./

Learning Report – Applied System Development Life Cycle and Software Testing



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **To be approved By** | **Remarks/Revision Details** |
| 1 |  | Name/PS No | Name/PS No | Module Owner Name | Comments |
| 2 | 15/02/21 | 99003779 |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Document History**

Table of Contents

ACTIVITY 1……………………………………………………………………………………………………………………………………………………………………4

STATE OF ART…………………………………………………………………………………………………………………………………………………………….4

COST AND FEATURES…………………………………………………………………………………………………………………………………………………4

DEFINING OUR SYSTEM……………………………………………………………………………………………………………………………………………..5

SWOT ANALYSIS…………………………………………………………………………………………………………………………………………………………5

4W1H………………………………………………………………………………………………………………………………………………………………………..6

UML DIAGRAMS………………………………………………………………………………………………………………………………………………………..6

TEST PLANS……………………………………………………………………………………………………………………………………………………………….9

GITHUB……………………………………………………………………………………………………………………………………………………………………11

GIT ISSUES……………………………………………………………………………………………………………………………………………………………….11

MAIN PROJECT ……………………………………………………………………………………………………………………………………………………….12

GIT COMMITS………………………………………………………………………………………………………………………………………………………….15

[Table of Figures 3](#_Toc64289204)

[Table of Tables](#_Toc64289205) 3

## Table of Figures

[Figure 1 COST VS FEATURES](#_Toc52177314) 5

[Figure 2 CLASS DIAGRAM (HIGH LEVEL)](#_Toc52177315) 6

[Figure 3 ACTIVITY DIAGRAM (HIGH LEVEL)](#_Toc52177316) 6

Figure 4 INTERACTION OVERVIEW (HIGH LEVEL)……………………………………………………………………………………………..7

[Figure 5 OBJECT DIAGRAM (LOW LEVEL)](#_Toc52177317) 7

[Figure 6 SEQUENCE DIAGRAM (LOW LEVEL)](#_Toc52177318) 8

[Figure 7 STATE DIAGRAM (LOW LEVEL)](#_Toc52177319) 8

[Figure 8 GITHUB LINK](#_Toc52177324) 12

[Figure 9 GIT ISSUES](#_Toc52177326) 13

[Figure 10 GIT COMMITS](#_Toc52177327) 22

## Table of Tables

[Table 1 SWOT ANALYSIS](#_Toc52177304) 5

[Table 2 TEST PLAN OF HLR](#_Toc52177305) 9

[Table 3 ADDITION TEST PLAN (LOW LEVEL)](#_Toc52177306) 9

[Table 4 SUBTRACTION TEST PLAN (LOW LEVEL)](#_Toc52177307) 10

[Table 5 MULTIPLICATION TEST PLAN (LOW LEVEL) 1](#_Toc52177308)0

[Table 6 DIVISION TEST PLAN (LOW LEVEL) 1](#_Toc52177309)0

[Table 7 REMAINDER TEST PLAN (LOW LEVEL) 1](#_Toc52177309)1

[Table 8 POWER TEST PLAN (LOW LEVEL) 1](#_Toc52177309)1

[Table 9 PERCENTAGE TEST PLAN (LOW LEVEL) 1](#_Toc52177309)1

[Table 10 GCD TEST PLAN (LOW LEVEL) 1](#_Toc52177309)2

**ACTIVITY 1**

**TO DESIGN A CALCULATOR USING C PROGRAMMING**

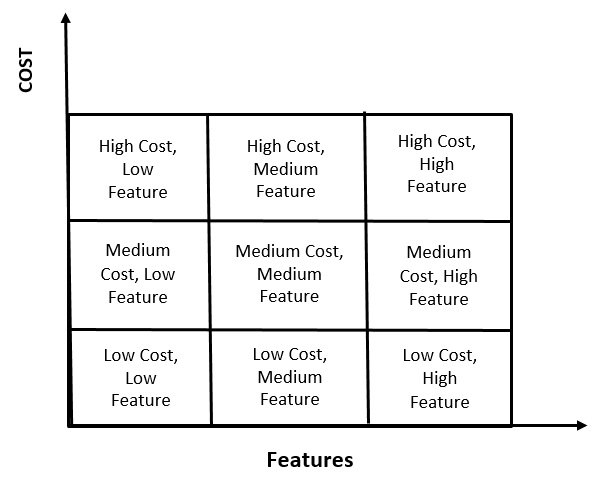
**State of Art :**

In order to meet the requirement of today’s world, man has to be very fast. To do so, it is very genuine to face difficulties to meet some basic essentials. So we did some brain storming to search for them. The Research has been divided on the basis of cost and features of different calculators. The price ranges from Rs 100-300 which can be used by students in school & in universities, scientist and scholars. This device includes a large range of features at lower cost. The features include basic arithmetic operations, BMI calculation, and conversion of Numeric into binary, octal, hexadecimal and vice versa. It can also perform basic trigonometry calculations. It includes nth root and power of n calculating features. Another