Task Scheduler using Min Heap in C

A Real-Time Task Management System with Auto-Completion

Introduction

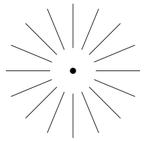
- A Task Scheduler helps manage and execute tasks efficiently.
- This project implements a real-time scheduler that:
- Adds and manages tasks.
- Automatically completes tasks after a time limit.
- Allows updating, canceling, or marking tasks manually.
- The scheduling logic is based on a Min Heap data structure.

Objective

1. Implement an efficient task scheduling system in C.
2. Handle automatic task completion based on real-time tracking.
3. Support manual operations like update, cancel, and serve.

Features

task scheduler



Add new tasks with ID, time, and description.

View all tasks with their status (Pending/Completed).

Automatically mark tasks as completed after a time.

Manually mark or cancel tasks.

Update submission time for any existing task.

Menu-driven, real-time console interface.

Why Min Heap?

Operation	Time Complexity	Description
Insertion	O(log n)	Add new task
Deletion	O(log n)	Cancel task
Access smallest	O(1)	Next earliest task
Search (via map)	O(1)	Find any task by ID

Min Heap ensures tasks are managed in increasing order of submission time.

System Architecture

01

main.c → User interface & menu

02

scheduler.c → Heap operations & task logic.

03

scheduler.h → Structure & function declarations.

Data Structures Used

Task Structure

```
typedef struct {
    int taskId;
    int submissionTime;
    char details[100];
    int completed;
    time_t startTime;
} Task;
```

Min Heap Structure

```
typedef struct {
    Task **arr;
    int size;
    int capacity;
    int *indexMap;
} MinHeap
```

Working Principle

Step 1: Add a task → stored in heap.

Step 2: Timer starts (startTime).

Step 3: Every few seconds, system checks elapsed time.

Step 4: If elapsed ≥ submissionTime → marked Completed.

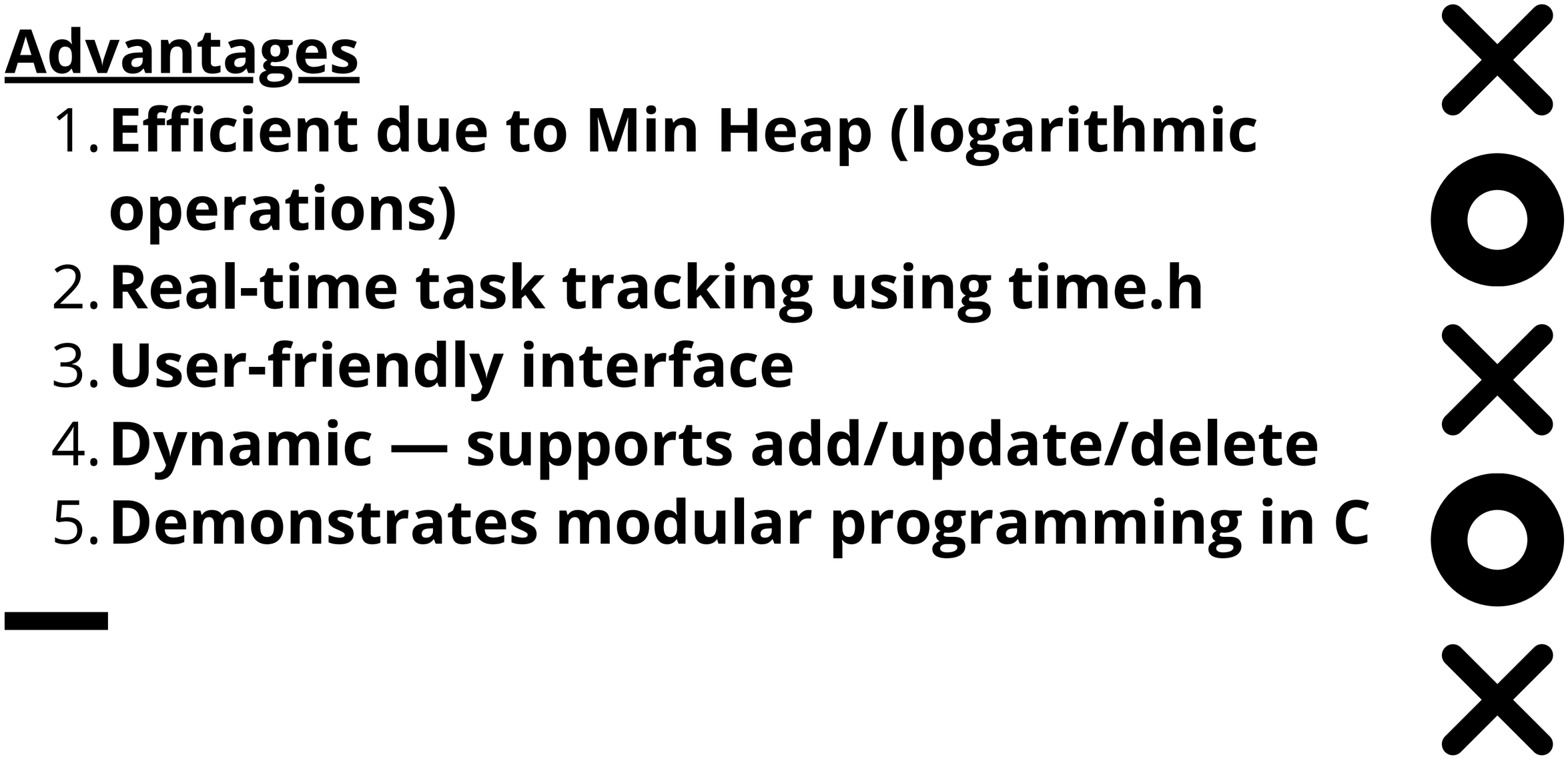
note : User can also manually complete, update, or cancel tasks.

Major Functions

Function	Purpose
addTask()	Add new task
checkAndUpdateTasks()	Auto-complete tasks
serveTask()	Mark task complete manually
cancelTask()	Delete task from scheduler
updateSubmissionTime()	Modify a task’s time
printTasks()	Display current tasks

Advantages

1. **Efficient due to Min Heap (logarithmic operations)**
2. **Real-time task tracking using time.h**
3. **User-friendly interface**
4. **Dynamic — supports add/update/delete**
5. **Demonstrates modular programming in C**



Division of Work

Team Member	Contribution
Mitesh	Worked on the modules, including addTask, heapify functions, and overall debugging & testing also created the system architecture
Mayank	Developed checkAndUpdateTasks, updateSubmissionTime, and handled the main.c file logic.
Muaz	Implemented printTasks, serveTask, and cancelTask functions and created the PowerPoint presentation.

Conclusion

01 The project successfully demonstrates:

03 Real-time scheduling

04 Heap-based task management

05 Efficient time-based automation

06 Provides a strong foundation for more complex schedulers or OS-level task systems.