The control flow of this desktop application mostly follows the button clicks of a user. They firstly specify their major, which prompts the program to access relevant data from majors.cpp. This data contains all required courses within a major, total credit hours, total hub units, current/completed credit hours, current/completed hub credits, and a text block displaying non-major/elective requirements. Due to the variation in HTML tags in non-major/elective requirements and the text itself from major-major, we were unable to implement web-scraping with Python to attain all courses at BU. All courses and their information is stored in the file all\_courses.cpp in the form of a linked list. Each object of the list contains the data College, Course #, Credit hours, and Hub credits, all separated by commas. This allows users to search for elements containing specific hub credits, course #, college, or credit hours, which creates a skipped linked list storing data from all matching elements. Performing a search by iteration through a linked list is not time efficient, however, it makes the later deletion and insertion of elements very efficient. After a search, the user can view all skipped list elements in the form of buttons. These buttons have display strings that are separate from how the data is actually stored, as the commas are removed. If they see a course they would like to add, they select a button, and it is added to the currently opened semester. This addition will invoke the bool Update() function in majors.cpp, which checks the relevant required courses list if this performed through the required course search bar. If it exists in required\_courses[], it is removed from the list and remaining credit hours, current credit hours, remaining hub credits, and current hub credits are updated. If they search within the hub/elective course search bar, required\_courses[] is not parsed, but the other variables are updated. If they would like to remove a course, the opposite is performed with the rUpdate() function in majors.cpp. These changes will show their remaining credit hours, current credit hours, remaining hub credits, and current hub credits, but not the remaining required\_courses[]. Aside from the course planning side of the application, users are able to calculate their cumulative GPA. Additionally, they are able to specify goal GPA, which will calculate the average GPA for ungraded courses to attain a GPA.