Projekt 7:

Woche 1: Gruppe gefunden,Treffen vereinbart.
Woche 2: Erste Vorbesprechung:

public class Hauptklasse{} public static void main(String[]args){} -startet die GUI

public class Uebungsaufgabewahl{} - Auswahlkasten für Übungsaufgaben, sortiert durch Nummerierung

public class Textfelder{} -Titel,Klassenname und Kommentar über Textfeld -Rotes und Grünes Textfeld für Test und Programm -Deaktivierung des entsprechenden Textfeldes -Wechsel der Scenes -Alert box falls Code inkorrekt -Möglichkeit

Aufteilung der Aufgaben. Erster Bau-Plan.

Woche 2:

Zweites Treffen:
Umgang mit GitHub verbessert , Präsentation der Ergebnisse Schritt 2. und 3 wurden zusammengefügt und fertiggestellt. Wir haben Teil Rot und Grün untereinander aufgeteilt und kamen zu der Erkenntnis dass beide Schritte sehr analog sind. Bei unserem Wöchentlichen Treffen haben wir die einzelnen Klassen zusammengefügt. Folgende Klassen und Methoden wurden implementiert:

public class Main extends Application {}

enthält die Methoden:

public void start(Stage primaryStage) throws Exception{}

-beinhält die GUI

public static void compiler(String ext,String exxt){}

-erstellt zwei CompilationUnits für jedes Textfeld.
Diese werden in einen CompilationUnit-Array gepackt um damit den InternalCompiler zu initialisieren. Anschliessend wird der Internalcompiler mittels der Methode compileAndRunTests() ausgeführt. Wir erzeugen neue Compiler und Test Results. Wenn die Anzahl der fehlerhaften Tests 1 ist und das Programm Kompilierungsfehler hat so wird das Testfeld deaktiviert. Die Methode gibt in der Konsole die Anzahl der erfolgreichen/fehlgeschlagenen Tests an und die Methodennamen der Tests die fehlgeschlagen sind (für Debugging)

public static void zuRot(){}

-aktiviert das Textfeld für die Tests
-deaktiviert das Textfeld für die Klasse
-deaktiviert den Button Rot
-deaktiviert den Button Refactor

public static void zuGruen(){}

-deaktiviert das Textfeld für die Tests
-aktiviert das Textfeld für die Klasse
-deaktiviert den Button Gruen
-deaktiviert den Button Refactor

public static void zuRefactor(){}

-aktiviert das Textfeld für die Tests
-deaktiviert das Textfeld für die Klasse
-deaktiviert den Button Rot
-deaktiviert den Button Refactor

public String refactorerror(CompilationUnit[] cs,CompilerResult comres){}

-bekommt die in compiler erzeugten Compilation Units und Compiler Results und printed sie als Strings aus

public class LoadnSave{ }

enthält die Mehtoden

public static void load(String FileName,TextArea txtarea,boolean isTest){ }

Prüft ob die ladende Datei ein Test oder eine Klasse ist. Anschliessend werden alle geladenen Strings zur Textarea hinzugefügt.

public static void save(String text){ }

Woche 3: Drittes Treffen: Refactor Funktion und der XML-Katalog wurden zum Hauptprogramm hinzugefügt. Umwandlung des normalen Java Projektes in ein Gradle Projekt. Aufteilung der nächsten Themen: Babysteps und Tracking.