

COMSATS UNIVERSITY ISLAMABAD
ATTOCK CAMPUS
DEPARTMENT OF COMPUTER SCIENCE
PROGRAM BSE

NAME: **Muhammad Usman**

REG. NO: **SP23-BSE-055**

ASSIGNMENT: **01(Data Structure)**

DATE: **24-SEP-2024**

SUBMITTED TO: **Sir M. Kamran**

Introduction:

This assignment's goal was to use a linked list data structure to build a task management system. The task management system offers the ability to create tasks, see all tasks, delete the task with the highest priority, and remove a task based on its ID. The linked list structure is ideal for handling a dynamic list of tasks because it guarantees effective insertion and deletion operations.

Code Overview:

Task-Node Structure:

- Describes a linked list node.
- comprises the task ID, a pointer to the following node, a description, and a priority.
- The node's properties are initialized by the constructor.

TaskList Structure:

- Represent the complete list of tasks.
- It has a pointer to the linked list's head, or initial node.
- **It carries out the following tasks:**

Add task:

Builds a new TaskNode and adds it, prioritized, to the linked list.

The new task becomes the head if it has the highest importance.

In all other cases, it is appended to the list at the proper location.

Remove highest priority task:

Eliminates the head, or first node from the linked list signifying the job with the highest priority.

Removing a task by ID:

Finds the node with the given ID by iterating through the linked list.

Removes the node from the list if it is discovered.

If unavailable, an error message appears.

View tasks:

Iterates through the linked list, printing details about every task along the way.

Destructor:

To stop memory leaks, all nodes in the linked list have their memory allocated.

Main Function:

- Generates a TaskList class instance.

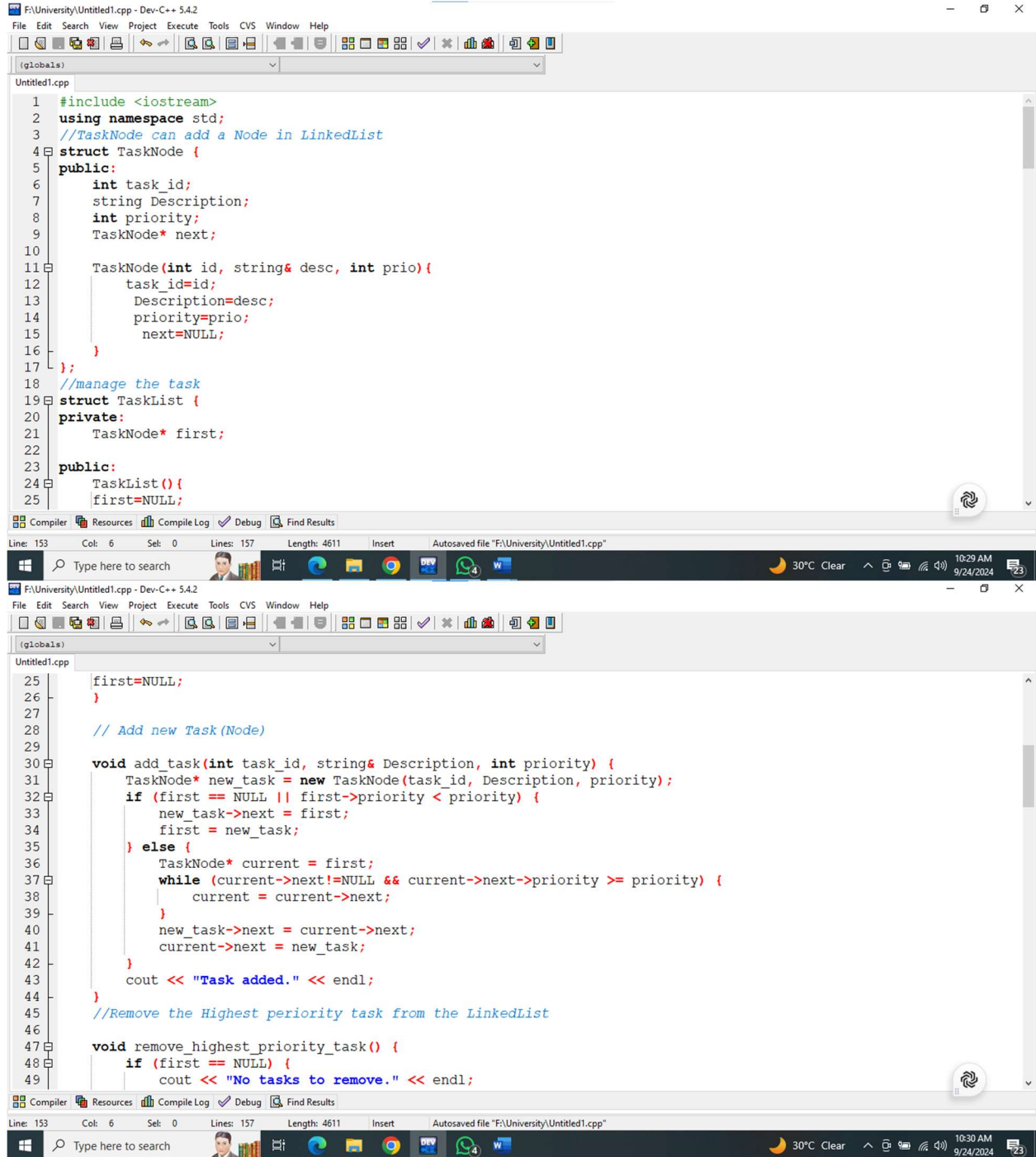
- Offers a menu-driven user interface so that users can communicate with task management systems.
- Takes in user input and, in accordance with the user's selection, calls the relevant functions.

