# Phase 3 – Failure Table:

**Note:** Most, if not all of the tests were modified to suit the actual outputs of the program. The reasoning for this was because the diff command in Linux would interpret text files that had only a newline for being a difference to be different. Thus, the expected output files were modified to be exactly like what would be outputted, but this was as much as the modification extended. For the sake of simplicity, all outputs that were different and wrong solely from whitespace and newline differences are omitted from this table.

| Name: | Observed Test Failure | What it was testing: | How its output was wrong: | Error in Code: | Changes to fix it: |
|---|---|---|---|---|---|
| Create2 | Created a username with non-alphanumeric characters | Catching a non-alphanumeric string and outputting an error. | No error was produced and program was allowed to proceed | No statement to output the error | An if statement catching this and sending the user back |
| addCredit2 | Error statements prompting to enter a valid command | Buy Standard account adding credits to their account | Infinite loop, program could not terminate | No checks for the account not being an admin, assumed it was an admin and prompted a username input | A boolean value to check if the current user is an admin and goes straight to asking for amount of credits |
| addCredit3 | Error statements prompting to enter a valid command | Sell Standard account adding credits to their account | Infinite loop, program could not terminate | No checks for the account not being an admin, assumed it was an admin and prompted a username input | A boolean value to check if the current user is an admin and goes straight to asking for amount of credits |
| addCredit4 | Error statements prompting to enter a valid command | Full Standard account adding credits to their account | Infinite loop, program could not terminate | No checks for the account not being an admin, assumed it was an admin and prompted a | A boolean value to check if the current user is an admin and goes straight to asking for amount of credits |

**Phase 3 – Failure Table:**

| | | | | | |
|---|---|---|---|---|---|
| | | | | username input | |
| Refund5 | No errors from entering a negative number | Entering a negative number to test if an error would occur. | Program worked normally as if no improper input had occurred | No checks for a non-positive number. | An if statement to check for non-positive numbers and to output an error. |
| Buy6 | No errors from attempting to buy from an event with less tickets than asked for. | Entering a value for buying tickets that the event doesn't have enough for. | Program worked with no error outputs and acted as if it was a proper input. | No checks if the event has less tickets than asked for. | Proper syntax for checking the amount of remaining tickets left and an if statement to check this and output an error. |
| Buy9 | No error if the entered event name exceeded 25 characters | If an error would occur when the event name specified is longer than 25 characters. | Program outputted an error (Event didn't exist), but did not output an error for exceeded character count. | No checks for exceeded characters. | An if statement to check for the input being more than 25 characters. |
| Sell2,3,4,7,10,11 | Daily Transaction File was missing the Seller's username on each line | A proper input for the sell transaction, each should output a successful transaction | Daily Transaction File did not have the proper format for the sell transactions | Syntax error where the function that writes to the DTF was missing the seller's username input. | Corrected the syntax to actually have the seller's username in the DTF. |