Project 3: To-Do List

1. Introduction

A To-Do List is a basic yet powerful productivity tool that allows users to create, manage, and organize their daily tasks efficiently. This project aims to build a simple command-line-based To-Do List using Python. It includes essential task management operations such as adding, editing, completing, and deleting tasks.

2. Objectives

- Enable users to add new tasks.
- Allow marking tasks as completed.
- Provide options to edit and delete tasks.
- Store and retrieve tasks from a file.

3. Features

- Add new tasks with descriptions.
- Mark tasks as complete.
- Edit existing tasks.
- Delete tasks.
- Save tasks to a text file.
- Load tasks from a file when the program starts.

4. Technologies Used

- Programming Language: Python
- File Handling: For persistent storage

5. Implementation

Code:

```
import os
import json

TASKS_FILE = "tasks.json"

def load_tasks():
    if os.path.exists(TASKS_FILE):
        with open(TASKS_FILE, 'r') as f:
            return json.load(f)
    return []

def save_tasks(tasks):
    with open(TASKS_FILE, 'w') as f:
        json.dump(tasks, f, indent=4)

def show_tasks(tasks):
    if not tasks:
        print("No tasks available.")
        return
    for i, task in enumerate(tasks, 1):
        status = "Done" if task['done'] else "Pending"
        print(f"{i}. {task['task']} - [{status}]")

def add_task(tasks):
    task_text = input("Enter task description: ")
    tasks.append({"task": task_text, "done": False})
    save_tasks(tasks)

def mark_done(tasks):
    show_tasks(tasks)
    index = int(input("Enter task number to mark as done: ")) - 1
    if 0 <= index < len(tasks):
        tasks[index]['done'] = True
        save_tasks(tasks)

def edit_task(tasks):
    show_tasks(tasks)
    index = int(input("Enter task number to edit: ")) - 1
    if 0 <= index < len(tasks):
        new_text = input("Enter new task description: ")
        tasks[index]['task'] = new_text
        save_tasks(tasks)

def delete_task(tasks):
    show_tasks(tasks)
    index = int(input("Enter task number to delete: ")) - 1
    if 0 <= index < len(tasks):
        tasks.pop(index)
        save_tasks(tasks)


def main():
    tasks = load_tasks()
    while True:
        print("\n1. Show Tasks\n2. Add Task\n3. Mark Task as Done\n4. Edit Task\n5. Delete Task\n6. Exit")
        choice = input("Choose an option: ")

        if choice == '1':
            show_tasks(tasks)
        elif choice == '2':
            add_task(tasks)
        elif choice == '3':
            mark_done(tasks)
        elif choice == '4':
            edit_task(tasks)
        elif choice == '5':
            delete_task(tasks)
        elif choice == '6':
            break
        else:
            print("Invalid choice. Please try again.")


if __name__ == "__main__":
    main()
```

~~~
```
~~~

6. Data Storage

Tasks are stored in a `tasks.json` file in JSON format. This ensures persistence across program runs and allows structured data management.

7. Conclusion

This To-Do List project serves as a foundational task management tool. It introduces basic programming concepts like file handling, data structures, and user interaction in Python. It's easily extendable to a GUI or web-based version.

8. Future Enhancements

- Add a graphical interface using Tkinter or PyQt.
- Implement due dates and priorities.
- Add task categorization or tagging.
- Enable syncing with cloud storage or databases.