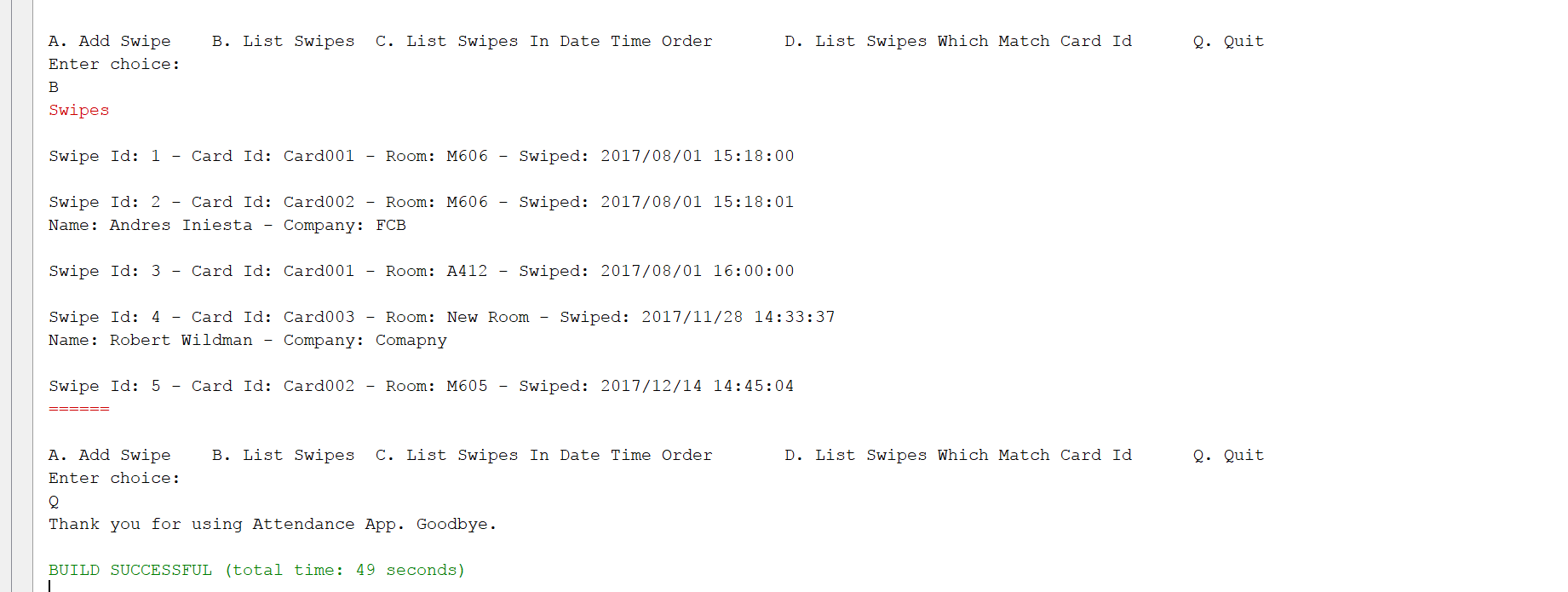
**Summary of Programming Course Work – Robert Wildman S1509388**