category includes Body mass index (BMI) feature, calculation of trigonometric functions including exponential, logarithms and number conversion. The features can be enhanced further but at the cost will increase accordingly. At a range of Rs 600-650, complex calculations and imaginary numbers can be included. Calculators having medium costs are having medium set of features which includes matrix and calculus. Higher features at medium cost of price includes binary conversions and are foldable. Calculators of higher prices includes the functions of database management, higher accuracy, wider display for plots and graphs. It also includes smart touch, solar cell operations, battery charging and a waterproof.

**Cost and Features:**

The whole document has been divided on the basis of cost and features of different calculators. Following is a list of features based on different cost and prices of calculators.

1. Low cost and low featured Calculator: These types of calculator ranges from Rs. 50-200. It will be able to calculate basic arithmetic operations and are mobile.
2. Low cost and medium featured: Such calculator ranges from Rs. 100-300 which will be able to perform arithmetic calculations including fractions, nth root and power of n.
3. Low cost and high featured: It includes operation of trigonometry, logarithm, exponential, inversion and degrees. These calculators ranges from Rs. 250-500.
4. Medium Cost and Low featured: It ranges from Rs. 500-650 with features of solving complex calculations, imaginary number and is having a larger display.
5. Medium cost and medium featured: It ranges from Rs. 750-1500 including matrix operation, calculus and statistics.
6. Medium cost and high featured: These types of calculator ranges from Rs 1200-2500. These are able to perform number calculations and are foldable.
7. High cost and low featured: Such calculator ranges from Rs. 5000-10000. It includes features for respective business fields with high accuracy.
8. High cost and medium featured : It includes printing features with touch, solar charging, data security and internet at a price range of Rs 12000-25000
9. High cost and high featured: Its price ranges from 30000 to 1 lakhs and are named as programmable calculators.

****

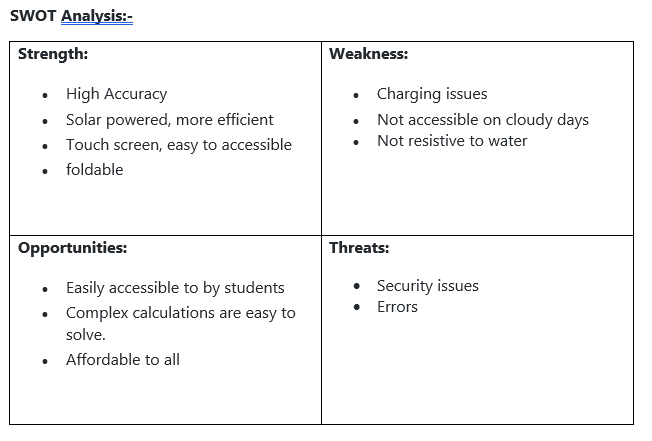
**Figure 1: Cost vs features**

**Defining our system:**

The designed product has all the necessary features required by the undergraduates and postgraduates students including scholars. The High level requirements include arithmetic operations, decimals, trigonometric functions, nth root, power of n, fractions, percentage, logarithms, exponentials, binary conversions etc. The low level requirements of the product includes addition, subtraction, multiplication, division under arithmetic operation, decimal upto 8 digits, trigonometric functions with square root and radians.

**SWOT Analysis:**

**Table 1: SWOT Analysis**

[](https://user-images.githubusercontent.com/78867425/107847807-b9c48f00-6e14-11eb-9598-0ddc49f4c83b.PNG)

# 4W's and 1'H :

# Who: Basically used by students of UG and PG.

## **What:** It is a highend, affordable calculator.

## **When:** Easily accessible and performs complex calculations.

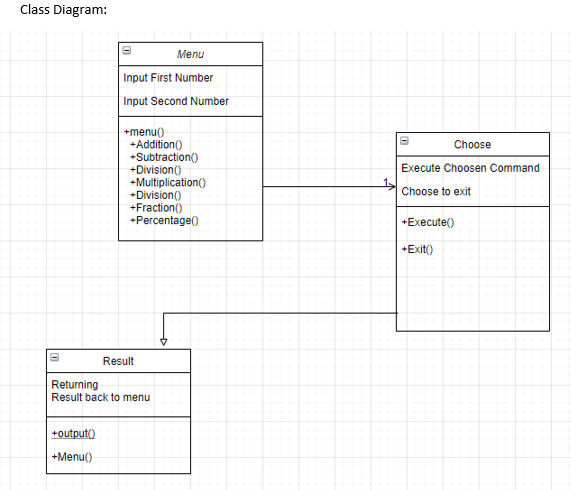
## **Where:** To solve simple as well as complex calculations.

