The in-class report should provide: i) a summary list of your test cases in Part A and test cases in Part B, ii) execution test results (numbers passed and numbers failed – with your defined severity level), iii) any fixing or re-test performed and iv) a discussion of your experiences (e.g. what was difficult to design; how much effort in person hours did it take to design test cases, execute the test cases and record the material; what was effective and what was not, etc.). (*Look at Lecture Notes on Test Planning for hint of Test Result Report.)*



The full list of our test cases can be found in appendix A and B. A quick breakdown of the black box testing can be seen above in our decision table. This method allowed us to quickly determine which tests would be most likely to produce errors and bust through robust boundary value testing theoretically. Our path (white box) testing was performed later and can be seen in appendix B. It covered all main paths, but not all possible paths due to time constraints. Input validation tests were also performed.

Our levels of severity were as follows: 1. The program fails to complete/run. 2. The program produces improper output. 3. Any formatting problems or any noted lack of optimization.

Our first set of errors was encountered during input validation testing. If invalid data was entered the program would loop infinitely asking for user input. This was due to a failure to clear the input stream and was quickly resolved. This was the only level 1 error found. Three level 2 errors were found, and they were found during black box testing. First, there were some round off errors in the tax percentage output e.g. .25 became .25000000001. This was caused by using doubles in Java and was quickly resolved by using decimals instead. Secondly the program didn’t output the amount of money owed, it output only the tax percentage. This was caused by the programmer misreading the requirements document and was resolved after a brief discussion. The final error found in this program was the tax percentage was not properly calculated if the number of dependents equaled zero. The exasperated programmer quickly added an equals to his switch statement, and resolved this problem as well. Only two level 3 errors were found. The first printed statement said “Please enter you’re annual income”. This used the wrong “your”, and was replaced with your in the final version. The other error was a memory leak in the output streams, which were not closed initially, and was also fixed in the final release. All of these errors were found during black box testing. A few days later the white box tests were completed and all main paths operated perfectly. No defects were found. The black box tests were run again after all errors were resolved and no further issues were found.