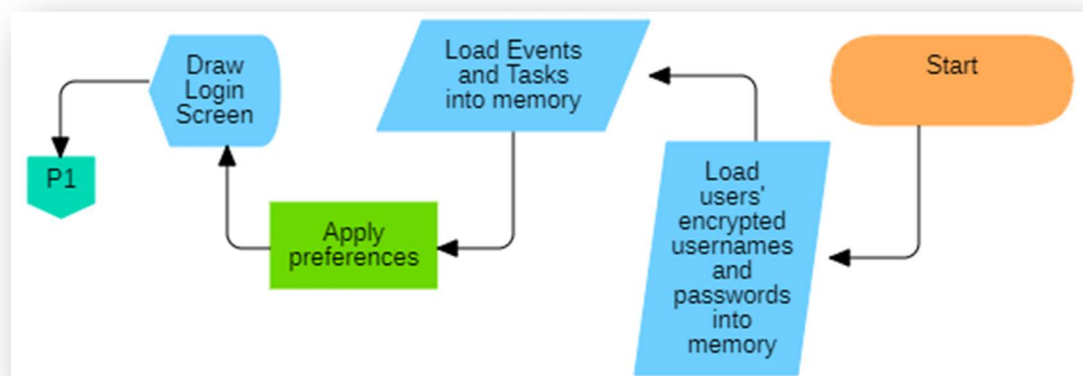


## Criterion B:

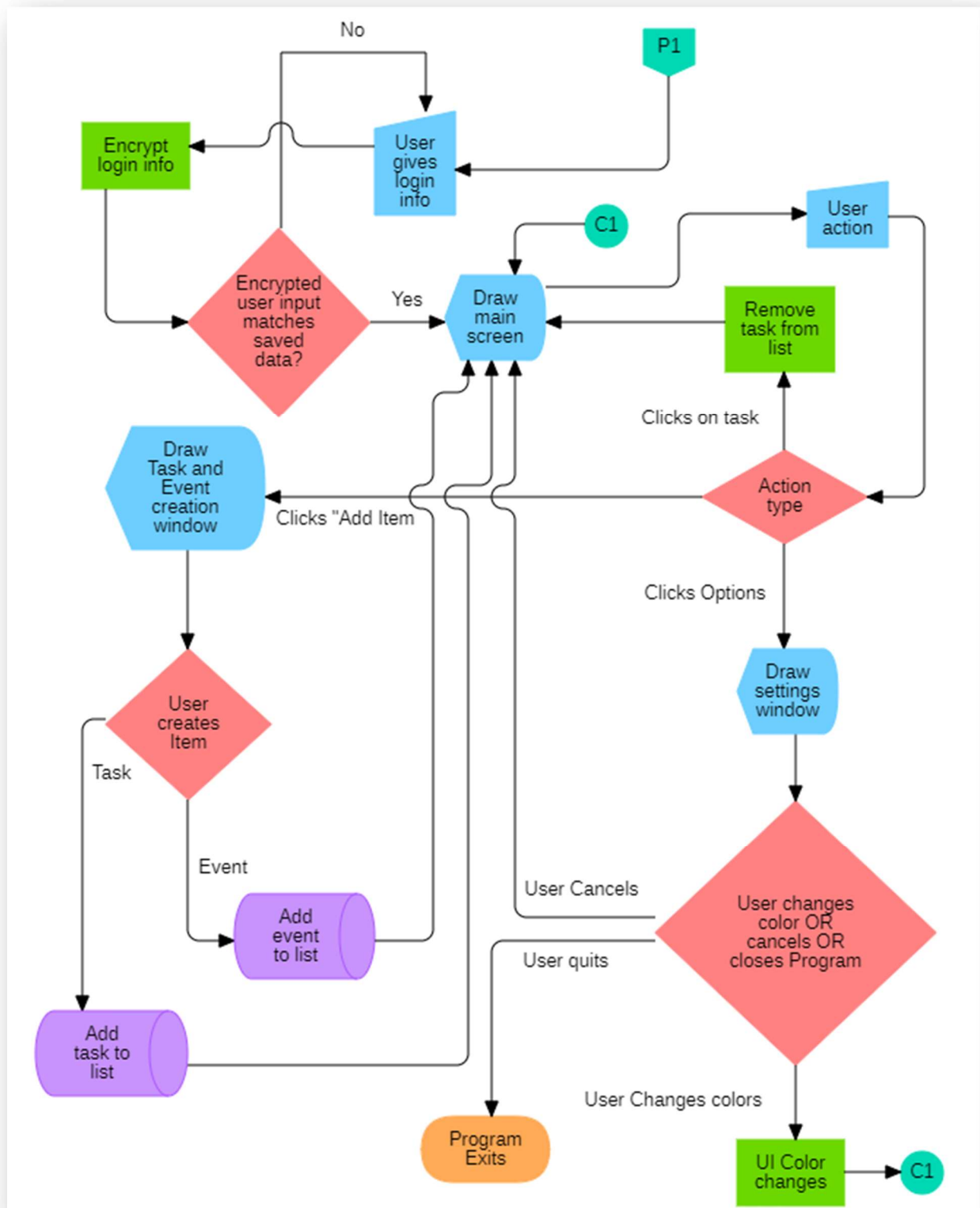
### Tests:

