

C:\turboc\Untitled-1.cpp

```
1  #include <iostream>
2  #include <string>
3  #include <vector>
4
5  using namespace std;
6
7  // Define Task class
8  class Task {
9  private:
10     string name; // Task name
11     string description; // Task description
12     string dueDate; // Task due date
13     bool completed; // Task completion status
14
15 public:
16     // Constructor to initialize a task
17     Task(const string& name, const string& description,
18         const string& dueDate)
19         : name(name)
20         , description(description)
21         , dueDate(dueDate)
22         , completed(false)
23     {
24     }
25
26     // Getter for task name
27     string getName() const { return name; }
28
29     // Getter for task description
30     string getDescription() const { return description; }
31
32     // Getter for task due date
33     string getDueDate() const { return dueDate; }
34
35     // Getter for task completion status
36     bool isCompleted() const { return completed; }
37
38     // Setter for task name
39     void setName(const string& name) { this->name = name; }
40
41     // Setter for task description
42     void setDescription(const string& description)
43     {
44         this->description = description;
45     }
46
47     // Setter for task due date
48     void setDueDate(const string& dueDate)
49     {
50         this->dueDate = dueDate;
51     }
52
53     // Mark the task as completed
54     void markCompleted() { completed = true; }
```

```
55
56 // Display task details
57 void displayTask() const
58 {
59     cout << name << " ("
60         << (completed ? "Completed" : "Pending")
61         << ") - Due: " << dueDate << endl
62         << "    Description: " << description << endl;
63 }
64 };
65
66 // Define ToDoList class
67 class ToDoList {
68 private:
69     vector<Task> tasks; // List of tasks
70
71 public:
72     // Display the menu
73     void displayMenu()
74     {
75         cout
76             << "\n----- To-Do List Menu ----- \n";
77         cout << "1. Add Task\n";
78         cout << "2. Delete Task\n";
79         cout << "3. Display Tasks\n";
80         cout << "4. Mark Task as Completed\n";
81         cout << "5. Edit Task\n";
82         cout << "6. Exit\n";
83         cout << "-----"
84             << "\n";
85     }
86
87     // Add a new task
88     void addTask()
89     {
90         string name, description, dueDate;
91         cout << "Enter task name: ";
92         cin.ignore();
93         getline(cin, name);
94         cout << "Enter task description: ";
95         getline(cin, description);
96         cout << "Enter task due date (YYYY-MM-DD): ";
97         getline(cin, dueDate);
98
99         tasks.emplace_back(name, description, dueDate);
100         cout << "Task added successfully!" << endl;
101     }
102
103     // Delete a task
104     void deleteTask()
105     {
106         if (tasks.empty()) {
107             cout << "No tasks to delete!" << endl;
108             return;
109         }
110         cout << "Tasks:" << endl;
```

```
111     for (int i = 0; i < tasks.size(); ++i) {
112         cout << i + 1 << ". " << tasks[i].getName()
113             << endl;
114     }
115     cout << "Enter the task number to delete: ";
116     int taskNumber;
117     cin >> taskNumber;
118     if (taskNumber >= 1 && taskNumber <= tasks.size()) {
119         tasks.erase(tasks.begin() + taskNumber - 1);
120         cout << "Task deleted successfully!" << endl;
121     }
122     else {
123         cout << "Invalid task number!" << endl;
124     }
125 }
126
127 // Display all tasks
128 void displayTasks()
129 {
130     if (tasks.empty()) {
131         cout << "No tasks to display!" << endl;
132         return;
133     }
134     cout << "Tasks:" << endl;
135     for (int i = 0; i < tasks.size(); ++i) {
136         cout << i + 1 << ". ";
137         tasks[i].displayTask();
138     }
139 }
140
141 // Mark a task as completed
142 void markTaskCompleted()
143 {
144     if (tasks.empty()) {
145         cout << "No tasks to mark as completed!"
146             << endl;
147         return;
148     }
149     cout << "Tasks:" << endl;
150     for (int i = 0; i < tasks.size(); ++i) {
151         cout << i + 1 << ". " << tasks[i].getName()
152             << endl;
153     }
154     cout << "Enter the task number to mark as "
155         << "completed: ";
156     int taskNumber;
157     cin >> taskNumber;
158     if (taskNumber >= 1 && taskNumber <= tasks.size()) {
159         tasks[taskNumber - 1].markCompleted();
160         cout << "Task marked as completed!" << endl;
161     }
162     else {
163         cout << "Invalid task number!" << endl;
164     }
165 }
166
```

```
167 // Edit a task
168 void editTask()
169 {
170     if (tasks.empty()) {
171         cout << "No tasks to edit!" << endl;
172         return;
173     }
174     cout << "Tasks:" << endl;
175     for (int i = 0; i < tasks.size(); ++i) {
176         cout << i + 1 << ". " << tasks[i].getName()
177             << endl;
178     }
179     cout << "Enter the task number to edit: ";
180     int taskNumber;
181     cin >> taskNumber;
182     if (taskNumber >= 1 && taskNumber <= tasks.size()) {
183         Task& task = tasks[taskNumber - 1];
184         string name, description, dueDate;
185         cout << "Enter new task name (current: "
186             << task.getName() << "): ";
187         cin.ignore();
188         getline(cin, name);
189         cout << "Enter new task description (current: "
190             << task.getDescription() << "): ";
191         getline(cin, description);
192         cout << "Enter new task due date (current: "
193             << task.getDueDate() << "): ";
194         getline(cin, dueDate);
195
196         task.setName(name);
197         task.setDescription(description);
198         task.setDueDate(dueDate);
199         cout << "Task updated successfully!" << endl;
200     }
201     else {
202         cout << "Invalid task number!" << endl;
203     }
204 }
205
206 // Run the to-do list application
207 void run()
208 {
209     int choice;
210     do {
211         displayMenu();
212         cout << "Enter your choice: ";
213         cin >> choice;
214
215         switch (choice) {
216             case 1:
217                 addTask();
218                 break;
219             case 2:
220                 deleteTask();
221                 break;
222             case 3:
```

```
223         displayTasks();
224         break;
225     case 4:
226         markTaskCompleted();
227         break;
228     case 5:
229         editTask();
230         break;
231     case 6:
232         cout << "Exiting program. Bye!" << endl;
233         break;
234     default:
235         cout << "Invalid choice. Please try again!"
236              << endl;
237     }
238 } while (choice != 6);
239 }
240 };
241
242 // Main function
243 int main()
244 {
245     // Create a ToDoList object and run the application
246     ToDoList toDoList;
247     toDoList.run();
248     return 0;
249 }
```