Conclusion :

I now have a better knowledge of linked lists and how they are used in data structures thanks to this assignment. I gained knowledge on how to perform insertion, deletion, and traversal among other operations on linked lists. Managing edge circumstances, such empty lists or jobs with the same priority, presented additional difficulties for me. By carefully examining these possibilities and designing robust code I was able to create a viable and efficient task management system.

This assignment has improved my abilities to deal with data and solve problems.

Screenshots of Code



The image displays two screenshots of a C++ code editor, likely Dev-C++, showing the implementation of a linked list for task management. The code is written in a file named 'Untitled1.cpp'.

Top Screenshot: Shows the initial part of the code, including the inclusion of `<iostream>`, using the `std` namespace, and defining the `TaskNode` struct. The struct has attributes `task_id`, `Description`, `priority`, and a pointer to the next node (`TaskNode* next`). A constructor `TaskNode(int id, string& desc, int prio)` initializes these attributes. Below this, the `TaskList` struct is defined with a private attribute `TaskNode* first` and a public constructor `TaskList()` that sets `first` to `NULL`.

```
1 #include <iostream>
2 using namespace std;
3 //TaskNode can add a Node in LinkedList
4 struct TaskNode {
5     public:
6         int task_id;
7         string Description;
8         int priority;
9         TaskNode* next;
10
11     TaskNode(int id, string& desc, int prio){
12         task_id=id;
13         Description=desc;
14         priority=prio;
15         next=NULL;
16     }
17 };
18 //manage the task
19 struct TaskList {
20     private:
21         TaskNode* first;
22     public:
23         TaskList(){
24             first=NULL;
25 }
```