Criteria	Test	Result
1. Can the client add and remove tasks and events with ease?	Check that tasks and events can be added with ease.	Pressing the add button brings up a menu where tasks are added to the task list.
2. Can the client add due dates to tasks?	Check that tasks can have due dates added to them.	Clients can be notified when tasks are due.
3. Can the client add reminders to events and tasks?	Check that events and tasks can be given reminders.	In the “Add item” dialog there is a place to add reminders.
4. Can the client add priority levels to their events and tasks?	Check that tasks can be given priority levels.	In the “Add item” dialog there is a JSlider to choose priority levels.
5. Can the client see completed tasks?	Check that completed tasks are still accessible to the client.	Clients can press a button and be shown completed tasks.
6. Can the client change the proportion of their UI used by the calendar and task list?	Check that the calendar and task lists are resizable.	The JSplitPane has a built in resize bar.
7. Can the client change the colors of their UI?	Check that UI colors are user-mutable.	The “Options” dialog has color settings.

### Flow Charts



## Flow Charts



## Planned Graphics

Productivity App

Task 1

Task 2

Task 3

7 am

8 am

9 am

10 am

11 am

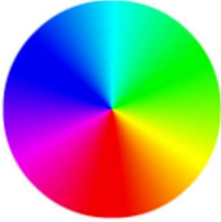
12 pm

1 pm

2 pm

Event

Choose background color:



Confirm

Cancel

Create new event:

Name

Time

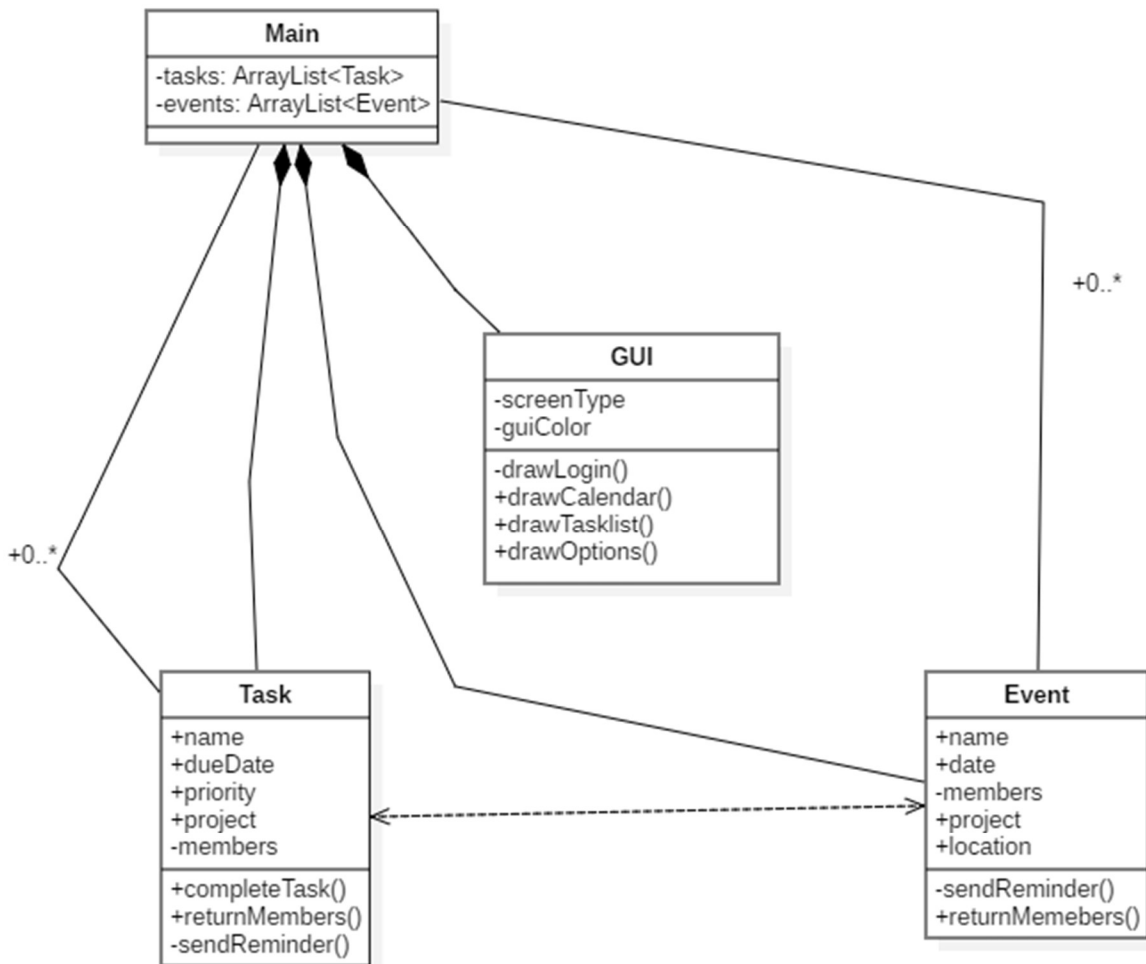
+ Add User

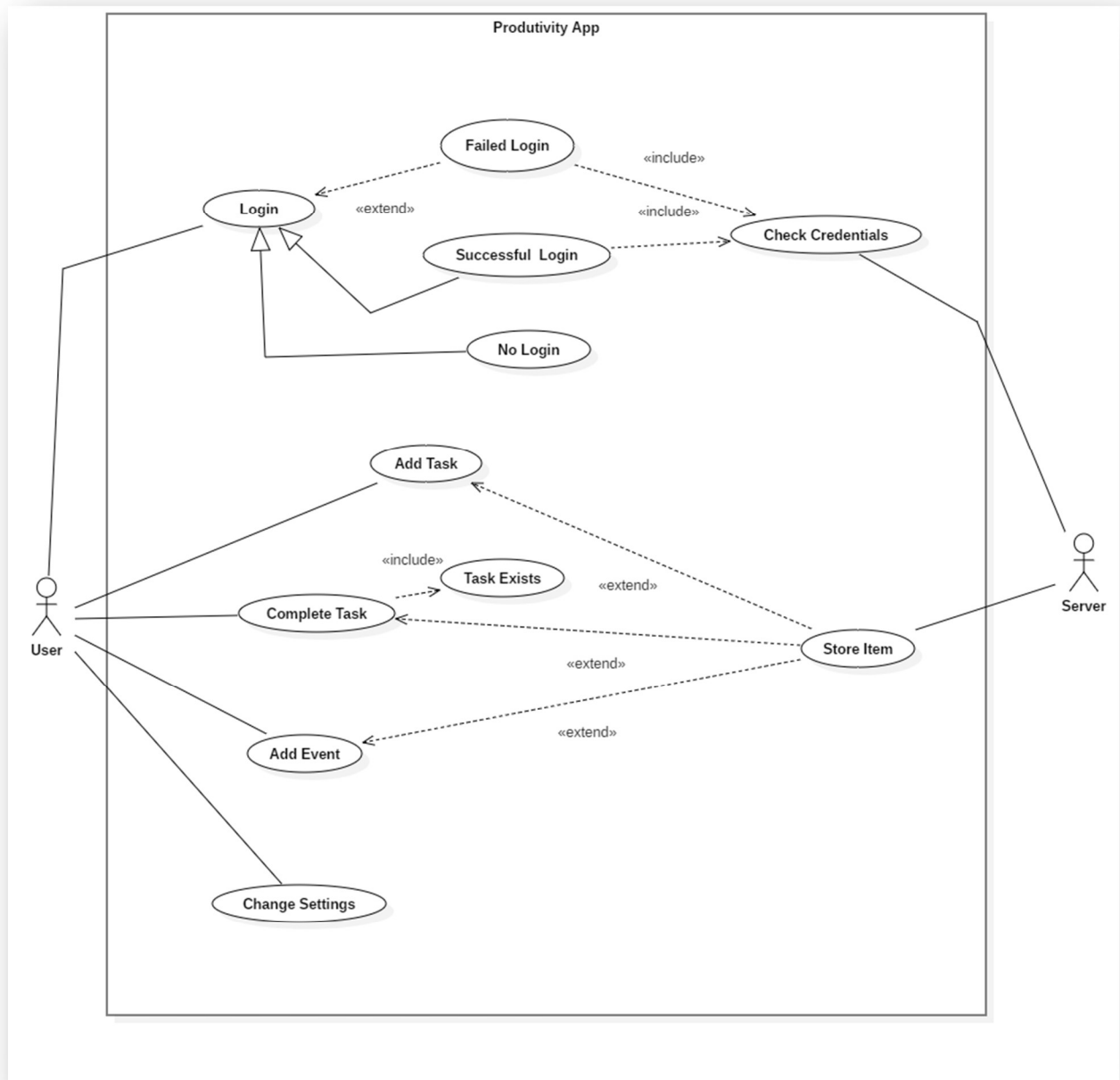
+ Add Reminder

Confirm

Cancel

## Planning Documents



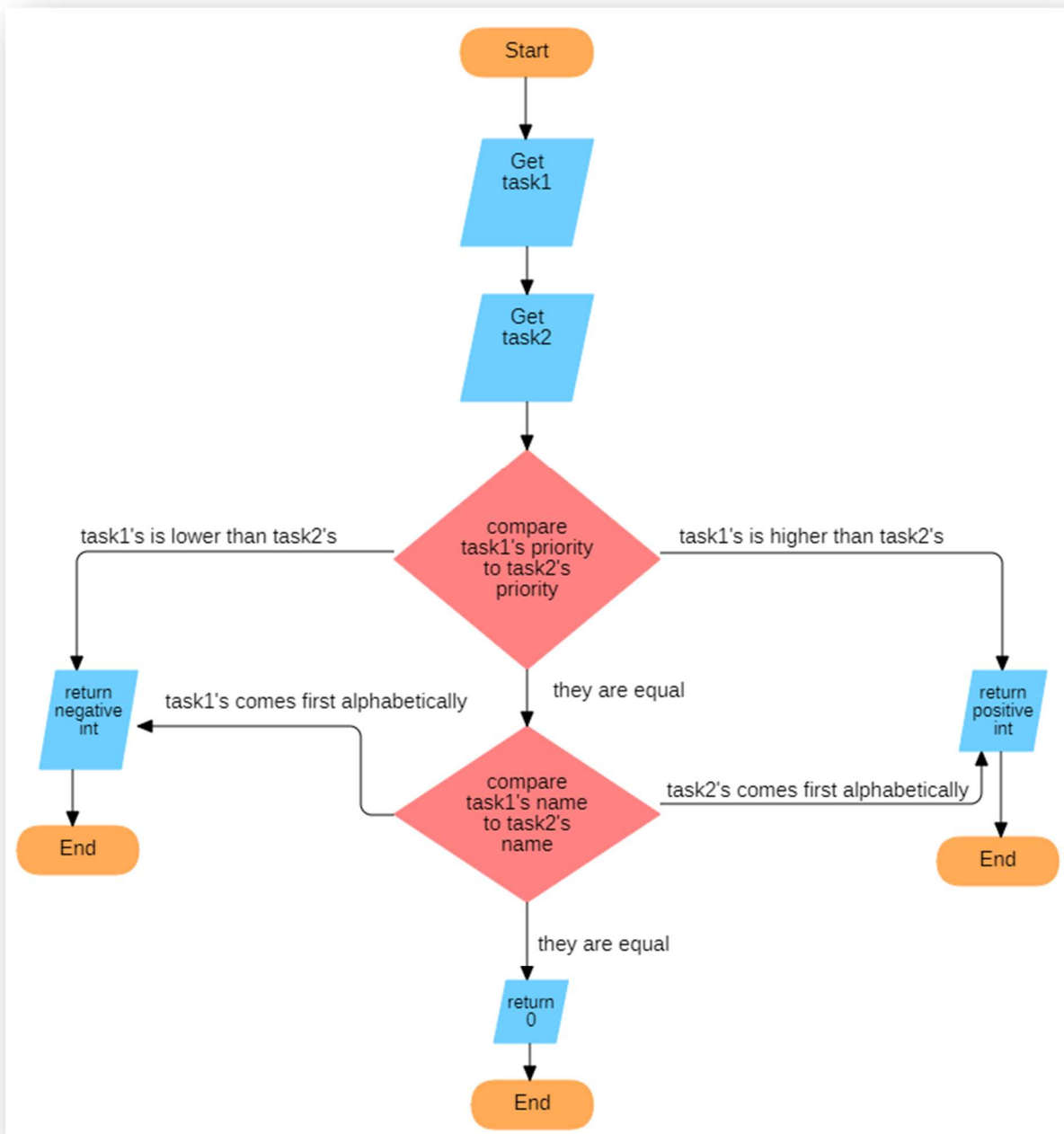


## Key Algorithms

### TaskComparator

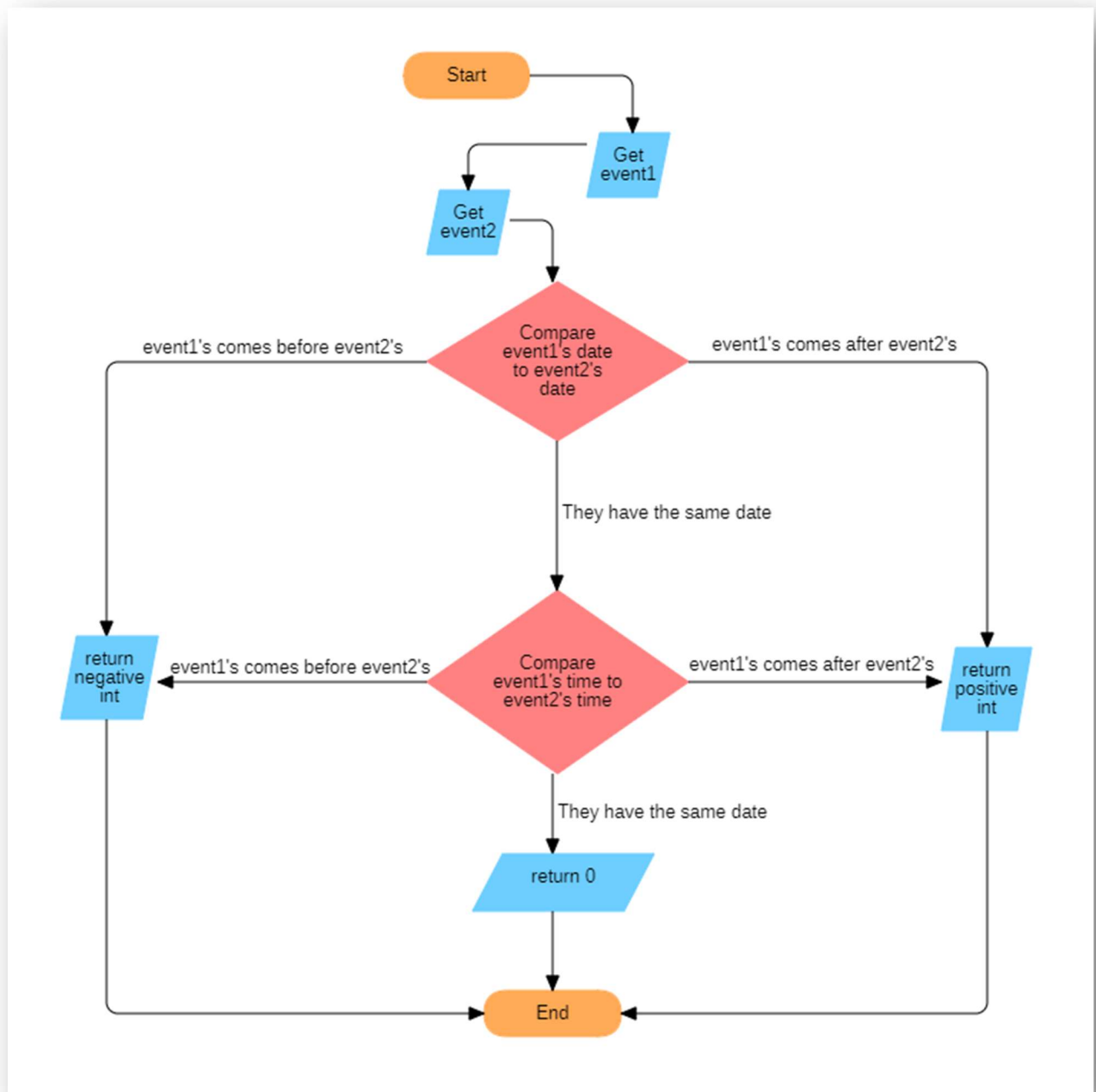
I had to implement a comparator for my custom “Task” class so I could order my tasks by priority in Java Swing. This comparator compares task priorities and then, as a last resort, to settle differences, task names.

## TaskComparator



## EventComparator

I had to implement a comparator for my custom “Event” class so I could order my events by date in Java Swing. This comparator compares event dates and then, as a last resort, to settle differences, event times. This ensures the proper ordering of events.



## TaskCellRenderer

In order to draw tasks in Java swing, I had to create a cell renderer. This extends JLabel so I can use the rendering methods and properties that exist in the JLabel class. This renderer essentially just takes the Task's name and displays it in a JLabel. However, it does have an important functionality where, when the cell is selected, that Task is removed from the JList model (this removes it from the list of drawn tasks, but not from the task list or drive).