## **How:** End user friendly and easily accessible.

**UML Diagram:**

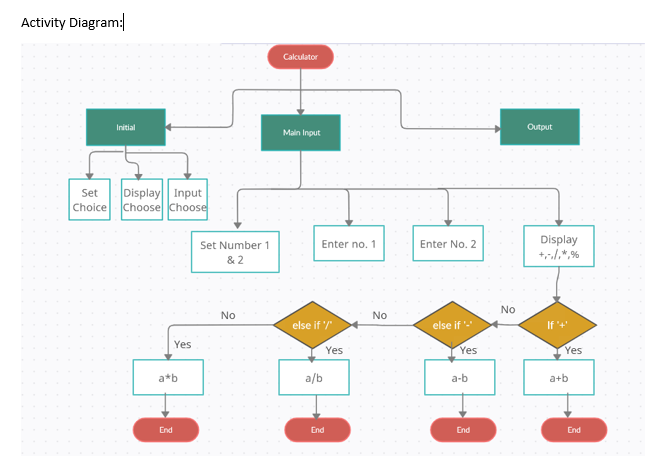
High Level Design:

Structural Diagram

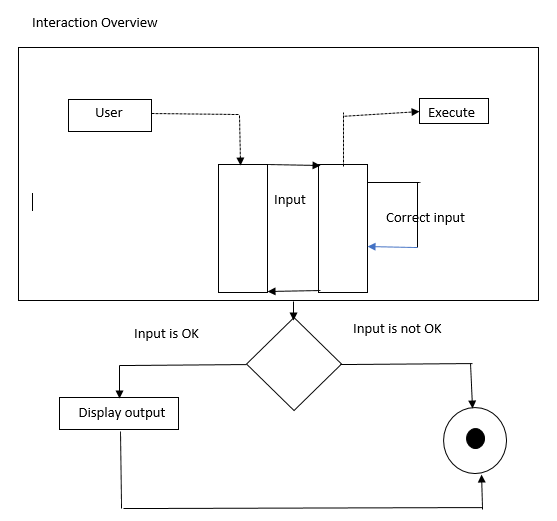


**Figure 2: Class Diagram of arithmetic calculator**

Behavioural Diagram:



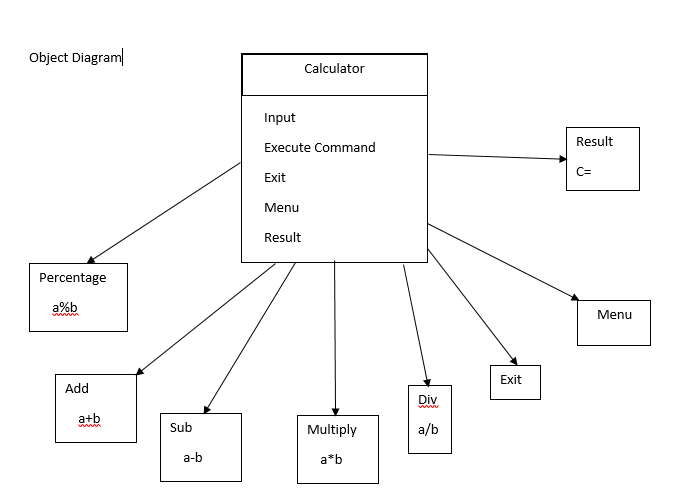
**Figure 3: Activity Diagram of arithmetic calculator**



**Figure 4: Interaction overview of Arithmetic calculator**

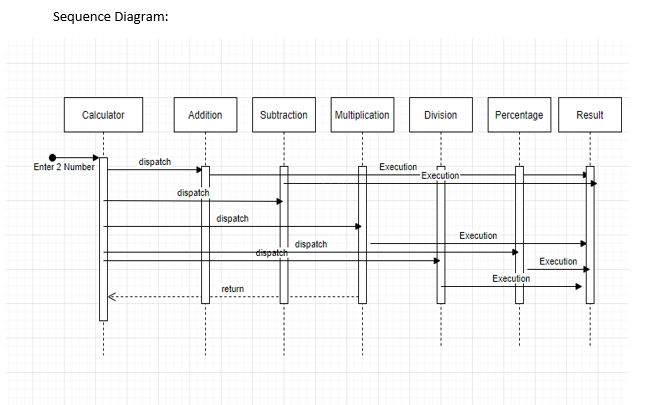
Low Level Design:

Structural Diagram:

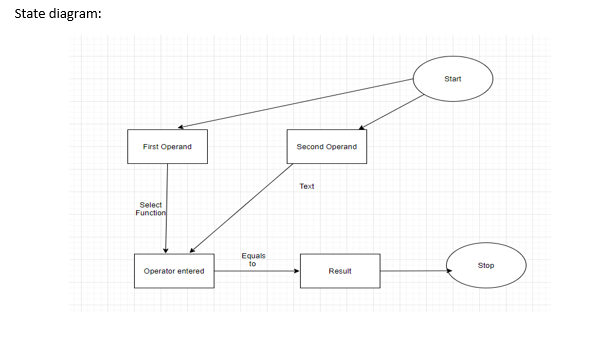


**Figure 5: Object diagram of arithmetic table**

Behavioural Diagram:



**Figure 6: Sequence Diagram of Arithmetic Calculator**

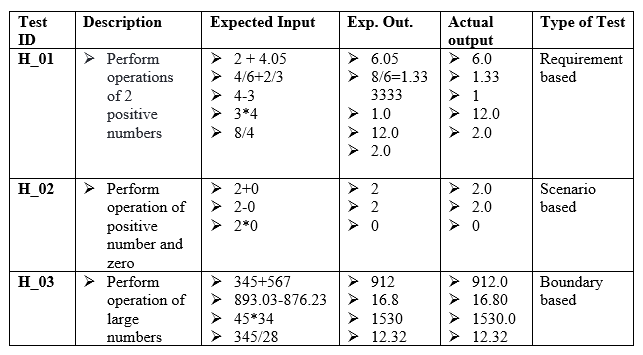


**Figure 7: State Diagram of Arithmetic Calculator**

**Test Plan:-**

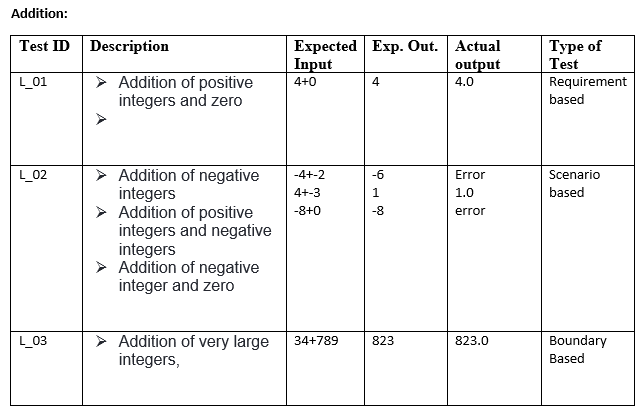
**HLR:**

**Table 2: Test Plan of HLR**

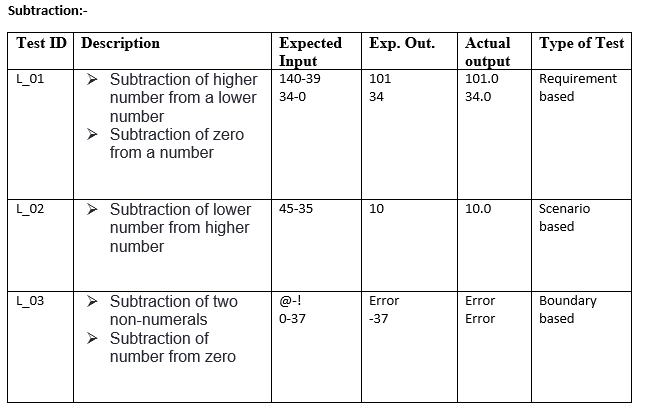


**LLR:**

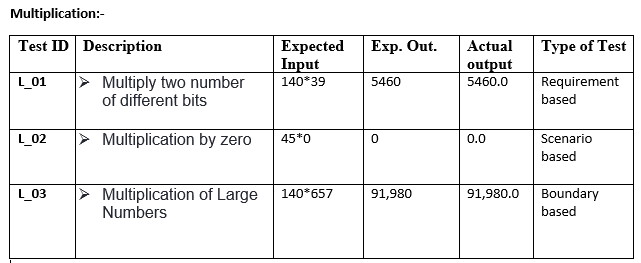
**Table 3: Addition test plan**



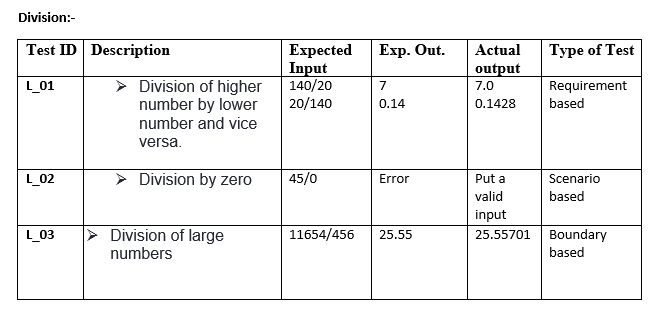
**Table 4: Subtraction Test Plan**

****

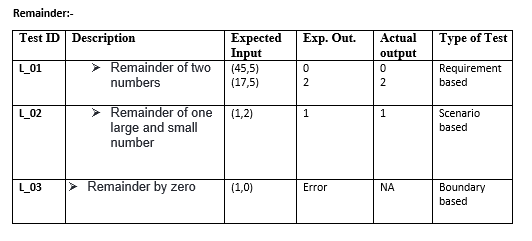
**Table 5: Multiplication Test Plan**



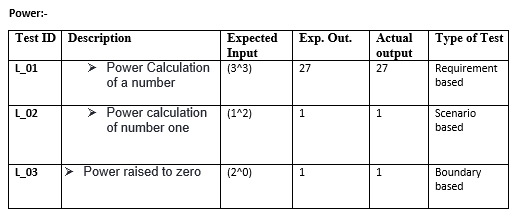
**Table 6: Division Test Plan**



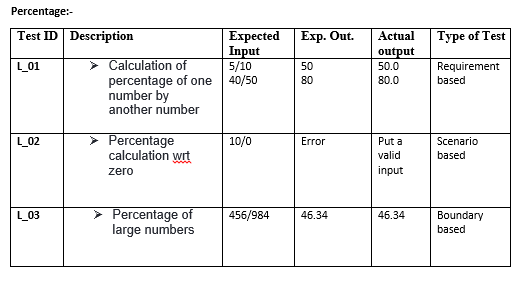
**Table 7: Remainder Test Plan**

****

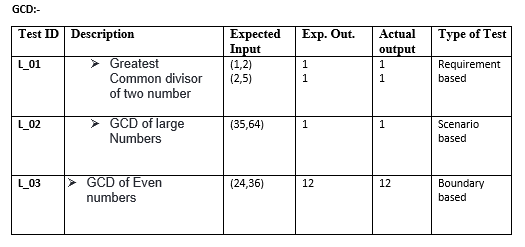
**Table 8: Power Test Plan**

****

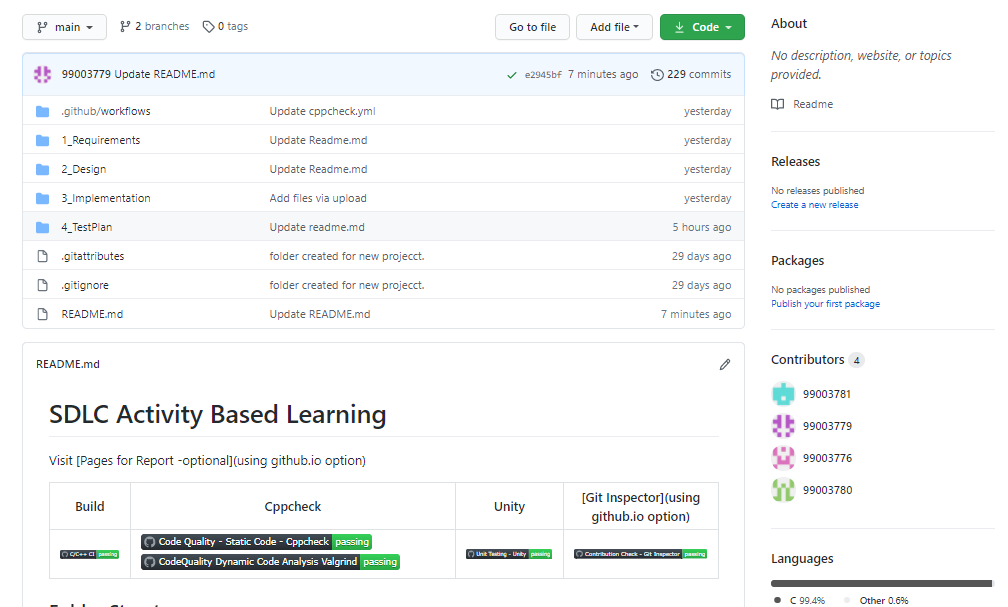
**Table 9: Percentage Test Plan**

****

**Table 10: GCD Test Plan**

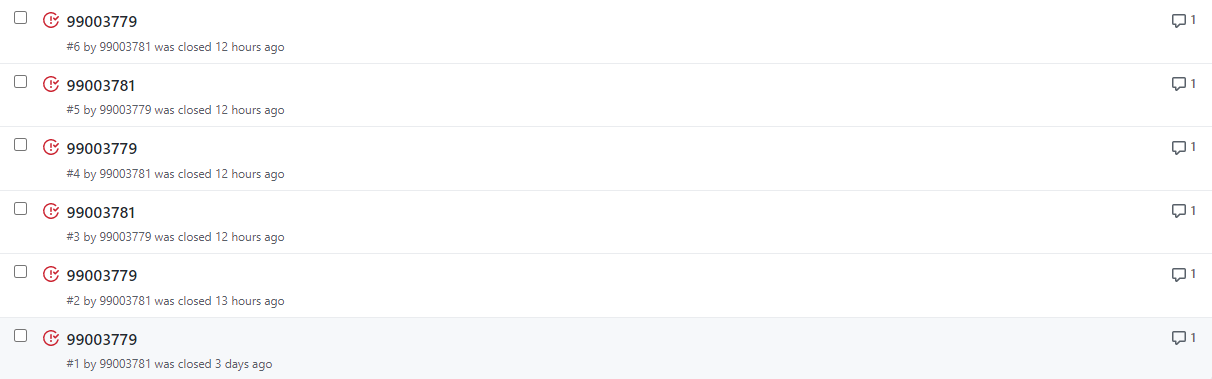
****

**Git :**



**Figure 8:** [**https://github.com/99003781/N8-Calculator.git**](https://github.com/99003781/N8-Calculator.git)

**Git Issues:**



**Figure 9: Screenshots of Git issues raised and closed**

**Main Project:**

#include "unity.h"