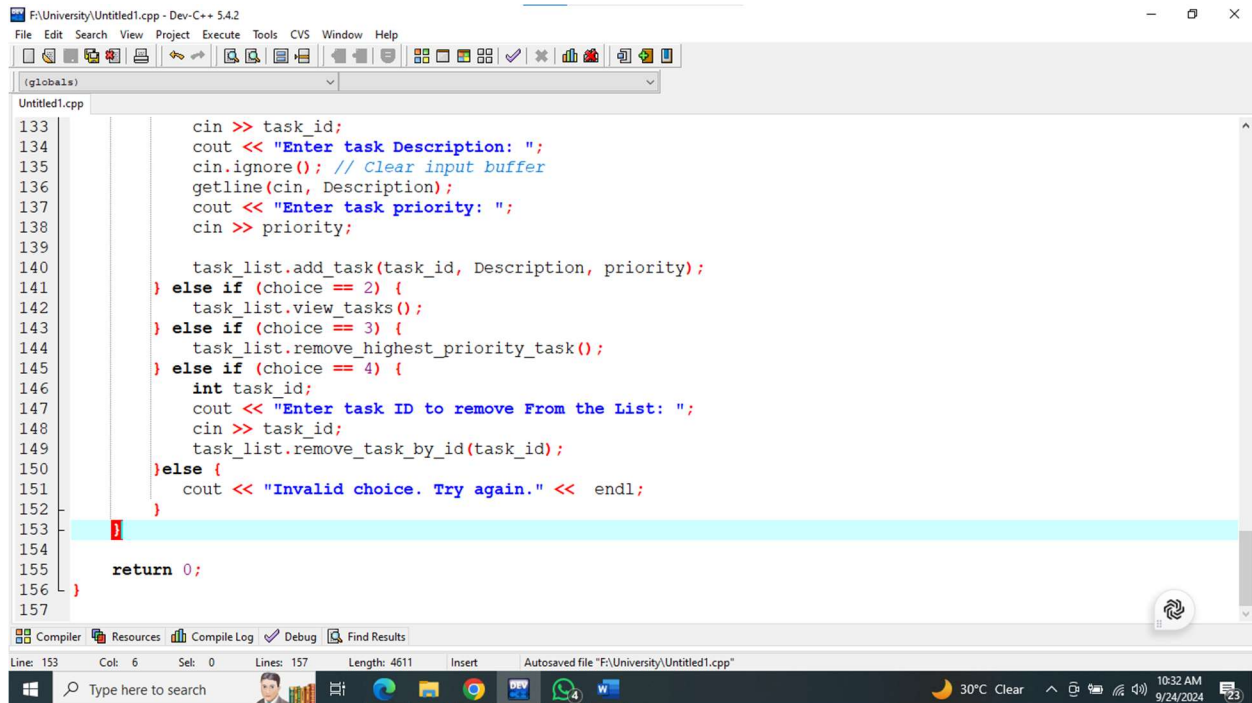
Bottom Screenshot: Shows the continuation of the code, including the implementation of the `add_task` function and the `remove_highest_priority_task` function. The `add_task` function takes `task_id`, `Description`, and `priority` as arguments, creates a new `TaskNode`, and inserts it into the linked list based on its priority. The `remove_highest_priority_task` function removes the task with the highest priority from the linked list.

```
25     first=NULL;
26 }
27
28 // Add new Task(Node)
29
30 void add_task(int task_id, string& Description, int priority) {
31     TaskNode* new_task = new TaskNode(task_id, Description, priority);
32     if (first == NULL || first->priority < priority) {
33         new_task->next = first;
34         first = new_task;
35     } else {
36         TaskNode* current = first;
37         while (current->next!=NULL && current->next->priority >= priority) {
38             current = current->next;
39         }
40         new_task->next = current->next;
41         current->next = new_task;
42     }
43     cout << "Task added." << endl;
44 }
45 //Remove the Highest periority task from the LinkedList
46
47 void remove_highest_priority_task() {
48     if (first == NULL) {
49         cout << "No tasks to remove." << endl;
50     }
51 }
```

```
F:\University\Untitled1.cpp - Dev-C++ 5.4.2
File Edit Search View Project Execute Tools CVS Window Help
(globals)
Untitled1.cpp
49         cout << "No tasks to remove." << endl;
50         return;
51     }
52     TaskNode* removed_task = first;
53     first = first->next;
54     cout << "Removed task with ID: " << removed_task->task_id
55           << ", Description: " << removed_task->Description << endl;
56     delete removed_task;
57 }
58 //Remove task by ID
59 void remove_task_by_id(int task_id) {
60     if (first == NULL) {
61         cout << "No tasks to remove." << endl;
62         return;
63     }
64
65     if (first->task_id == task_id) {
66         TaskNode* removed_task = first;
67         first = first->next;
68         cout << "Removed task with ID: " << task_id << endl;
69         delete removed_task;
70         return;
71     }
72
73     TaskNode* current = first;
```

```
F:\University\Untitled1.cpp - Dev-C++ 5.4.2
File Edit Search View Project Execute Tools CVS Window Help
(globals)
Untitled1.cpp
73     TaskNode* current = first;
74     while (current->next != NULL && current->next->task_id != task_id) {
75         current = current->next;
