How to run the program:
Set run configurations from the main…(String[] args) in Class Main. Here you can create a new StudyPlanner and run methods. To create the GUI, use the created controller and call controller.startApplication().

N.b. Level 3 uses JavaFX. My design is that the controller class creates and instance of both the model and the view, manipulates the model, and updates the view. Therefore, the implemented setGUI() method was not used in the StudyPlanner Class.

Test Name	Test Goal	Expected Outcome
PlannerLevel1Test.shorterStudyBlockCreated	Test whether a shorter study block is created towards the end of a topic's duration	The planner should create a shorter study block if is not enough time in the topics duration for a full study block.
PlannerLevel1Test.manipulateTopicArray()	Manipulate the size of the array and make sure it is being accessed and updated properly whether it is being added to or deleted from	The planner should test exceptions of the same topic being added, and also test the size of the array.
PlannerLevel1Test.planWithoutBreak()	Because 'breaks' are seen as events, I test whether these events are added or not by looking at the size.	The plan's size should be exactly the amount of blocks that are created from the duration.
PlannerLevel2Test.minimumStudyBlockBeforeEndOfDay()	Test whether a minimum study block is enforced before the end of a day	The planner should enforce the minimum block length at the end of the day even if there is time left to study in the topic
PlannerLevel2Test.doEventsOverlap()	Test whether an event overlaps with another.	Because there is an algorithm to scan over every minutes of event in the calendar events arrayList, it should easily be able to compare and see if any events coincide with one another.
PlannerLevel2Test.testEventInsideDayWindow()	Test whether the event is in the day window, and whether a study block is created before and after. i.e. the event interrupts the study block and the study block is split up.	Study blocks should be added in and around events. This test hopes to add a minimum study block before an event and then resume like normal again afterwards.
PlannerLevel3Test.testPlannersTopicsUpdatestheObservableArrayList	Test whether the observable array list (the information in the table) is properly updates when the model is manipulated. In this case, I use the topics list to test this.	When a controller is created, it creates a model and view. Anytime the controller manipulates the model, the model sends messages back to the controller. Then the controller notifies the view and updates the view's observable fx arraylist.

Test Name	Test Goal	Expected Outcome
PlannerLevel3test.testNullableClasses()	Test whether the model and the view are initialised properly when the controller is created.	The controller should create the view and the model, and control actions that occur in both.