#include "main.h"

#include "calculator\_operations.h"

/\* Prototypes for all the test functions \*/

void test\_add(void);

void test\_add\_testcase2(void);

void test\_add\_testcase3(void);

void test\_add\_testcase4(void);

void test\_add\_testcase5(void);

void test\_subtract(void);

void test\_subtract\_testcase2(void);

void test\_subtract\_testcase3(void);

void test\_subtract\_testcase4(void);

void test\_subtract\_testcase5(void);

void test\_multiply(void);

void test\_multiply\_testcase2(void);

void test\_multiply\_testcase3(void);

void test\_multiply\_testcase4(void);

void test\_multiply\_testcase5(void);

void test\_divide(void);

void test\_divide\_testcase2(void);

void test\_divide\_testcase3(void);

void test\_divide\_testcase4(void);

void test\_divide\_testcase5(void);

void test\_percentage(void);

void test\_percentage\_testcase2(void);

void test\_percentage\_testcase3(void);

void test\_percentage\_testcase4(void);

void test\_percentage\_testcase5(void);

void test\_gcd(void);

void test\_gcd\_testcase2(void);

void test\_gcd\_testcase3(void);

void test\_gcd\_testcase4(void);

void test\_gcd\_testcase5(void);

void test\_rem(void);

void test\_rem\_testcase2(void);

void test\_rem\_testcase3(void);