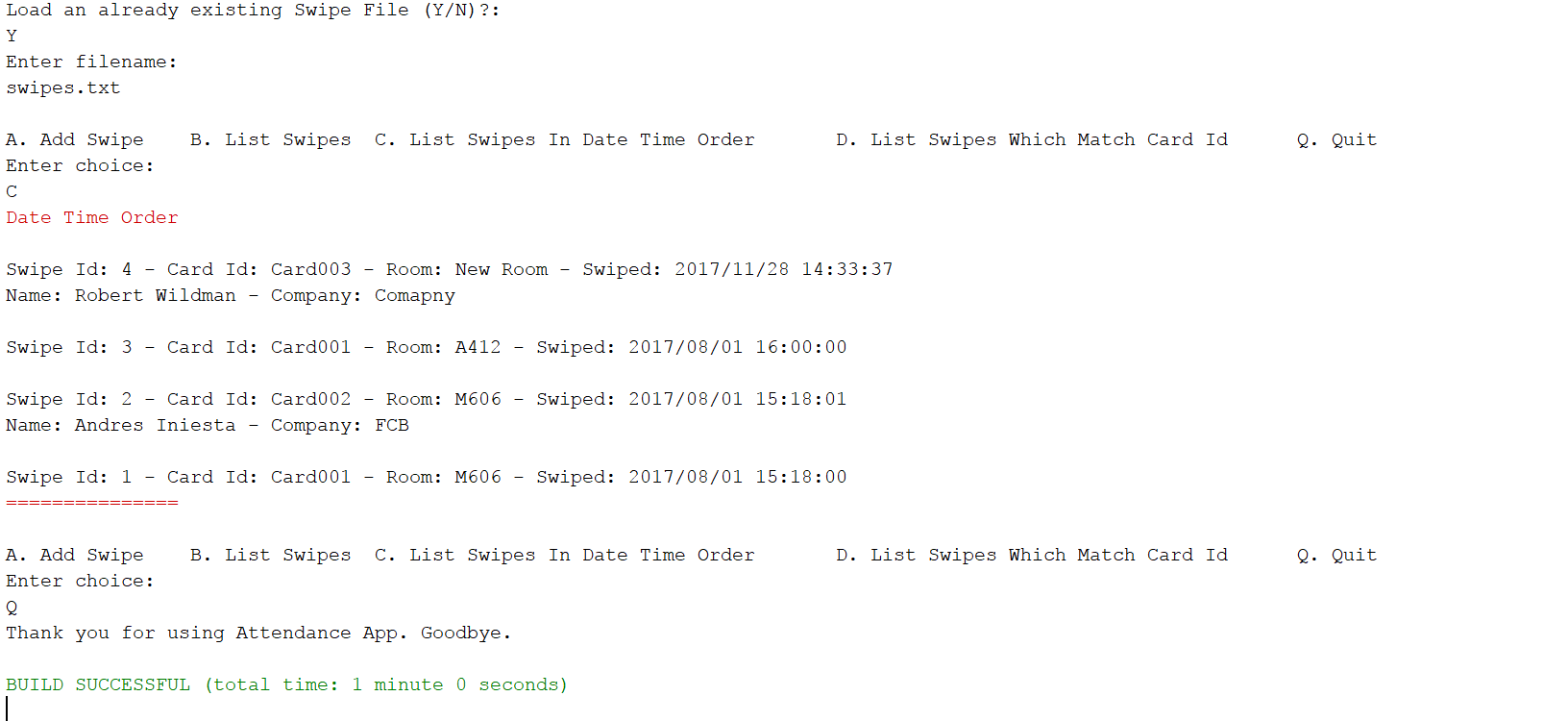
Increment 1



In the first Increment I completed the Swipe and Visitor Swipe Class to the Class Diagram in Appendix 1, Which included some of the code needed in other class but through it was best to fill out these classes first then work towards the increments. I decided to use ArrayList as my collection in the Repository class as I have used them a lot before and they worked well with my need for the system and what I need the Repository to be able to do. Was able to add the methods need for the Listing of the swipes and was able to print both Normal Swipes and Visitors Swipes to the system. Created the DAOImpl Class and had it read the text file line by line for swipes. To work out the different between a normal swipe and a visitor swipe I checked the size of the array when we spilt line the up using the delimiter. If the Array was greater then 4 it would be a visitor swipe and we add it in as one.

Increment2  

For increment 2 I had to be able to add a new swipe into the system. When adding a new swipe into the system I set it as the current date as specified in the Specification. The user gets asked if it’s a Visitor Swipe or a Normal Swipe. Then depending on the inputted value, we ask question to make the swipe possible. There is error checking just in case a user enters a wrong value. This information is then used to create a Swipe object and then saved to the Repository Object. Listing the swipes will now show the new swipe and will show the date as the time we submitted the add request.

