Name: Ojas Bardiya

UID: 505145284

1. One challenging aspect about developing the code for this project was making sure that if there was an error or multiple errors in the input, the error messages were displayed accurately. I had to carry out multiple tests for ascertaining that in case a user entered a or several wrong input value, the error code would be the same as that explained in the project specifications.

Another aspect I found difficult at first was making sure that the empty string was displayed as an error, but after reading the notes on String provided on CCLE, I understood how using getline instead of cin would solve that issue.

1. Test cases for my program (according to the sequence of inputs)
2. Error in BruinCard /Ticket input – (error) – can be used to determine whether the error message “The kind value must either be Ticket or BruinCard” works correctly.
3. Error in any other input value – (Ticket, error); (BruinCard, Lunch, Student, error); (BruinCard, error, Staff); (BruinCard, Breakfast, error) – to determine the corresponding error messages are displayed correctly.
4. Multiple errors in an input – (BruinCard, error, error) – In this case the error message for only the first error must be displayed and the second must be ignored.
5. Using Ticket – (Ticket, Dinner/Breakfast/lunch) – make sure price displayed is correct for a particular meal with ticket.
6. Using BruinCard and Staff – (BruinCard, Dinner/Breakfast/Lunch, Staff) - make sure price displayed in each corresponding case is correct.
7. Using BruinCard and Student – (BruinCard, Dinner/Breakfast/Lunch, Student, Yes/No) – make sure price displayed in each corresponding case is correct.
8. Making sure empty space is an error – (Ticket, ““); (BruinCard, “”, Staff); (BruinCard, Dinner, Student, “”) – and the error message is displayed accordingly.