7 Strings and dictionary operations.

This notebook features Python programs (61 to 70) focused on string manipulation and dictionary operations. It's ideal for beginners to practice data handling, text processing, and core Python concepts.

The notebook can be viewed here (https://github.com/amitvsuryavanshi04/SIC_programming_and_coding)

Program 61 spilt and join a string

```
# Split a string into a list of words
input_str = "Python program to split and join a string"
word_list = input_str.split() # By default, split on whitespace
# Join the list of words into a string
separator = " " # specify the separator between words
output_str = separator.join(word_list)
# Print the results
print("Original String:", input_str)
print("List of split Words:", word_list)
print("Joined String:", output_str)
```

```
Original String: Python program to split and join a string
List of split Words: ['Python', 'program', 'to', 'split', 'and', 'join', 'a', 'string']
Joined String: Python program to split and join a string
```

Program 62 to check if a given string is binary string or not binary string are those who have 0's and 1's in them

```
def is_binary_str(input_str):
# Iterate through each character in the input string
  for i in input_str:
# Check if the i is not '0' or '1'
    if i not in '01':
      return False # If any character is not '0' or '1', it's no
  return True # If all characters are '0' or '1', it's a binary stri
# Input string to check
input_str = "1001110"
```

```
# Check if the input string is a binary string
if is_binary_str(input_str):
  print(f"'{input_str}' is a binary string.")
else:
  print(f"'{input_str}' is not a binary string.")
```

```
'1001110' is a binary string.
```

Program 63 code to find uncommon words from two strings.

```
def uncommon_words(str1, str2):
# Split the strings into words and convert them to sets
  words1 = set(str1.split())
  words2 = set(str2.split())
# Find uncommon words by taking the set difference
  uncommon_words_set = words1.symmetric_difference(words2)
# Convert the set of uncommon words back to a list
  uncommon_words_list = list(uncommon_words_set)
  return uncommon_words_list
# Input two strings
string1 = "This is the first string"
string2 = "This is the second string"
# Find uncommon words between the two strings
uncommon = uncommon_words(string1, string2)
# Print the uncommon words
print("Uncommon words:", uncommon)
```

```
Uncommon words: ['second', 'first']
```

Program 64 code to find all duplicate characters in string.

```
def find_duplicates(input_str):
# Create an empty dictionary to store character counts
  char_count = {}
# Initialize a list to store duplicate characters
  duplicates = []
# Iterate through each character in the input string
  for i in input_str:
# If the character is already in the dictionary, increment its
```

```
    if i in char_count:
      char_count[i] += 1
    else:
      char_count[i] = 1
# Iterate through the dictionary and add characters with count > 1
  for i, count in char_count.items():
    if count > 1:
      duplicates.append(i)
  return duplicates
# Input a string
input_string = "amit suryavanshi"
# Find duplicate characters in the string
duplicate_chars = find_duplicates(input_string)
# Print the list of duplicate characters
print("Duplicate characters:", duplicate_chars)
```

```
Duplicate characters: ['a', 'i', 's']
```

Program 65 to check if string contains any special character

```
import re
def check_special_char(in_str):
# Define a regular expression pattern to match special characters
  pattern = r'[!@#$%^&*()_+{}\[\]:;<>,.?~\\\/\'"\-=]'
# Use re.search to find a match in the input string
  if re.search(pattern, in_str):
    return True
  else:
    return False
# Input a string
input_string = str(input("Enter a string: "))
# Check if the string contains any special characters
contains_special = check_special_char(input_string)
# Print the result
if contains_special:
  print("The string contains special characters.")
else:
  print("The string does not contain special characters.")
```

```
Enter a string: a@#@$D#dfasfa@#!~
The string contains special characters.
```

Program 66 to extract unique dictionary values.

```
# Sample dictionary
my_dict = {
  'a': 10,
  'b': 20,
  'c': 10,
  'd': 30,
  'e': 20
}
# Initialize an empty set to store unique values
uni_val = set()
# Iterate through the values of the dictionary
for i in my_dict.values():
# Add each value to the set
  uni_val.add(i)
# Convert the set of unique values back to a list (if needed)
unique_values_list = list(uni_val)
# Print the unique values
print("Unique values in the dictionary:", unique_values_list)
```

```
Unique values in the dictionary: [10, 20, 30]
```

Program 67 to find all the sum of all items in dictionary.

```
# Sample dictionary
my_dict = {
'a': 10,
'b': 20,
'c': 30,
'd': 40,
'e': 50
}
# Initialize a variable to store the sum
total_sum = 0
```

```
# Iterate through the values of the dictionary and add them to the tota
for i in my_dict.values():
  total_sum += i
# Print the sum of all items in the dictionary
print("Sum of all items in the dictionary:", total_sum)
```

```
Sum of all items in the dictionary: 150
```

Program 68 merging two dictionaries

```
# 1. Using the update() method:
dict1 = {'a': 1, 'b': 2}
dict2 = {'c': 3, 'd': 4}
dict1.update(dict2)
# The merged dictionary is now in dict1
print("Merged Dictionary (using update()):", dict1)
```

```
Merged Dictionary (using update()): {'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

```
# 2. Using dictionary unpacking
dict1 = {'a': 1, 'b': 2}
dict2 = {'c': 3, 'd': 4}
# Merge dict2 into dict1 using dictionary unpacking
merged_dict = {**dict1, **dict2}
# The merged dictionary is now in merged_dict
print("Merged Dictionary (using dictionary unpacking):", merged_dict)
```

```
Merged Dictionary (using dictionary unpacking): {'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

Program 69 code to convert ky-values list to flat dictionary.

```
key_values_list = [('a', 1), ('b', 2), ('c', 3), ('d', 4)]
# Initialize an empty dictionary
flat_dict = {}
# Iterate through the list and add key-value pairs to the dictionar
for key, value in key_values_list:
  flat_dict[key] = value
```

```
# Print the resulting flat dictionary
```

```
Flat Dictionary: {'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

Program 70 insertion at the beginning in ordered Dictionary

```
from collections import OrderedDict
# Create an OrderedDict
ordered_dict = OrderedDict([('b', 2), ('c', 3), ('d', 4)])
# Item to insert at the beginning
new_item = ('a', 1)
# Create a new OrderedDict with the new item as the first element
new_ordered_dict = OrderedDict([new_item])
# Merge the new OrderedDict with the original OrderedDict
new_ordered_dict.update(ordered_dict)
# Print the updated OrderedDict
print("Updated OrderedDict:", new_ordered_dict)
```

```
Updated OrderedDict: OrderedDict([('a', 1), ('b', 2), ('c', 3), ('d', 4)])
```