void test\_rem\_testcase4(void);

void test\_rem\_testcase5(void);

void test\_power(void);

void test\_power\_testcase2(void);

void test\_power\_testcase3(void);

void test\_power\_testcase4(void);

void test\_power\_testcase5(void);

void setUp(){}

void tearDown(){}

int main(void)

{

UNITY\_BEGIN();

RUN\_TEST(test\_add);

RUN\_TEST(test\_add\_testcase2);

RUN\_TEST(test\_add\_testcase3);

RUN\_TEST(test\_add\_testcase4);

RUN\_TEST(test\_add\_testcase5);

/\*SUBTRACTION\*/

RUN\_TEST(test\_subtract);

RUN\_TEST(test\_subtract\_testcase2);

RUN\_TEST(test\_subtract\_testcase3);

RUN\_TEST(test\_subtract\_testcase4);

RUN\_TEST(test\_subtract\_testcase5);

/\*MULTIPLICATION\*/

RUN\_TEST(test\_multiply);

RUN\_TEST(test\_multiply\_testcase2);

RUN\_TEST(test\_multiply\_testcase3);

RUN\_TEST(test\_multiply\_testcase4);

RUN\_TEST(test\_multiply\_testcase5);

/\*DIVISION\*/

RUN\_TEST(test\_divide);

RUN\_TEST(test\_divide\_testcase2);

RUN\_TEST(test\_divide\_testcase3);

RUN\_TEST(test\_divide\_testcase4);

RUN\_TEST(test\_divide\_testcase5);

/\*PERCENTAGE\*/

RUN\_TEST(test\_percentage);

RUN\_TEST(test\_percentage\_testcase2);

RUN\_TEST(test\_percentage\_testcase3);

RUN\_TEST(test\_percentage\_testcase4);

RUN\_TEST(test\_percentage\_testcase5);

