Assignment 4

Code :

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

class Node {
public:
    string task_name;
    int priority;
    int exe_time;
    Node* next;

    Node(string t, int p, int e) {
        task_name = t;
        priority = p;
        exe_time = e;
        next = NULL;
    }

    void display() {
        cout << "Task: " << task_name
             << ", Priority: " << priority
             << ", Execution Time: " << exe_time << " ms" << endl;
    }
};

void insertByPriority(Node*& head, string tn, int p, int e) {
    Node* temp = new Node(tn, p, e);

    if (head == NULL || head->priority < p) {
        temp->next = head;
        head = temp;
    } else {
        Node* current = head;
        while (current->next != NULL && current->next->priority >= p) {
```

```
            current = current->next;
        }
        temp->next = current->next;
        current->next = temp;
    }
}

// Insert node in sorted list by execution time
void sortedInsert(Node*& sortedHead, Node* newNode) {
    if (sortedHead == NULL || sortedHead->exe_time >= newNode->exe_time) {
        newNode->next = sortedHead;
        sortedHead = newNode;
    } else {
        Node* temp = sortedHead;
        while (temp->next != NULL && temp->next->exe_time < newNode->exe_time) {
            temp = temp->next;
        }
        newNode->next = temp->next;
        temp->next = newNode;
    }
}

Node* sortByExecutionTime(Node* head) {
    Node* sorted = NULL;
    Node* current = head;

    while (current != NULL) {
        Node* next = current->next;  // store next node
        current->next = NULL;        // detach current node
        sortedInsert(sorted, current);
        current = next;
    }
    return sorted;
}

int main() {
    int n;
    Node* head = NULL;

    cout << "Enter number of tasks: ";
```

```
    cin >> n;

    for (int i = 0; i < n; i++) {
        string tn;
        int p, e;

        cout << "\nEnter Task Name: ";
        cin >> tn;
        cout << "Enter Priority: ";
        cin >> p;
        cout << "Enter Execution Time: ";
        cin >> e;

        insertByPriority(head, tn, p, e);
    }

    cout << "\nScheduled Tasks (Highest Priority First):\n\n";
    Node* current = head;
    while (current != NULL) {
        current->display();
        current = current->next;
    }

    head = sortByExecutionTime(head);

    cout << "\nExecuting Tasks according to execution time:\n\n";
    current = head;
    while (current != NULL) {
        cout << "Executing Task '" << current->task_name
             << "' : " << current->exe_time << " ms...\n";
        current = current->next;
    }

    return 0;
}
```

Output :

```
Enter Priority: 2
Enter Execution Time: 455

Enter Task Name: run
Enter Priority: 4
Enter Execution Time: 222

Enter Task Name: learn
Enter Priority: 1
Enter Execution Time: 666

Scheduled Tasks (Highest Priority First):

Task: run, Priority: 4, Execution Time: 222 ms
Task: work, Priority: 3, Execution Time: 243 ms
Task: clean, Priority: 2, Execution Time: 455 ms
Task: learn, Priority: 1, Execution Time: 666 ms

Executing Tasks according to execution time:

Executing Task 'run' : 222 ms...
Executing Task 'work' : 243 ms...
Executing Task 'clean' : 455 ms...
Executing Task 'learn' : 666 ms...


------------------
(program exited with code: 0)

Press any key to continue . . .
```