## To-Do List Python Program Documentation

### Overview

This Python program allows users to manage a simple to-do list. The user can add tasks, mark them as completed, remove tasks, and display the list of tasks. Each task has a name and a status indicating whether it is completed or pending.

### Program Structure

The program is composed of several functions, each responsible for a specific operation related to the to-do list. The main function serves as the entry point and provides a menu for the user to choose actions.

### Functions

#### display\_todo\_list(todo\_list)

**Description**: This function prints the current to-do list to the console. Each task is displayed with its index, name, and completion status.

**Parameters**:

* todo\_list (list): A list of dictionaries where each dictionary represents a task with keys 'name' (string) and 'completed' (boolean).

**Example Usage**:

**Code:**

todo\_list = [{'name': 'Task 1', 'completed': False}, {'name': 'Task 2', 'completed': True}]

#### add\_task(todo\_list)

**Description**: This function prompts the user to enter the name of a new task and adds it to the to-do list with a status of 'Pending'.

**Parameters**:

* todo\_list (list): The list of tasks to which the new task will be added.

**Example Usage**:

Python code:

todo\_list = []

add\_task(todo\_list)

#### mark\_task\_completed(todo\_list)

**Description**: This function displays the to-do list and prompts the user to select a task to mark as completed.

**Parameters**:

* todo\_list (list): The list of tasks from which a task will be marked as completed.

**Example Usage**:

python

Copy code

todo\_list = [{'name': 'Task 1', 'completed': False}]

mark\_task\_completed(todo\_list)

#### remove\_task(todo\_list)

**Description**: This function displays the to-do list and prompts the user to select a task to remove from the list.

**Parameters**:

* todo\_list (list): The list of tasks from which a task will be removed.

**Example Usage**:

**Code:**

todo\_list = [{'name': 'Task 1', 'completed': False}]

remove\_task(todo\_list)

### Main Function

#### main()

**Description**: This function serves as the entry point of the program. It initializes an empty to-do list and provides a menu for the user to perform various actions on the list. The user can choose to display the list, add a task, mark a task as completed, remove a task, or quit the program.

**Example Usage**:

**Code:**

if \_\_name\_\_ == "\_\_main\_\_":

main()

### Complete Python Code :

def display\_todo\_list(todo\_list):

print("\nTo-Do List:")

for i, task in enumerate(todo\_list, 1):

status = "Completed" if task['completed'] else "Pending"

print(f"{i}. {task['name']} - {status}")

print()

def add\_task(todo\_list):

task\_name = input("Enter the task name: ")

todo\_list.append({'name': task\_name, 'completed': False})

print("Task added.\n")

def mark\_task\_completed(todo\_list):

display\_todo\_list(todo\_list)

task\_number = int(input("Enter the task number to mark as completed: "))

if 1 <= task\_number <= len(todo\_list):

todo\_list[task\_number - 1]['completed'] = True

print("Task marked as completed.\n")

else:

print("Invalid task number.\n")

def remove\_task(todo\_list):

display\_todo\_list(todo\_list)

task\_number = int(input("Enter the task number to remove: "))

if 1 <= task\_number <= len(todo\_list):

todo\_list.pop(task\_number - 1)

print("Task removed.\n")

else:

print("Invalid task number.\n")

def main():

todo\_list = []

while True:

print("Options:")

print("1. Display to-do list")

print("2. Add a task")

print("3. Mark a task as completed")

print("4. Remove a task")

print("5. Quit")

choice = input("Enter your choice: ")

if choice == '1':

display\_todo\_list(todo\_list)

elif choice == '2':

add\_task(todo\_list)

elif choice == '3':

mark\_task\_completed(todo\_list)

elif choice == '4':

remove\_task(todo\_list)

elif choice == '5':

print("Goodbye!")

break

else:

print("Invalid choice. Please enter a number between 1 and 5.\n")

if \_\_name\_\_ == "\_\_main\_\_":

main()

### Execution

To run the program, simply execute the script. The program will present a menu to the user for interacting with the to-do list. The user can perform the following actions:

* Display the current to-do list
* Add a new task
* Mark an existing task as completed
* Remove a task from the list
* Quit the program

### Notes

* Ensure that the input values for task numbers are within the valid range of existing tasks to prevent errors.
* The to-do list is stored in memory, so it will be reset each time the program is run.

This documentation provides a comprehensive guide to understanding and using the to-do list program.