
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set1.update(set2)#updating the set
print("Updated set1:", set1)
```

```
Updated set1: {1, 2, 3, 4, 5}
```

In [14]:
```python
"Question:-21"
first_five_primes = {2, 3, 5, 7, 11}
print(first_five_primes)
```

```
{2, 3, 5, 7, 11}
```

In [15]:
```
"Question:-22"
set1 = {1, 2, 3, 4, 5}
set2 = {1, 2, 3, 4, 5}
are_identical_operator = set1 == set2
print("Are sets identical using == operator?", are_identical_operator)
```

```
Are sets identical using == operator? True
```

In [19]:
```
"Question:-23"
my_set = {1, 2, 3, 4, 5}
frozen_set = frozenset(my_set)
print("Original set", my_set)
print("Frozen set", frozen_set)
```

```
Original set {1, 2, 3, 4, 5}
Frozen set frozenset({1, 2, 3, 4, 5})
```

In [20]:
```
"Question:-24"
set1 = {1, 2, 3, 4, 5}
set2 = {6, 7, 8, 9, 10}
are_disjoint = set1.isdisjoint(set2)
if are_disjoint:
    print("The sets are disjoint (have no elements in common).")
else:
    print("The sets are not disjoint (have elements in common).")
```

```
The sets are disjoint (have no elements in common).
```

In [21]:
```
"Question:-25"
squares_set = {x**2 for x in range(1, 6)}
print(squares_set)
```

```
{1, 4, 9, 16, 25}
```

In [22]:
```
"Question:-26"
original_set = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
filtered_set = {x for x in original_set if x % 2 != 0}
print(filtered_set)
```

```
{1, 3, 5, 7, 9}
```

In [23]:
```
"Question:-27"
original_set = {1, 2, 3, 4, 5}
multiplied_set = {x * 2 for x in original_set}
print(multiplied_set)
```

```
{2, 4, 6, 8, 10}
```

In [24]:
```python
"Question:-28"
import random
random_set = set(random.sample(range(1, 11), 5))
print(random_set)
```

```
{2, 4, 5, 6, 10}
```

In [25]:
```python
"Question:-29"
my_set = set()
if len(my_set) == 0:
    print("The set is empty.")
else:
    print("The set is not empty.")
```

```
The set is empty.
```

In [26]:
```python
"Question:-30"
set_level_3_1 = frozenset({'item_3_1_1', 'item_3_1_2'})
set_level_3_2 = frozenset({'item_3_2_1', 'item_3_2_2'})

set_level_2_1 = frozenset({'item_2_1_1', 'item_2_1_2', set_level_3_1, set_level_3_2})
set_level_2_2 = frozenset({'item_2_2_1', 'item_2_2_2'})

nested_set = frozenset({'item_1', set_level_2_1, set_level_2_2})
print(nested_set)
```

```
frozenset({frozenset({'item_2_2_1', 'item_2_2_2'}), frozenset({frozenset({'item_3_2_1', 'item_3_2_2'}), frozenset({'ite
m_3_1_2', 'item_3_1_1'}), 'item_2_1_1', 'item_2_1_2'}), 'item_1'})
```

In [27]:
```python
"Question:-31"
my_set = {1, 2, 3, 4, 5}
element_to_remove = 3
my_set.discard(element_to_remove)
print(my_set)
```

```
{1, 2, 4, 5}
```

In [28]:
```python
"Question:-32"
set1 = {1, 2, 3}
set2 = {3, 2, 1}
if set1 == set2:
    print("The sets are equal.")
else:
    print("The sets are not equal.")
```

```
The sets are equal.
```

In [30]:
```
"Question:-33"
my_string = "Deepak singh"
my_set = set(my_string)
print(my_set)
```

```
{'a', 'i', 'D', ' ', 'h', 'g', 'e', 'k', 's', 'n', 'p'}
```

In [31]:
```
"Question:-34"
string_set = {'1', '2', '3', '4', '5'}
int_set = {int(x) for x in string_set}
print(int_set)
```

```
{1, 2, 3, 4, 5}
```

In [32]:
```
"Question:-35"
string_set = {'1', '2', '3', '4', '5'}
int_set = {int(x) for x in string_set}
print(int_set)
```

