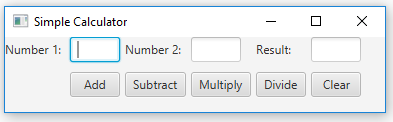
**System Level Testing**

In order to test the capabilities and efficiency of my application, we will be performing some tests.

I will be testing an application that I wrote that is a simple calculator. I will be listing the tests that I performed, the results and conclusion.



1. **Ease of Access** – This test will determine how easy the app is to use from the user perspective. A user does not need nor want to know the inner workings of an application, as long as it does what they want it to do fast and effectively. The application is a simple calculator that does simple math, so I had my wife test it out. Everything seemed to work, one of the problems that I noticed though was that the calculator does not handle fractions well when represented as (1/2) instead of 1.5. Another problem is that the calculator will accept input from the keyboard and will accept characters but will crash once it tries to calculate them.

**Conclusion:** I need to add number buttons to the application to limit the choices the user can perform. I would also need to find a way to only accept input from the provided buttons and not from the keyboard. That way the user will be forced wo only use valid characters and numbers. If somehow the user is able to input an invalid character, I need to implement some error handling to let the user know of the mistake but not crash the whole application.

1. **Stress testing:** We tested the amount of calculations that the calculator would handle. As the calculator uses the resources from the computer we found that it can handle some very big numbers. Although it can handle huge numbers, we did find that when using exponentials the calculator somehow got confused and just crashed. This might be due to the sheer number of calculations that are needed to come up with an answer or the calculator misinterpreted our input.

**Conclusion:** I need to further investigate this problem, further tests need to be performed to determine what the real cause for the crash is. I need to find the real limit of the calculator and determine if it can really handle exponentials or if we just messed up on the inputs. So far the calculator seems to be doing good

1. **Button Positioning and logic:** We compared my calculator with common physical and logical calculators. As mentioned before, the calculator is lacking number buttons, which allow for some serious human error. The calculation buttons are described by words and not symbols, which I personally like but some users might be confused by it. The calculator also only allows for two numbers to be inserted which limits the amount of calculations that are possible. We found that even though the buttons are clear and big enough as to not to confuse them one with another, or press them by accident, it would probably be best to change them from words to symbols

**Conclusion:** The calculator is very simple and primitive compared to others. There is much work to be done. I would love to make it more of a scientific calculator. In order to do so the whole UI has to change, and some more code needs to be added to handle the calculations. As far as the calculator goes in its current state, it is useful and would work for small calculations.

|  |  |  |  |
| --- | --- | --- | --- |
| **Function tested** | **What’s expected** | **Test with right/wrong input** | **Result pass/fail** |
| Application controller | Options to select from a new or old list. It is a string input. Numbers or invalid inputs should produce an error. | Right  Wrong | Pass  Partially fail; it throws error message, but ends application |
| Input Mechanism | It should only take numbers as input. | Right  wrong | Pass  Fail |
| Adding quantity of items | It should only take integers as input. | Right  Wrong | Pass  Pass |
| Application compiles | The application should compile without errors. |  | Pass |