/\*GCD\*/

RUN\_TEST(test\_gcd);

RUN\_TEST(test\_gcd\_testcase2);

RUN\_TEST(test\_gcd\_testcase3);

RUN\_TEST(test\_gcd\_testcase4);

RUN\_TEST(test\_gcd\_testcase5);

/\*REMAINDER\*/

RUN\_TEST(test\_rem);

RUN\_TEST(test\_rem\_testcase2);

RUN\_TEST(test\_rem\_testcase3);

RUN\_TEST(test\_rem\_testcase4);

RUN\_TEST(test\_rem\_testcase5);

/\*POWER\*/

RUN\_TEST(test\_power);

RUN\_TEST(test\_power\_testcase2);

RUN\_TEST(test\_power\_testcase3);

RUN\_TEST(test\_power\_testcase4);

RUN\_TEST(test\_power\_testcase5);

return UNITY\_END();

}

/\*ADDITION\*/

void test\_add(void)

{

TEST\_ASSERT\_EQUAL(30, add(10, 20));

}

void test\_add\_testcase2(void)

{

TEST\_ASSERT\_EQUAL(-10, add(10, -20));

}

void test\_add\_testcase3(void)

{

TEST\_ASSERT\_EQUAL(-20.5, add(-40.5, 20.0));

}

void test\_add\_testcase4(void)

{

TEST\_ASSERT\_EQUAL(41.0, add(15.5, 25.5));

}

void test\_add\_testcase5(void)

{

TEST\_ASSERT\_EQUAL(15000, add(7500, 7500));

}

/\*SUBTRACTION\*/

void test\_subtract(void)

{

TEST\_ASSERT\_EQUAL(-3, subtract(0, 3));

}

void test\_subtract\_testcase2(void)

{

TEST\_ASSERT\_EQUAL(2, subtract(5, 3));

}

void test\_subtract\_testcase3(void)

{

TEST\_ASSERT\_EQUAL(7, subtract(10, 3));

}

void test\_subtract\_testcase4(void)

{

TEST\_ASSERT\_EQUAL(-4, subtract(3, 7));

}

void test\_subtract\_testcase5(void)

{

TEST\_ASSERT\_EQUAL(100, subtract(1000, 900));

}

/\*MULTIPLICATION\*/

void test\_multiply(void)

{

TEST\_ASSERT\_EQUAL(0, multiply(1, 0));

}

void test\_multiply\_testcase2(void)

{

TEST\_ASSERT\_EQUAL(15, multiply(5, 3));

}

void test\_multiply\_testcase3(void)

{

TEST\_ASSERT\_EQUAL(0, multiply(10, 0));

}

void test\_multiply\_testcase4(void)

{

TEST\_ASSERT\_EQUAL(-30, multiply(6, -5));

}

void test\_multiply\_testcase5(void)

{

TEST\_ASSERT\_EQUAL(10, multiply(2, 5));

}

/\*DIVISION\*/

void test\_divide(void)

{

TEST\_ASSERT\_EQUAL(0, divide(1, 0));

}

void test\_divide\_testcase2(void)

{

TEST\_ASSERT\_EQUAL(5, divide(10, 2));

}

void test\_divide\_testcase3(void)

{

TEST\_ASSERT\_EQUAL(5, divide(5, 1));

}

void test\_divide\_testcase4(void)

{

TEST\_ASSERT\_EQUAL(-6, divide(-30, 5));

}

void test\_divide\_testcase5(void)

{

TEST\_ASSERT\_EQUAL(1, divide(2, 2));

}

/\*PERCENTAGE\*/

void test\_percentage(void)

{

TEST\_ASSERT\_EQUAL(100, percentage(1, 1));

}

void test\_percentage\_testcase2(void)

{

TEST\_ASSERT\_EQUAL (40, percentage(2,5));

}

void test\_percentage\_testcase3(void)

{

TEST\_ASSERT\_EQUAL(20, percentage(20, 100));

}

void test\_percentage\_testcase4(void)

{

TEST\_ASSERT\_EQUAL(40, percentage(2, 5));

}

void test\_percentage\_testcase5(void)

{

TEST\_ASSERT\_EQUAL(20, percentage(2, 10));

}

/\*GCD\*/

void test\_gcd(void)

{

TEST\_ASSERT\_EQUAL(1, gcd(1, 2));

}

void test\_gcd\_testcase2(void)

{

TEST\_ASSERT\_EQUAL (20, gcd(20, 40));

}

void test\_gcd\_testcase3(void)

{

TEST\_ASSERT\_EQUAL (1, gcd(35, 64));

}

void test\_gcd\_testcase4(void)

{

TEST\_ASSERT\_EQUAL (12, gcd(24, 36));

}

void test\_gcd\_testcase5(void)

{

TEST\_ASSERT\_EQUAL(1, gcd(2, 5));

}

/\*REMAINDER\*/

void test\_rem(void)

{

TEST\_ASSERT\_EQUAL(1, rem(1, 2));

}

void test\_rem\_testcase2(void)

{

TEST\_ASSERT\_EQUAL (0, rem(45,5));

}

void test\_rem\_testcase3(void)

{

TEST\_ASSERT\_EQUAL (2, rem(17,5));

}

void test\_rem\_testcase4(void)

{

TEST\_ASSERT\_EQUAL (2, rem(18,4));

}

void test\_rem\_testcase5(void)

{

TEST\_ASSERT\_EQUAL(1, rem(5, 4));

}

/\*POWER\*/

void test\_power(void)

{

TEST\_ASSERT\_EQUAL (1, power(1, 2));

}

void test\_power\_testcase2(void)

{

TEST\_ASSERT\_EQUAL (27, power(3,3));

}

void test\_power\_testcase3(void)

{

TEST\_ASSERT\_EQUAL (1, power(3,0));

}

void test\_power\_testcase4(void)

{

TEST\_ASSERT\_EQUAL (1, power(1,4));

}

void test\_power\_testcase5(void)

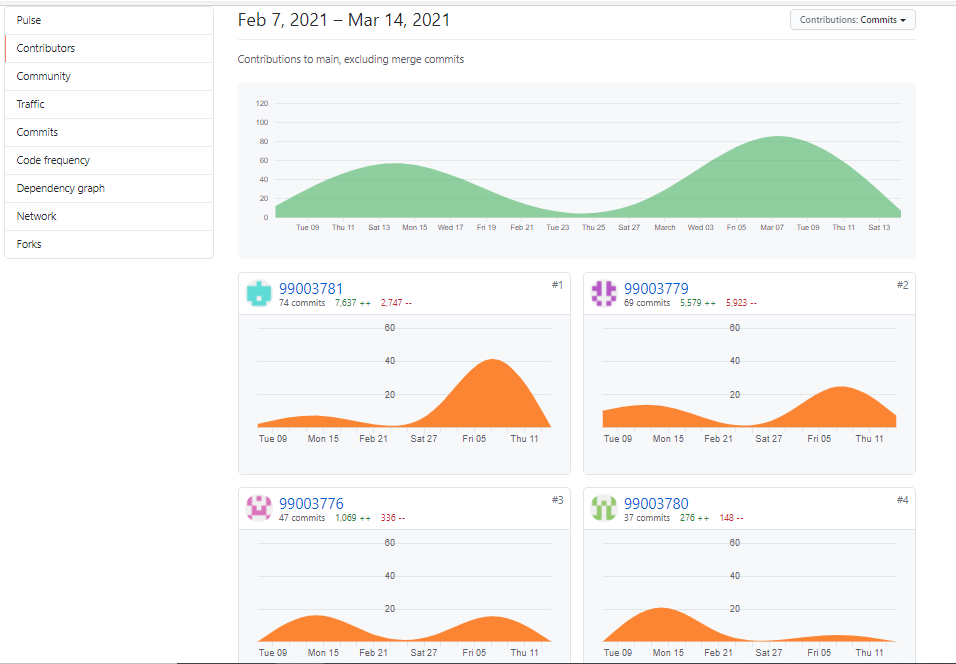
{

TEST\_ASSERT\_EQUAL(25, power(5, 2));

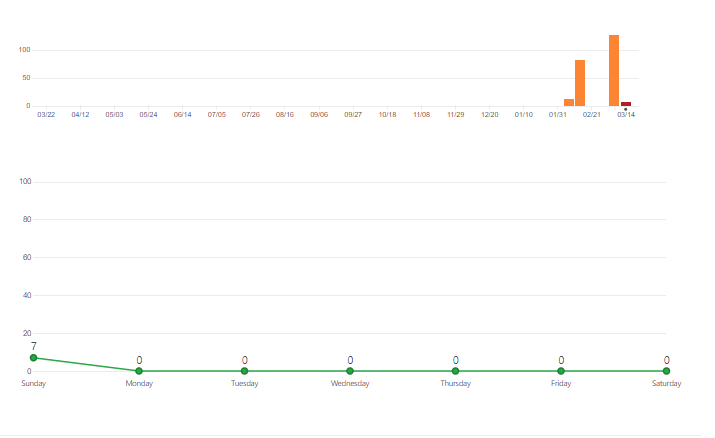
}

}

**Git commits:**



**Figure 10: Graph of commits on Github**



**Figure 11: Plot of Git Commits on github**