Java

```java
class TaskManager {

    public TaskManager(List<List<Integer>> tasks) {

    }

    public void add(int userId, int taskId, int priority) {

    }

    public void edit(int taskId, int newPriority) {

    }

    public void rmv(int taskId) {

    }

    public int execTop() {

    }
}

/**
 * Your TaskManager object will be instantiated and called as such:
 * TaskManager obj = new TaskManager(tasks);
 * obj.add(userId,taskId,priority);
 * obj.edit(taskId,newPriority);
 * obj.rmv(taskId);
 * int param_4 = obj.execTop();
```

```
 */
--------------------------------------------------------------------------
```

JavaScript

```javascript
/**
 * @param {number[][]} tasks
 */
var TaskManager = function(tasks) {

};

/**
 * @param {number} userId
 * @param {number} taskId
 * @param {number} priority
 * @return {void}
 */
TaskManager.prototype.add = function(userId, taskId, priority) {

};

/**
 * @param {number} taskId
 * @param {number} newPriority
 * @return {void}
 */
TaskManager.prototype.edit = function(taskId, newPriority) {

};

/**
```

```
 * @param {number} taskId
 * @return {void}
 */
TaskManager.prototype.rmv = function(taskId) {

};

/**
 * @return {number}
 */
TaskManager.prototype.execTop = function() {

};

/**
 * Your TaskManager object will be instantiated and called as such:
 * var obj = new TaskManager(tasks)
 * obj.add(userId,taskId,priority)
 * obj.edit(taskId,newPriority)
 * obj.rmv(taskId)
 * var param_4 = obj.execTop()
 */
```

TypeScript

```
class TaskManager {
    constructor(tasks: number[][]) {

    }

    add(userId: number, taskId: number, priority: number): void {
```

```
    }

    edit(taskId: number, newPriority: number): void {

    }

    rmv(taskId: number): void {

    }

    execTop(): number {

    }
}

/**
 * Your TaskManager object will be instantiated and called as such:
 * var obj = new TaskManager(tasks)
 * obj.add(userId,taskId,priority)
 * obj.edit(taskId,newPriority)
 * obj.rmv(taskId)
 * var param_4 = obj.execTop()
 */
```

C++

```
class TaskManager {
public:
    TaskManager(vector<vector<int>>& tasks) {
```

```
    }

    void add(int userId, int taskId, int priority) {

    }

    void edit(int taskId, int newPriority) {

    }

    void rmv(int taskId) {

    }

    int execTop() {

    }
};

/**
 * Your TaskManager object will be instantiated and called as such:
 * TaskManager* obj = new TaskManager(tasks);
 * obj->add(userId,taskId,priority);
 * obj->edit(taskId,newPriority);
 * obj->rmv(taskId);
 * int param_4 = obj->execTop();
 */
```

C#

```
public class TaskManager {
```

```
    public TaskManager(IList<IList<int>> tasks) {

    }

    public void Add(int userId, int taskId, int priority) {

    }

    public void Edit(int taskId, int newPriority) {

    }

    public void Rmv(int taskId) {

    }

    public int ExecTop() {

    }
}

/**
 * Your TaskManager object will be instantiated and called as such:
 * TaskManager obj = new TaskManager(tasks);
 * obj.Add(userId,taskId,priority);
 * obj.Edit(taskId,newPriority);
 * obj.Rmv(taskId);
 * int param_4 = obj.ExecTop();
 */
```

Kotlin

```kotlin
class TaskManager(tasks: List<List<Int>>) {

    fun add(userId: Int, taskId: Int, priority: Int) {

    }

    fun edit(taskId: Int, newPriority: Int) {

    }

    fun rmv(taskId: Int) {

    }

    fun execTop(): Int {

    }

}

/**
 * Your TaskManager object will be instantiated and called as such:
 * var obj = TaskManager(tasks)
 * obj.add(userId,taskId,priority)
 * obj.edit(taskId,newPriority)
 * obj.rmv(taskId)
 * var param_4 = obj.execTop()
 */
```

Go

```go
type TaskManager struct {

}

func Constructor(tasks [][]int) TaskManager {

}

func (this *TaskManager) Add(userId int, taskId int, priority int)  {

}

func (this *TaskManager) Edit(taskId int, newPriority int)  {

}

func (this *TaskManager) Rmv(taskId int)  {

}

func (this *TaskManager) ExecTop() int {

}

/**
```

```
 * Your TaskManager object will be instantiated and called as such:
 * obj := Constructor(tasks);
 * obj.Add(userId,taskId,priority);
 * obj.Edit(taskId,newPriority);
 * obj.Rmv(taskId);
 * param_4 := obj.ExecTop();
 */
```
