TO DO LIST

```
#include <iostream>
#include <vector>
#include <string>
using namespace std;

// Task structure to hold task description and its completion status
struct Task {
    string description;
    bool isCompleted;
};

// Function to add a task
void addTask(vector<Task>& tasks) {
    string taskDescription;
    cout << "Enter task description: ";
    cin.ignore(); // To ignore the newline character left in the buffer
    getline(cin, taskDescription); // Take multi-word task description
    tasks.push_back({taskDescription, false}); // Add task to the list with pending status
    cout << "Task added successfully!" << endl;
}
```

```
// Function to view tasks
void viewTasks(const vector<Task>& tasks) {
    if (tasks.empty()) {
        cout << "No tasks in the list!" << endl;
        return;
    }

    cout << "\nTask List:" << endl;
    for (size_t i = 0; i < tasks.size(); ++i) {
        cout << (i + 1) << ". " << tasks[i].description
             << " [" << (tasks[i].isCompleted ? "Completed" :
"Pending") << "]" << endl;
    }
}

// Function to mark a task as completed
void markTaskCompleted(vector<Task>& tasks) {
    int taskIndex;
    viewTasks(tasks);

    cout << "Enter the task number to mark as completed: ";
    cin >> taskIndex;

    if (taskIndex > 0 && taskIndex <= tasks.size()) {
        tasks[taskIndex - 1].isCompleted = true;
        cout << "Task marked as completed!" << endl;
    } else {
```

```
        cout << "Invalid task number!" << endl;
    }
}

// Function to remove a task
void removeTask(vector<Task>& tasks) {
    int taskIndex;
    viewTasks(tasks);

    cout << "Enter the task number to remove: ";
    cin >> taskIndex;

    if (taskIndex > 0 && taskIndex <= tasks.size()) {
        tasks.erase(tasks.begin() + taskIndex - 1); // Remove task
from list
        cout << "Task removed successfully!" << endl;
    } else {
        cout << "Invalid task number!" << endl;
    }
}

int main() {
    vector<Task> tasks;
    int choice;

    do {
        cout << "\n--- To-Do List Manager ---" << endl;
```

```
cout << "1. Add Task" << endl;
cout << "2. View Tasks" << endl;
cout << "3. Mark Task as Completed" << endl;
cout << "4. Remove Task" << endl;
cout << "5. Exit" << endl;
cout << "Enter your choice: ";
cin >> choice;

switch (choice) {
    case 1:
        addTask(tasks);
        break;
    case 2:
        viewTasks(tasks);
        break;
    case 3:
        markTaskCompleted(tasks);
        break;
    case 4:
        removeTask(tasks);
        break;
    case 5:
        cout << "Exiting program..." << endl;
        break;
    default:
        cout << "Invalid choice! Please try again." << endl;
}
```

```
    } while (choice != 5);

    return 0;
}
```