Laboratory Exercise No 2B:
Testing:
Prime Number Function

Prince Ngema (754774)
Tholithemba Mngomezulu (1512124)
Luyanda Makhoba (834867)
Takatso Molekane (569869)

Contents

1 Purpose

Ensuring that a system's faults have been eradicated is of most importance and this document describes how we have gone about the process of inspecting the PrimeNumbers function.

2 Test Program Description

The test program has been designed to detect and eliminate all possible bugs in the system. This has been achieved by listing all possible input variable types,both valid and invalid tests were conducted using exhaustive input testing types. Appropriate Error codes are returned for invalid input. Results of these tests were tabulated as comparisons were made with the expected results and simulation results were recorded.

2.1 Modification to parameters passed

- **Invalid parameters:**
  - String of characters
  - Negative integers
- **Valid parameters:**
  - Numbers that can be stored as long
  - positive Integers

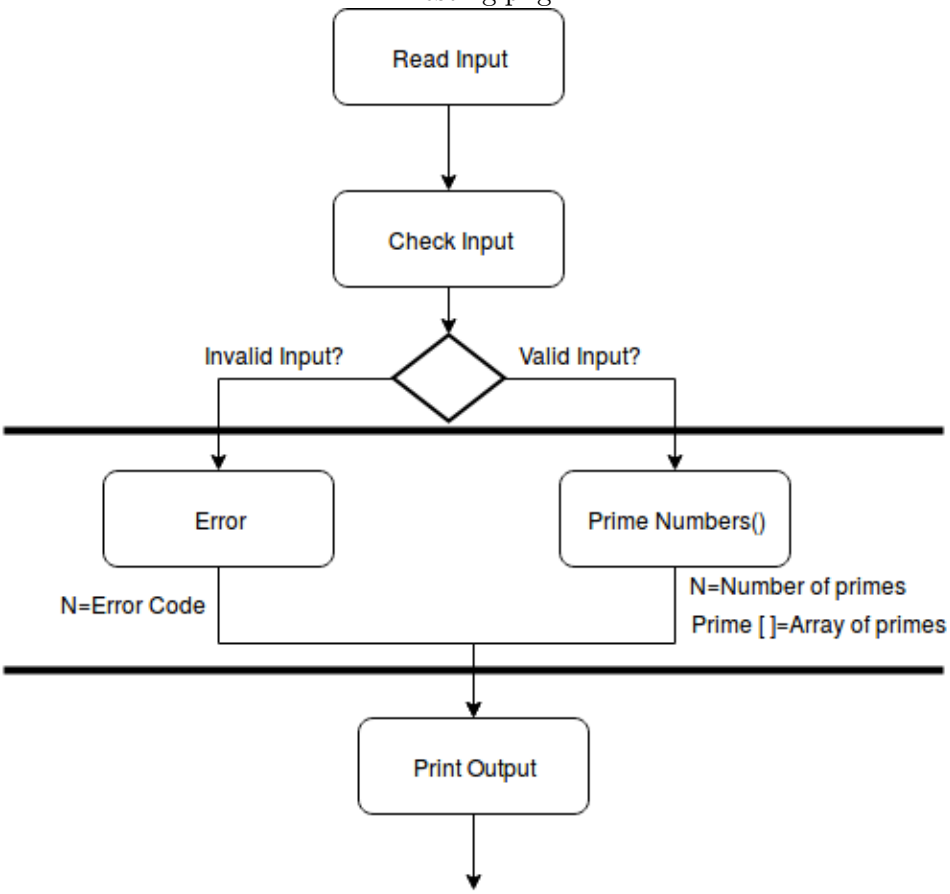
3 Program Module Testing

3.1 Unit Testing

Testing of individual components using white-box method to evaluate all units of code in the program.

Floor of testing events

Testing.png



3.1.1 Scenarios tested and results

Test case description	Test Data	Expected Result	Actual Result
user inputs a positive value	x= 10	4 primes and 1 3 5 7	pass
user inputs a negative value	x= -1	Negative Integer and Error Code 101-Negative int. Enter positive int	pass
user inputs a char value string	x = Hello	String of char and Error Code 102-String of character.Enter positive int0	pass
user inputs a char value string	x = 10iksj	String containing an Integer, Our program will work with the integer part and truncate the alphanumeric characters.	pass

3.1.2 White-box Testing

One approach used involved using data breakpoints to view the values of variables at different points of the code during iterations. Using an IDE such as Visual Studio one can track and see as soon as the content of a variable changes.

The internal structure of PrimeNumber()

3.2 Integration Testing

Integration testing is a kind of testing that is done by integrating all the methods and checking their interaction, for example observing that one methods output is read accordingly as the input of the relevant methods.
In our program, we have defined a function that takes in the input and then

validates it to check if it conforms. If the input is valid(Positive integer), it is then passed from the checkInput() model to the PrimeNumber() function. The PrimeNumber() method is called inside the checkInput() method if the conditions are met. Both the methods are tested separately to ensure that they yield desired results. Then the 2 methods are integrated, we've tested them again and got desired results. They both passed. We also checked to see that all method calls happened correctly and they passed these tests.
A common issue with integration testing involves database locking of data which was not a problem we encountered.

3.3 System Testing

Testing the overall program. Checks overall run-through of the system. The testProg() is used to automate and test various inputs as specified in the test program description above. The aim was to see of the primeNumbers() function returns the correct output of prime numbers. The output of our primeNumbers() function, given input x, is compared to a list of prime numbers that are known to be correct listed by the testPrime() function which gets the correct list of primes in an external file called prim_number.txt

System Testing was the last form of testing conducted, it followed integration testing, after all the system module had been integrated. This form of testing is only concerned with the inputs and outputs.

3.3.1 Black Box (Functional) testing

High level design done by independent testers.

Box Diagram.png