**CS-172 Final Requirement Specification**

**Single Student Class Scheduler**

**Needed Classes and Relationship:**

* StoredInfo *(contains all collected info converted into C++)*
* UserConfig *(contains user’s name and other info defined at beginning)*
* SingleCourse *(contains single course info and a list of* ***SingleSection*** *classes)*
* SingleSection *(contains single section info – time, professor, days of week)*
* ChosenCourse::SingleCourse *(inherits* ***SingleCourse*** *class + contains user set data for course)*
* Calculation *(pulls data from* ***StoredInfo*** *class and performs algorithm to reach final schedule)*
* InitialInputForm *(performs the collection of background info all going to a* ***UserConfig*** *object*)
* CourseSelectionForm *(displays a list of* ***ChosenCourse*** *classes and allows to add/remove them)*
* AddCoursePopup *(includes filters and a list of available* ***SingleCourse*** *classes)*
* ResultForm *(shows a printable timetable of the optimized schedule)*

**UML Diagrams:**

Finding Template…

**Initial Approach:**

For anticipated program workflow please review the *Forms Workflow* file. It contains the layout of different program windows and their intended components/functions.

In terms of code, we will begin by reading data into C# classes using the ExcelReader Library. The information would then need to be converted/stored in the C++ class *StoredInfo.* A Calculation Class would be initiated by the user which would go through the data and analyze it for the optimal arrangement of courses based on the available section times. The results would then need to be formatted to be visually appealing and preferably printable.

**Challenges:**

* Reading data from unique/ un-standardized excel spreadsheets
* Alternating between C++ and C# code
* Figuring out the inner workings of an algorithm that could handle class arrangement
* Displaying results in an organized manner