Increment 3

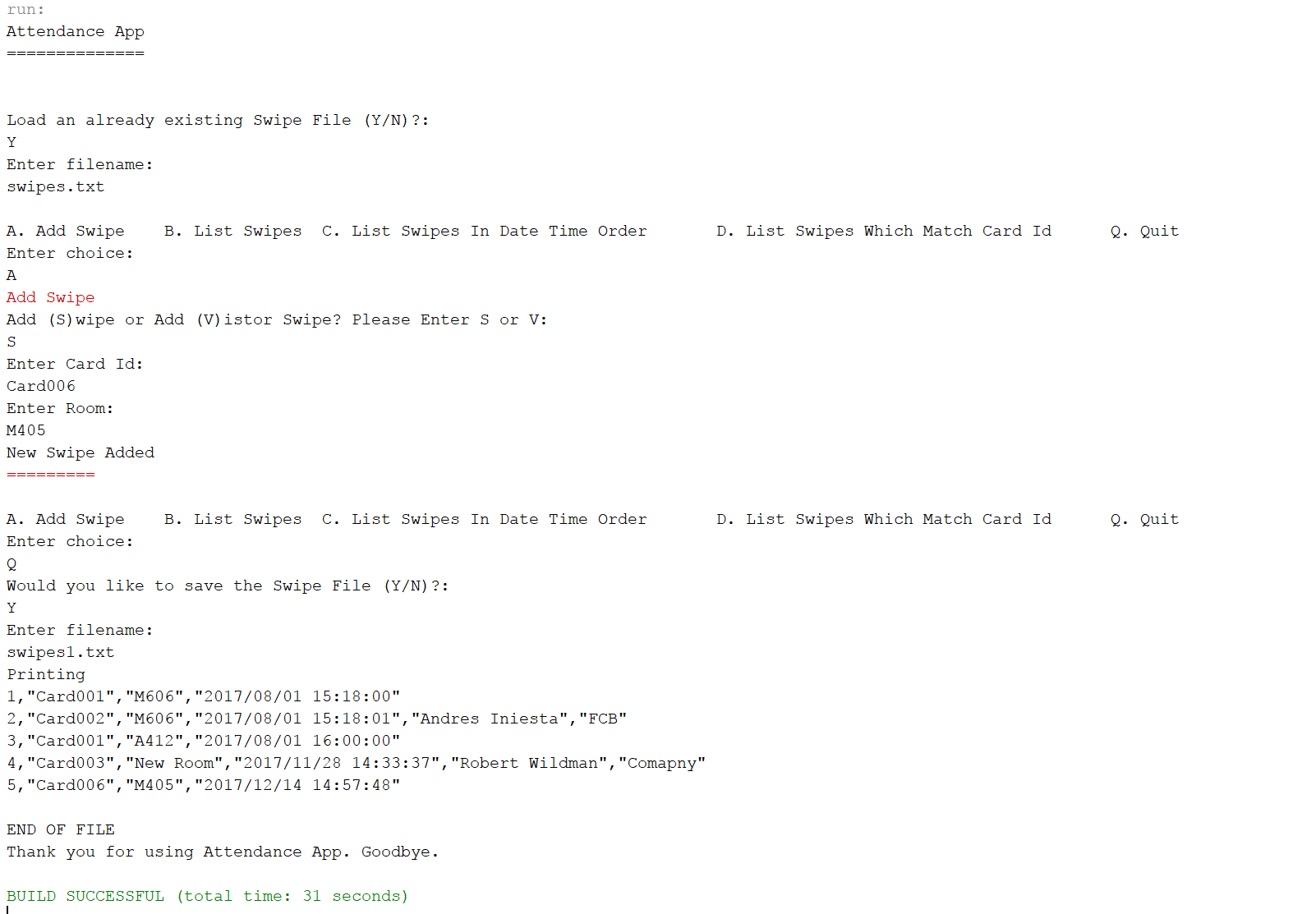
This increment was the hardest of the increments to implement as there was many ways of solving the problem. The way I decided to tackle this problem was to have the ArrayList change into a TreeSet which when you change between the List and Set it will Organise the list based on your Comparator in the Swipe Class. I had it set up so the compareTo method would compare the date time using the Calendar CompareTo method and will list the dates from newest to oldest. Then looped for the new TreeSet and displayed the swipe as normal. The other ways of doing this would be through stream which is the way I do it in Increment 4 or by implementing your own sorting system so you could sort the ArrayList on the fly.

Increment 4



For this increment the program will ask the user what card ID they would like to get the information about, Then call a stream of the ArrayList created in repository which holds all the swipes. With this stream I first filter by swipes matching the card id provide and with the reminding stream I do a count on so I know how many swipes there has been on that cardid. Next I want to get the Last Swipe inputted into the system so I do the same process as last time however this time I use the Sorted method to sort the stream using the Comparator method in the Swipe Class which I talked about in the last increment. Then with the sorted list I use the findFirst method to get the first Swipe on the sorted list and have a orElse catch at the end which will set that object to null. Then I print out the information to the user.

Increment 5





With this increment I had to be able to store the file at the end of the application. In the testing it shows the user adding a new file and then making sure that its getting listed. User then quits the application and gets asked the question if they would like to save the file. If the user says yes, then the program ask for a filename and then sends that filename to the Repository Class which calls all the swipes toString methods to print to the file with delimiters. To make sure the file is saved correctly to the file, the user tries to load in the file and list the swipes which is found to be correct and the file saved and loaded correctly.