

In Python, there are several built-in data structures, each with its own methods and functionalities. The commonly used ones are:

1. Lists

- Methods: **`append()`**, **`extend()`**, **`insert()`**, **`remove()`**, **`pop()`**, **`index()`**, **`count()`**, **`sort()`**, **`reverse()`**, etc.

Example of list methods

```
my_list = [1, 2, 3, 4, 5]
my_list.append(6)
print(my_list) # Output: [1, 2, 3, 4, 5, 6]
my_list.extend([7, 8])
print(my_list) # Output: [1, 2, 3, 4, 5, 6, 7, 8]
my_list.remove(3)
print(my_list) # Output: [1, 2, 4, 5, 6, 7, 8]
```

2. Tuples

- Tuples are immutable, so they have fewer methods compared to lists.
- Methods: **`count()`**, **`index()`**

Example of tuple methods

```
my_tuple = (1, 2, 2, 3, 4)
count_2 = my_tuple.count(2)
print(count_2) # Output: 2
index_3 = my_tuple.index(3)
print(index_3) # Output: 3
```

3. Sets

- Methods: **`add()`**, **`remove()`**, **`discard()`**, **`pop()`**, **`clear()`**, **`union()`**, **`intersection()`**, **`difference()`**, **`symmetric_difference()`**, **`etc.`**

Example of set methods

```
my_set1 = {1, 2, 3}
my_set2 = {3, 4, 5}
my_set1.add(6)
print(my_set1) # Output: {1, 2, 3, 6}
my_set2.remove(4)
print(my_set2) # Output: {3, 5}
set_union = my_set1.union(my_set2)
print(set_union) # Output: {1, 2, 3, 5, 6}
```

4. Dictionaries

- Methods: **keys()**, **values()**, **items()**, **get()**, **pop()**, **popitem()**, **update()**, **clear()**, etc.

Example of dictionary methods

```
my_dict = {'a': 1, 'b': 2, 'c': 3}
```

```
print(my_dict.keys()) # Output: dict_keys(['a', 'b', 'c'])
```

```
print(my_dict.values()) # Output: dict_values([1, 2, 3])
```

```
my_dict['d'] = 4
```

```
print(my_dict) # Output: {'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

5. Strings

- Strings are immutable sequences of characters.
- Methods: **upper()**, **lower()**, **capitalize()**, **split()**, **join()**, **strip()**, **replace()**, **find()**, **count()**, etc.

Example of string methods

```
my_string = "Hello, World!"
```

```
print(my_string.upper()) # Output: HELLO, WORLD!
```

```
print(my_string.split()) # Output: ['Hello,', 'World!']
```

```
print(my_string.replace('Hello', 'Hi')) # Output: Hi, World!
```

6. Arrays (from the `array` module)

- Arrays are similar to lists but can hold only a single data type.
- Methods: **append()**, **extend()**, **insert()**, **remove()**, **pop()**, **index()**, **count()**, **reverse()**, etc.

```
import array
```

Example of array methods

```
my_array = array.array('i', [1, 2, 3, 4])
```

```
my_array.append(5)
```

```
print(my_array) # Output: array('i', [1, 2, 3, 4, 5])
```

```
my_array.remove(3)
```

```
print(my_array) # Output: array('i', [1, 2, 4, 5])
```

PROJECT1:WORD COUNT TOOL

```
from collections import Counter
```

```
def count_words(text):
```

```
    words = text.split()
```

```
    word_count = len(words)
```

```
    return word_count
```

```
def analyze_word_frequency(text):
```

```
    words = text.split()
```

```
    word_frequency = Counter(words)
```

```
    return word_frequency
```

```
if __name__ == "__main__":
```

```
    print("Word Count Tool\n")
```

```
    choice = input("Enter 'T' to enter text manually or 'F' to input a text  
file: ")
```

```
    if choice.upper() == 'T':
```

```
        user_input = input("Enter your text: ")
```

```
        total_words = count_words(user_input)
```

```
        print(f"\nTotal words in the text: {total_words}")
```

```
        word_frequency = analyze_word_frequency(user_input)
```

```
        print("\nWord Frequency:")
```

```
        for word, count in word_frequency.items():
```

```
            print(f"{word}: {count}")
```

```
    elif choice.upper() == 'F':
```

```
        file_name = input("Enter the file name: ")
```

```

try:
    with open(file_name, 'r') as file:
        file_text = file.read()
        total_words = count_words(file_text)
        print(f"\nTotal words in the text: {total_words}")

        word_frequency = analyze_word_frequency(file_text)
        print("\nWord Frequency:")
        for word, count in word_frequency.items():
            print(f"{word}: {count}")
except FileNotFoundError:
    print("File not found. Please enter a valid file name.")
else:
    print("Invalid choice. Please enter 'T' or 'F'.")

```

PROJECT2:VIDEO TO AUDIO CONVERTER

pip install moviepy

```
from moviepy.editor import VideoFileClip
```

```

def video_to_audio(video_path, audio_path):
    try:
        video = VideoFileClip(video_path)
        audio = video.audio
        audio.write_audiofile(audio_path)
        print(f"Audio extracted successfully and saved as {audio_path}")
    except Exception as e:
        print(f"An error occurred: {str(e)}")

if __name__ == "__main__":
    input_video = input("Enter the path of the video file: ")

```

```
output_audio = input("Enter the path to save the audio file (including  
the file name and extension): ")
```

```
video_to_audio(input_video, output_audio)
```

PROJECT3:QUOTES APP

```
pip install requests
```

```
import requests
```

```
def fetch_quote():
```

```
    try:
```

```
        response = requests.get("https://api.quotable.io/random")
```

```
        if response.status_code == 200:
```

```
            quote_data = response.json()
```

```
            return quote_data.get("content"), quote_data.get("author")
```

```
        else:
```

```
            return None, None
```

```
    except requests.RequestException as e:
```

```
        print(f"Error occurred: {e}")
```

```
        return None, None
```

```
if __name__ == "__main__":
```

```
    print("Welcome to the Quotes App!\n")
```

```
    while True:
```

```
        input("Press Enter to get a new quote or 'Q' to quit: ")
```

```
        quote, author = fetch_quote()
```

```
        if quote and author:
```

```
    print(f"Quote: {quote}")
    print(f"Author: {author}\n")
else:
    print("Failed to fetch a quote. Please try again later or check
your internet connection.\n")

user_choice = input("Would you like another quote? (Y/N):
").strip().lower()
if user_choice != 'y':
    print("Thank you for using the Quotes App!")
    break
```