76     }
77
78     if (current->next == NULL) {
79         cout << "Task with ID: " << task_id << " not found." << endl;
80     } else {
81         TaskNode* removed_task = current->next;
82         current->next = current->next->next;
83         cout << "Removed task with ID: " << task_id << endl;
84         delete removed_task;
85     }
86 }
87 //for veiwing All Task
88
89 void view_tasks() {
90     if (first == NULL) {
91         cout << "No tasks available." << endl;
92         return;
93     }
94     TaskNode* current = first;
95     cout << "Tasks:" << endl;
96     while (current) {
97         cout << "ID: " << current->task_id
```

```
F:\University\Untitled1.cpp - Dev-C++ 5.4.2
File Edit Search View Project Execute Tools CVS Window Help
(globals)
Untitled1.cpp
97         cout << "ID: " << current->task_id
98             << ", Description: " << current->Description
99             << ", Priority: " << current->priority << endl;
100         current = current->next;
101     }
102 }
103 //destructor to free up the memory space
104 ~TaskList() {
105     while (first) {
106         TaskNode* temp = first;
107         first = first->next;
108         delete temp;
109     }
110 }
111 };
112
113 int main() {
114     TaskList task_list;
115
116     while (true) {
117         cout << "\nTask Management System" << endl;
118         cout << "1. Add a new task" << endl;
119         cout << "2. View all tasks" << endl;
120         cout << "3. Remove the highest priority task from the list" << endl;
121         cout << "4. Remove a task by ID" << endl;
122
123         int choice;
124         cout << "Enter your choice: ";
125         cin >> choice;
126
127         if (choice == 1) {
128             int task_id;
129             string Description;
130             int priority;
131
132             cout << "Enter task ID: ";
133             cin >> task_id;
134             cout << "Enter task Description: ";
135             cin.ignore(); // Clear input buffer
136             getline(cin, Description);
137             cout << "Enter task priority: ";
138             cin >> priority;
139
140             task_list.add_task(task_id, Description, priority);
141         } else if (choice == 2) {
142             task_list.view_tasks();
143         } else if (choice == 3) {
144             task_list.remove_highest_priority_task();
145         } else if (choice == 4) {
146             task_list.remove_task_by_id();
147         }
148     }
149 }
```

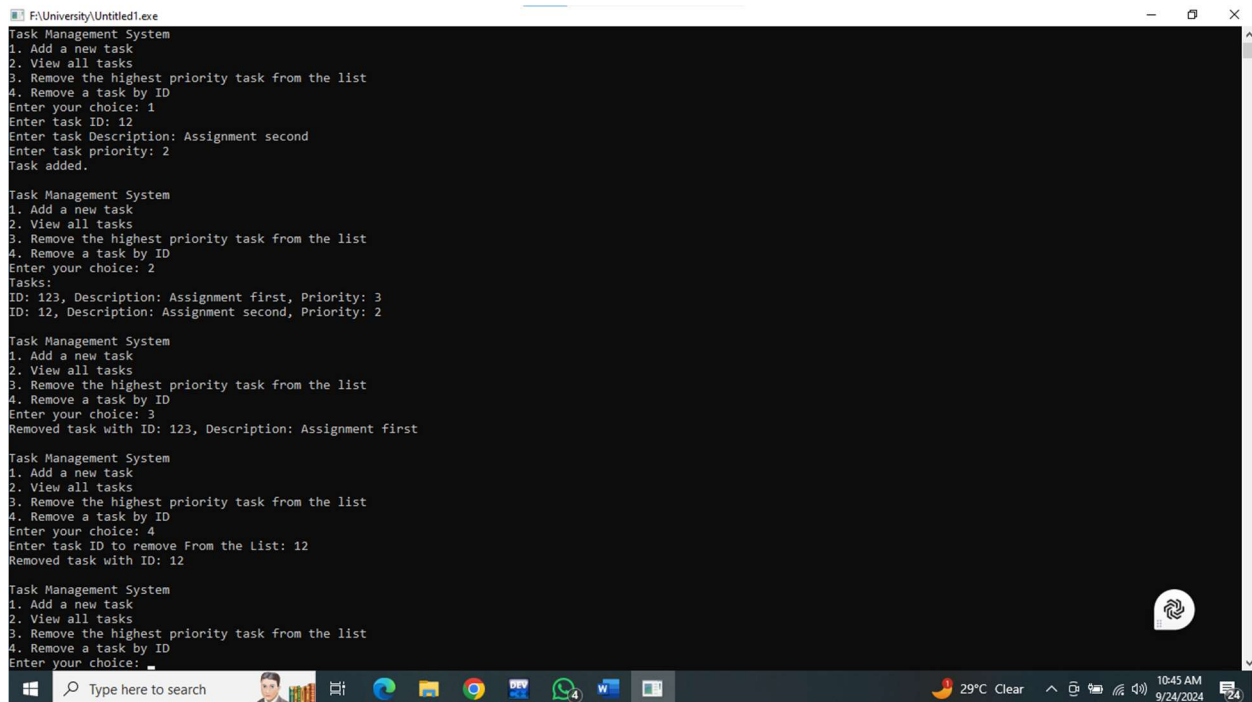


The screenshot shows a C++ IDE with the file 'Untitled1.cpp' open. The code is a task management system with the following logic:

```
133     cin >> task_id;
134     cout << "Enter task Description: ";
135     cin.ignore(); // Clear input buffer
136     getline(cin, Description);
137     cout << "Enter task priority: ";
138     cin >> priority;
139
140     task_list.add_task(task_id, Description, priority);
141 } else if (choice == 2) {
142     task_list.view_tasks();
143 } else if (choice == 3) {
144     task_list.remove_highest_priority_task();
145 } else if (choice == 4) {
146     int task_id;
147     cout << "Enter task ID to remove From the List: ";
148     cin >> task_id;
149     task_list.remove_task_by_id(task_id);
150 } else {
151     cout << "Invalid choice. Try again." << endl;
152 }
153
154
155 return 0;
156 }
157
```

The IDE interface includes a menu bar (File, Edit, Search, View, Project, Execute, Tools, CVS, Window, Help), a toolbar, and a status bar at the bottom showing 'Line: 153, Col: 6, Sel: 0, Lines: 157, Length: 4611, Insert, Autosaved file "F:\University\Untitled1.cpp"'. The Windows taskbar at the bottom shows the time as 10:32 AM on 9/24/2024.

OUTPUT:



The screenshot shows the output of the 'Task Management System' program. It displays the menu, user input, and the resulting task list after several operations.

```
Task Management System
1. Add a new task
2. View all tasks
3. Remove the highest priority task from the list
4. Remove a task by ID
Enter your choice: 1
Enter task ID: 12
Enter task Description: Assignment second
Enter task priority: 2
Task added.

Task Management System
1. Add a new task
2. View all tasks
3. Remove the highest priority task from the list
4. Remove a task by ID
Enter your choice: 2
Tasks:
ID: 123, Description: Assignment first, Priority: 3
ID: 12, Description: Assignment second, Priority: 2

Task Management System
1. Add a new task
2. View all tasks
3. Remove the highest priority task from the list
4. Remove a task by ID
Enter your choice: 3
Removed task with ID: 123, Description: Assignment first

Task Management System
1. Add a new task
2. View all tasks
3. Remove the highest priority task from the list
4. Remove a task by ID
Enter your choice: 4
Enter task ID to remove From the List: 12
Removed task with ID: 12

Task Management System
1. Add a new task
2. View all tasks
3. Remove the highest priority task from the list
4. Remove a task by ID
Enter your choice: _
```

The output window title is 'F:\University\Untitled1.exe'. The Windows taskbar at the bottom shows the time as 10:45 AM on 9/24/2024.