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

F:\University\Untitled1.cpp - Dev-C++ 5.4.2

Untitled1.cpp

```cpp
#include <iostream>
using namespace std;
//TaskNode can add a Node in LinkedList
struct TaskNode {
public:
    int task_id;
    string Description;
    int priority;
    TaskNode* next;

    TaskNode(int id, string& desc, int prio){
        task_id=id;
         Description=desc;
         priority=prio;
          next=NULL;
    }
};
//manage the task
struct TaskList {
private:
    TaskNode* first;

public:
    TaskList(){
    first=NULL;
```

Line: 153 Col: 6 Sel: 0 Lines: 157 Length: 4611 Insert Autosaved file "F:\University\Untitled1.cpp"

F:\University\Untitled1.cpp - Dev-C++ 5.4.2

Untitled1.cpp

```cpp
    first=NULL;
    }

    // Add new Task(Node)

    void add_task(int task_id, string& Description, int priority) {
        TaskNode* new_task = new TaskNode(task_id, Description, priority);
        if (first == NULL || first->priority < priority) {
            new_task->next = first;
            first = new_task;
        } else {
            TaskNode* current = first;
            while (current->next!=NULL && current->next->priority >= priority) {
                current = current->next;
            }
            new_task->next = current->next;
            current->next = new_task;
        }
        cout << "Task added." << endl;
    }
    //Remove the Highest periority task from the LinkedList

    void remove_highest_priority_task() {
        if (first == NULL) {
            cout << "No tasks to remove." << endl;
```

Line: 153 Col: 6 Sel: 0 Lines: 157 Length: 4611 Insert Autosaved file "F:\University\Untitled1.cpp"

F:\University\Untitled1.cpp - Dev-C++ 5.4.2

File Edit Search View Project Execute Tools CVS Window Help

(globals)

Untitled1.cpp

```cpp
            cout << "No tasks to remove." << endl;
            return;
        }
        TaskNode* removed_task = first;
        first = first->next;
        cout << "Removed task with ID: " << removed_task->task_id
                << ", Description: " << removed_task->Description << endl;
        delete removed_task;
    }
        //Remove task by ID
    void remove_task_by_id(int task_id) {
        if (first == NULL) {
            cout << "No tasks to remove." << endl;
            return;
        }

        if (first->task_id == task_id) {
            TaskNode* removed_task = first;
            first = first->next;
            cout << "Removed task with ID: " << task_id << endl;
            delete removed_task;
            return;
        }

        TaskNode* current = first;
```

Compiler Resources Compile Log Debug Find Results

Line: 153 Col: 6 Sel: 0 Lines: 157 Length: 4611 Insert Autosaved file "F:\University\Untitled1.cpp"

Type here to search 30°C Clear 10:30 AM 9/24/2024

F:\University\Untitled1.cpp - Dev-C++ 5.4.2

File Edit Search View Project Execute Tools CVS Window Help

(globals)

Untitled1.cpp

```cpp
        TaskNode* current = first;
        while (current->next!=NULL && current->next->task_id != task_id) {
            current = current->next;
        }

        if (current->next == NULL) {
            cout << "Task with ID: " << task_id << " not found." << endl;
        } else {
            TaskNode* removed_task = current->next;
            current->next = current->next->next;
            cout << "Removed task with ID: " << task_id << endl;
            delete removed_task;
        }
    }
    //for veiwing All Task

    void view_tasks()  {
        if (first == NULL) {
            cout << "No tasks available." << endl;
            return;
        }
        TaskNode* current = first;
        cout << "Tasks:" << endl;
        while (current) {
            cout << "ID: " << current->task_id
```

Compiler Resources Compile Log Debug Find Results

Line: 153 Col: 6 Sel: 0 Lines: 157 Length: 4611 Insert Autosaved file "F:\University\Untitled1.cpp"

Type here to search 30°C Clear 10:31 AM 9/24/2024

F:\University\Untitled1.cpp - Dev-C++ 5.4.2

File Edit Search View Project Execute Tools CVS Window Help

(globals)

Untitled1.cpp

```cpp
            cout << "ID: " << current->task_id
                 << ", Description: " << current->Description
                 << ", Priority: " << current->priority << endl;
            current = current->next;
        }
    }
    //destructor to free up the  memory space
    ~TaskList() {
        while (first) {
            TaskNode* temp = first;
            first = first->next;
            delete temp;
        }
    }
};

int main() {
    TaskList task_list;

    while (true) {
        cout << "\nTask Management System" << endl;
        cout << "1. Add a new task" << endl;
        cout << "2. View all tasks" << endl;
        cout << "3. Remove the highest priority task from the list" << endl;
        cout << "4. Remove a task by ID" << endl;
```

Compiler Resources Compile Log Debug Find Results

Line: 153 Col: 6 Sel: 0 Lines: 157 Length: 4611 Insert Autosaved file "F:\University\Untitled1.cpp"

Type here to search 30°C Clear 10:31 AM 9/24/2024

F:\University\Untitled1.cpp - Dev-C++ 5.4.2

File Edit Search View Project Execute Tools CVS Window Help

(globals)

Untitled1.cpp

```cpp
        cout << "4. Remove a task by ID" << endl;

        int choice;
        cout << "Enter your choice: ";
        cin >> choice;

        if (choice == 1) {
            int task_id;
            string Description;
            int priority;

            cout << "Enter task ID: ";
            cin >> task_id;
            cout << "Enter task Description: ";
            cin.ignore(); // Clear input buffer
            getline(cin, Description);
            cout << "Enter task priority: ";
            cin >> priority;

            task_list.add_task(task_id, Description, priority);
        } else if (choice == 2) {
            task_list.view_tasks();
        } else if (choice == 3) {
            task_list.remove_highest_priority_task();
        } else if (choice == 4) {
```

Compiler Resources Compile Log Debug Find Results

Line: 153 Col: 6 Sel: 0 Lines: 157 Length: 4611 Insert Autosaved file "F:\University\Untitled1.cpp"

Type here to search 30°C Clear 10:31 AM 9/24/2024

F:\University\Untitled1.cpp - Dev-C++ 5.4.2

```
            cin >> task_id;
            cout << "Enter task Description: ";
            cin.ignore(); // Clear input buffer
            getline(cin, Description);
            cout << "Enter task priority: ";
            cin >> priority;

            task_list.add_task(task_id, Description, priority);
        } else if (choice == 2) {
            task_list.view_tasks();
        } else if (choice == 3) {
            task_list.remove_highest_priority_task();
        } else if (choice == 4) {
            int task_id;
            cout << "Enter task ID to remove From the List: ";
            cin >> task_id;
            task_list.remove_task_by_id(task_id);
        }else {
           cout << "Invalid choice. Try again." <<  endl;
        }
    }

    return 0;
}
```

OUTPUT:

F:\University\Untitled1.exe

```
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:
```