```
{1, 2, 3, 4, 5}
```

In [33]:
```
"Question:-36"
my_tuple = (1, 2, 3, 4, 5)
my_set = set(my_tuple)
print(my_set)
```

```
{1, 2, 3, 4, 5}
```

In [34]:
```
"Question:-37"
my_set=(1,2,3,4,5)
my_tuple=set(my_set)
print(my_set)
```

```
(1, 2, 3, 4, 5)
```

In [35]:
```
"Question:-38"
my_set = {10, 5, 8, 20, 15}
max_value = max(my_set)
print("Maximum value", max_value)
```

```
Maximum value 20
```

In [36]:
```
"Question:-39"
my_set = {10, 5, 8, 20, 15}
```

```
min_value = min(my_set)
print("Minimum  value", min_value)
```

```
Minimum  value 5
```

In [39]:
```
"Question:-40"
user_input_str = input("Enter elements separated by spaces: ")
elements_list = user_input_str.split()
user_set = set(map(int, elements_list))
print("User Set", user_set)
```

```
Enter elements separated by spaces: 1 2 3
User Set {1, 2, 3}
```

In [40]:
```
"Question:-41"
set1 = {1, 2, 3, 4}
set2 = {5, 6, 7, 8}
if set1.isdisjoint(set2):
    print("The sets have no common elements (intersection is empty).")
else:
    print("The sets have common elements.")
```

```
The sets have no common elements (intersection is empty).
```

In [41]:
```
"Question:-42"
def generate_fibonacci(n):
    fibonacci_set = set()
    a, b = 0, 1
    for _ in range(n):
        fibonacci_set.add(a)
        a, b = b, a + b
    return fibonacci_set
first_5_fibonacci = generate_fibonacci(5)
print("First 5 Fibonacci numbers:", first_5_fibonacci)
```

```
First 5 Fibonacci numbers: {0, 1, 2, 3}
```

In [42]:
```
"Question:-43"
my_list = [1, 2, 3, 2, 4, 5, 1, 6, 7, 7]
unique_list = list(set(my_list))
print("Original List", my_list)
print("List without Duplicates", unique_list)
```

```
Original List [1, 2, 3, 2, 4, 5, 1, 6, 7, 7]
List without Duplicates [1, 2, 3, 4, 5, 6, 7]
```

In [43]:
```
"Question:-44"
set1 = {1, 2, 3, 4}
set2 = {4, 3, 2, 1}
if set1 == set2:
    print("The sets have the same elements.")
else:
    print("The sets do not have the same elements.")
```

```
The sets have the same elements.
```

In [44]:
```
"Question:-45"
def generate_powers_of_2(n):
    powers_of_2_set = set()
    for i in range(n):
        powers_of_2_set.add(2 ** i)
    return powers_of_2_set
first_5_powers_of_2 = generate_powers_of_2(5)
print("First 5 powers of 2:", first_5_powers_of_2)
```

```
First 5 powers of 2: {1, 2, 4, 8, 16}
```

In [45]:
```
"Question:-46"
my_set = {1, 2, 3, 4, 5}
my_list = [3, 4, 5, 6, 7]
common_elements = set(my_list).intersection(my_set)
print("Common elements:", common_elements)
```

```
Common elements: {3, 4, 5}
```

In [46]:
```
"Question:-47"
n = 5
triangular_numbers_set = {(i * (i + 1)) // 2 for i in range(1, n + 1)}
print("First 5 triangular numbers", triangular_numbers_set)
```

```
First 5 triangular numbers {1, 3, 6, 10, 15}
```

In [47]:
```
"Question:-48"
set1 = {1, 2, 3, 4, 5}
set2 = {2, 4}
if set2.issubset(set1):
    print("set2 is a subset of set1.")
else:
    print("set2 is not a subset of set1.")
```

```
set2 is a subset of set1.
```

In [48]:
```
"Question:-49"
n = 8
alternating_set = {i % 2 for i in range(n)}
print("Alternating set of 1s and 0s:", alternating_set)
```

```
Alternating set of 1s and 0s: {0, 1}
```

In [49]:
```
"Question:-50"
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set3 = {5, 6, 7}
merged_set = set1.union(set2, set3)
print("Merged set:", merged_set)
```

```
Merged set: {1, 2, 3, 4, 5, 6, 7}
```

In [ ]: