Meenakshi Shankar
S.R No. 22400
M.Tech EPD

EMBEDDED SYSTEM DESIGN (E3-257)

LAB ASSIGNMENT - 1

Explanation of Code and Outputs

In the zip is attached tcb_ll.c which is the C code implementation of a task scheduler, where tasks are represented by a linked list structure (`Task`). The program reads task information from a file, processes and manipulates these tasks, and allows certain operations on the tasks, such as adding, suspending, and deleting tasks.

Data Structures:

1. **Task Structure** :

- Represents a task with the following attributes:
  - `**taskId**` : Unique identifier for the task.
  - `**taskPriority**` : Priority level of the task.
  - `**taskBurst**` : The time required for the task to execute.
  - `**taskRemain**` : Remaining execution time for the task.
  - `**taskeventId**` : Event identifier associated with the task.
  - `**state**` : Current state of the task (e.g., "Ready," "Wait," "Running," "Done").
  - `**next**` : Pointer to the next task in the linked list.

2**. Node Structure (`struct Node`):**

- Used to create a linked list of tasks.
- Contains:
  - `**task**` : Pointer to a task (`Task*`).
  - `**nex**t` : Pointer to the next node in the linked list (`Node*`).

Meenakshi Shankar
S.R No. 22400
M.Tech EPD

Functions:

1. `createTask(Task** head, ...)`:

- Creates a new task and adds it to the end of the linked list.
- Handles memory allocation and initialization of task attributes.

2. `**insertSorted(Task** head, Task* newTask)**`:

- Inserts a task into a sorted linked list based on taskPriority.
- Maintains the order of tasks based on their priority.

3. `**sortLinkedList(Task** head)**`:

- Sorts the linked list of tasks based on their priorities using the `insertSorted` function.

4. `**addNodeByPriority(Task** head, Task* newTask)**`:

- Adds a task to the linked list while maintaining the sorted order based on taskPriority.

5. `**createNode(Task* task)**`:

- Creates a new node with a given task.

6. `**insertNodeAtEnd(Node** head, Task* task)`:**

- Inserts a node at the end of a linked list of nodes.

7. `**separateReadyAndWaitLists(Task* head, Node** readyList, Node** waitList, Node** deleteList)`:**

- Separates tasks into "Ready" and "Wait" lists based on their states.
- Also includes a "New" list for tasks with state "New."

8. `**updateReadyList(Task* head, Node** readyList, int count)**`:

- Updates the "Ready" list by moving a specified number of tasks from the head of the original list.

Meenakshi Shankar
S.R No. 22400
M.Tech EPD

9. `**deleteNodeById(Node** readyList, int taskId)`:

- Deletes a node with a specific taskId from the "Ready" list.

10. `**printLinkedList(Node* head)`:

- Prints the linked list of tasks.

`main` Function:

1. **File Reading**:

- Opens a file containing task information.

- Reads task details from the file and creates tasks using the `createTask` function.

2. **Task List Manipulation**:

- Initializes lists (`readyList`, `waitList`, `deleteList`) to organize tasks based on their states.

- Performs various operations on tasks based on user input:

  - Add a new task.

  - Execute tasks based on a time slice.

  - Print "Ready" and "Wait" lists.

  - Print all tasks after sorting by priority.

  - Suspend, resume, wait for, and delete tasks.

Meenakshi Shankar
S.R No. 22400
M.Tech EPD

Simulation:

- Simulates a basic task scheduling scenario.

- Updates task states, moves tasks between lists, and prints the state of the system.

Operations in main:

→'n': Adds a new task to the system.

Initial List after reading from .txt file

```
PS C:\Users\meena\OneDrive\Desktop\cpp_practise> cd "c:\Users\meena\OneDrive\Desktop\cpp_practise\" ; if ($?) { gcc tempCodeRunner
File.c -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
[Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 1, Priority: 1, Burst: 5, Remain: 5, Event Id: 456, State: Running]
[Task ID: 4, Priority: 5, Burst: 6, Remain: 6, Event Id: 789, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 6, Priority: 2, Burst: 7, Remain: 7, Event Id: 222, State: Ready]
[Task ID: 7, Priority: 1, Burst: 5, Remain: 5, Event Id: 333, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
Enter Function :
```

Trying to add a task of already existing Id:

```
Enter Function : n 2
Sorry ! Task Id already exixts
```

Adding a new Task and Printing Ready and Wait List:

```
0Enter Function : n 11
1Enter Task Priority : 4
Enter Task State : Ready
Enter Task Burst Time : 5
Enter Task Remain Time : 5
Enter Event Id : 121
```

```
Enter Function : R
Ready List: [Task ID: 6, Priority: 2, Burst: 7, Remain: 7, Event Id: 222, State: Ready]
[Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 11, Priority: 4, Burst: 5, Remain: 5, Event Id: 121, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
[Task ID: 4, Priority: 5, Burst: 6, Remain: 6, Event Id: 789, State: Ready]
Wait List: [Task ID: 7, Priority: 1, Burst: 5, Remain: 5, Event Id: 333, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
Enter Function :
```

Meenakshi Shankar
S.R No. 22400
M.Tech EPD

→'d': Deletes a task by taskId.

After “d 11”. 'R': Prints the current "Ready" and "Wait" lists.

```
Enter Function : R
Ready List: [Task ID: 6, Priority: 2, Burst: 7, Remain: 7, Event Id: 222, State: Ready]
[Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
Wait List: [Task ID: 7, Priority: 1, Burst: 5, Remain: 5, Event Id: 333, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
Enter Function :
```

After deleting all tasks in the ready queue a default task is set available. Here it is task id 9 which was not already set in the Linked List.

```
Enter Function : R
Ready List: [Task ID: 6, Priority: 2, Burst: 7, Remain: 7, Event Id: 222, State: Ready]
[Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
Wait List: [Task ID: 7, Priority: 1, Burst: 5, Remain: 5, Event Id: 333, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
Enter Function : d 6
[Task ID: 6, Priority: 2, Burst: 7, Remain: 7, Event Id: 222, State: Ready]
[Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
2 [Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]

3
Ready
2
Ready
[Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
Enter Function : d 3
[Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
1 [Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]

2
Ready
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
Enter Function : d 2
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
0 Ready List: [Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]

Enter Function : R
Ready List: [Task ID: 9, Priority: 1, Burst: 5, Remain: 5, Event Id: 999, State: Ready]
Wait List: [Task ID: 7, Priority: 1, Burst: 5, Remain: 5, Event Id: 333, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
Enter Function :
```

'E': Executes tasks based on a time slice.

'B': Prints all tasks after sorting by priority.

→'s', 'e', 'w': Suspends, resumes, and waits for a specific task, respectively.

After adding two wait state tasks to waitlist with same event_id:

Meenakshi Shankar
S.R No. 22400
M.Tech EPD

```
Ready List: [Task ID: 9, Priority: 1, Burst: 5, Remain: 5, Event Id: 999, State: Ready]
Wait List: [Task ID: 7, Priority: 1, Burst: 5, Remain: 5, Event Id: 333, State: Wait]
[Task ID: 12, Priority: 2, Burst: 5, Remain: 5, Event Id: 343, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
[Task ID: 11, Priority: 5, Burst: 5, Remain: 5, Event Id: 343, State: Wait]
```

“e 343” will put both task 11 and 12 to readyList

```
Enter Function : e 343
Ready List: [Task ID: 9, Priority: 1, Burst: 5, Remain: 5, Event Id: 999, State: Ready]
[Task ID: 12, Priority: 2, Burst: 5, Remain: 5, Event Id: 343, State: Ready]
[Task ID: 11, Priority: 5, Burst: 5, Remain: 5, Event Id: 343, State: Ready]
Wait List: [Task ID: 7, Priority: 1, Burst: 5, Remain: 5, Event Id: 333, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
Enter Function :
```

Again, from the initial list if we execute “w 4 500” we move task 4 of readyList to waitlist for any random eventTrigger

```
Enter Function : R
Ready List: [Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 4, Priority: 5, Burst: 6, Remain: 6, Event Id: 789, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
[Task ID: 6, Priority: 2, Burst: 7, Remain: 7, Event Id: 222, State: Ready]
Wait List: [Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 7, Priority: 1, Burst: 5, Remain: 5, Event Id: 333, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
Enter Function : w 4 500
Ready List: [Task ID: 6, Priority: 2, Burst: 7, Remain: 7, Event Id: 222, State: Ready]
[Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
Wait List: [Task ID: 7, Priority: 1, Burst: 5, Remain: 5, Event Id: 333, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
[Task ID: 4, Priority: 5, Burst: 6, Remain: 6, Event Id: 789, State: Wait]
Enter Function :
```

Executing “s 456” will remove the current running task to waitlist and next Priority task goes to executing

Meenakshi Shankar
S.R No. 22400
M.Tech EPD

```
Enter Function : s 456
Checking
Checking
Checking
Checking
Checking
Checking
Checking
Ready List: [Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 6, Priority: 2, Burst: 7, Remain: 7, Event Id: 222, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
Wait List: [Task ID: 1, Priority: 1, Burst: 5, Remain: 5, Event Id: 456, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 4, Priority: 5, Burst: 6, Remain: 6, Event Id: 789, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
Enter Function :
```

From the previous figure we note next in priority task was Id 6 in readyList so it gets Running state.

Using priority based scheduling algorithm and time slice if we execute E we can see sequence of the tasks being executed in readyList i.e. 3->6->2

```
Ready List: [Task ID: 3, Priority: 2, Burst: 7, Remain: 7, Event Id: 123, State: Ready]
[Task ID: 6, Priority: 2, Burst: 7, Remain: 7, Event Id: 222, State: Ready]
[Task ID: 2, Priority: 4, Burst: 4, Remain: 4, Event Id: 935, State: Ready]
Wait List: [Task ID: 1, Priority: 1, Burst: 5, Remain: 5, Event Id: 456, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 4, Priority: 5, Burst: 6, Remain: 6, Event Id: 789, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
Enter Function : E
Task:1
Task:7
Task:3
Running
Running
Running
Done
Task:6
Running
Running
Running
Done
Task:5
Task:10
Task:2
Running
Running
Done
Task:4
Task:8
Ready List: Wait List: [Task ID: 1, Priority: 1, Burst: 5, Remain: 5, Event Id: 456, State: Wait]
[Task ID: 5, Priority: 3, Burst: 7, Remain: 7, Event Id: 345, State: Wait]
[Task ID: 10, Priority: 3, Burst: 7, Remain: 7, Event Id: 666, State: Wait]
[Task ID: 4, Priority: 5, Burst: 6, Remain: 6, Event Id: 789, State: Wait]
[Task ID: 8, Priority: 5, Burst: 6, Remain: 6, Event Id: 444, State: Wait]
```

Meenakshi Shankar
S.R No. 22400
M.Tech EPD

Conclusion

- The code uses linked lists to manage tasks efficiently.

- The task states ("Ready," "Wait," "Running," "Done") are used to simulate a task's lifecycle.

- The program provides a simple interactive interface for users to perform various operations on tasks.