Nahrin Sharna

CIS 200L

# Program 1:

## Initial Test Plan

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Valid | User choose to find the root of a valid value | Choice = Y,  Value = 3.0 | 1.73214 |  |  |
| 2 | Invalid | User choose to find the root value of zero | Choice = y  Value = 0 | The program should stop. |  |  |
| 3 | Invalid | User choose to find the root of a negative value | Choice = y  Value = -3 | The program should stop |  |  |
| 4 | Invalid | User choose an invalid response | Choice = G | Display an error message |  |  |
| 5 | Valid | User choose not to find a root value | Choice = N | Display a greeting message |  |  |
| 6. | Invalid | User did not input an integer to find the root | Choice = Y,  Value = g | Display an error message |  |  |

## Source Code:

/\*

Author: Nahrin Sharna

Creation Date: 31 January, 2019

Modification Date: 01 January, 2019

Purpose: To display the approximate root value of an integer

\*/

#include<iostream>

#include <cassert>

using namespace std;

//function prototypes

void print\_number(double);

double squareRoot(double);

int main()

{

double value, root; // value = variable for input value of user

// root = variable for the root value of user input

char choice;

do {

cout << "Would you like to find the root of a value(Y or N): ";

cin >> choice;

cout << endl;

switch (toupper(choice)) {

case 'Y':

cout << "Please enter the value to find it's root: ";

cin >> value;

if (cin.fail()) { //Check for valid integer input

cout << "Invalid input" << endl; //Created by Spencer Wong on 2/15/18.

cin.clear(); //Copyright © 2018 Spencer Wong. All rights reserved.

cin.ignore(256, '\n');

}

else {

root = squareRoot(value); //function call

cout << "The root of " << value << " is: " << root << endl;

}

break;

case 'N' :

cout << "Thank you for using our software." << endl;

break;

default:

cout << "Sorry!!Invalid option has been chosen." << endl;

}

cout << endl;

} while (toupper(choice)!= 'N');

system("pause");

return 0;

}

void print\_number(double myInt) {

assert(myInt > 0);

}

double squareRoot(double x)

{

// assert that x is not negative

print\_number(x);

double x0 = x / 2;

double x1, x2;

do {

x1 = x / x0;

x2 = x0;

x0 = (x0 + x1) / 2;

} while (x0 - x1 < .0001); // the difference between x0 and x1 is less than .0001

return x0;

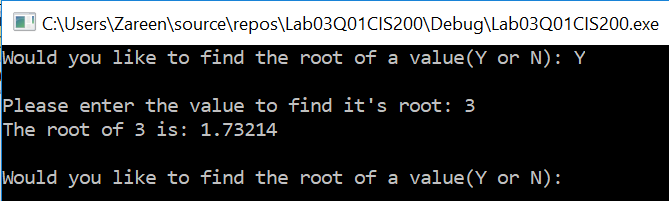
}

## Final Test Plan:

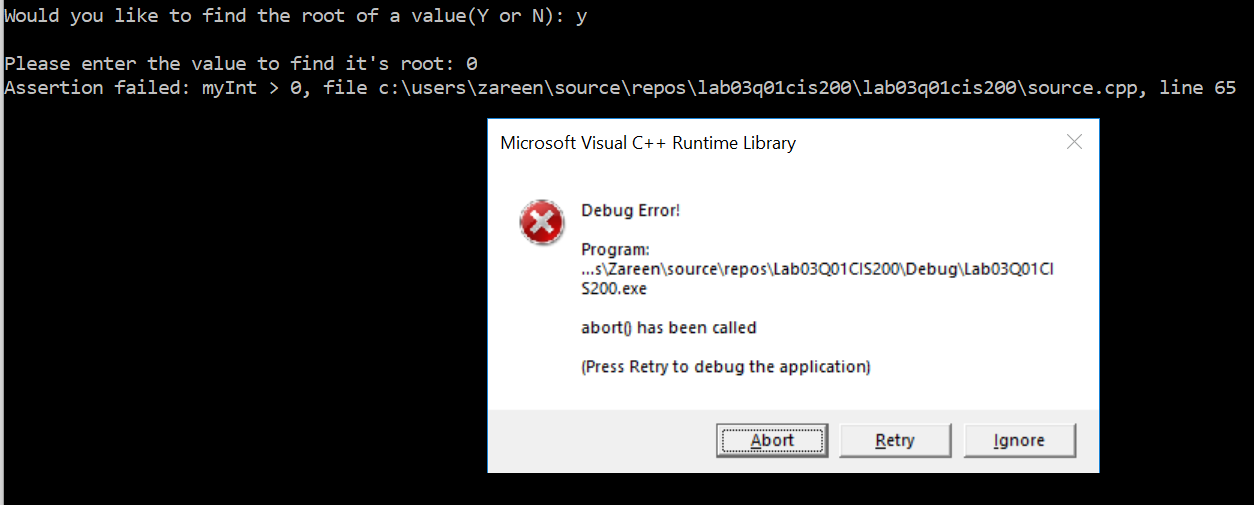
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Valid | User choose to find the root of a valid value | Choice = Y,  Value = 3 | 1.73214 | 1.73214 | Pass |
| 2 | Invalid | User choose to find the root value of zero | Choice = y  Value = 0 | The program should stop. | Assertion failed. The program stopped | Pass |
| 3 | Invalid | User choose to find the root of a negative value | Choice = y  Value = -3 | The program should stop | Assertion failed. The program stopped | Pass |
| 4 | Invalid | User choose an invalid response | Choice = G | Display an error message | “Sorry!! Invalid option has been chosen.” | Pass |
| 5 | Valid | User choose not to find a root value | Choice = N | Display a greeting message | “Thank you for using our software.” | Pass |
| 6 | Invalid | User did not input an integer to find the root | Choice = Y,  Value = g | Display an error message | “Invalid input” | Pass |

## Screenshots

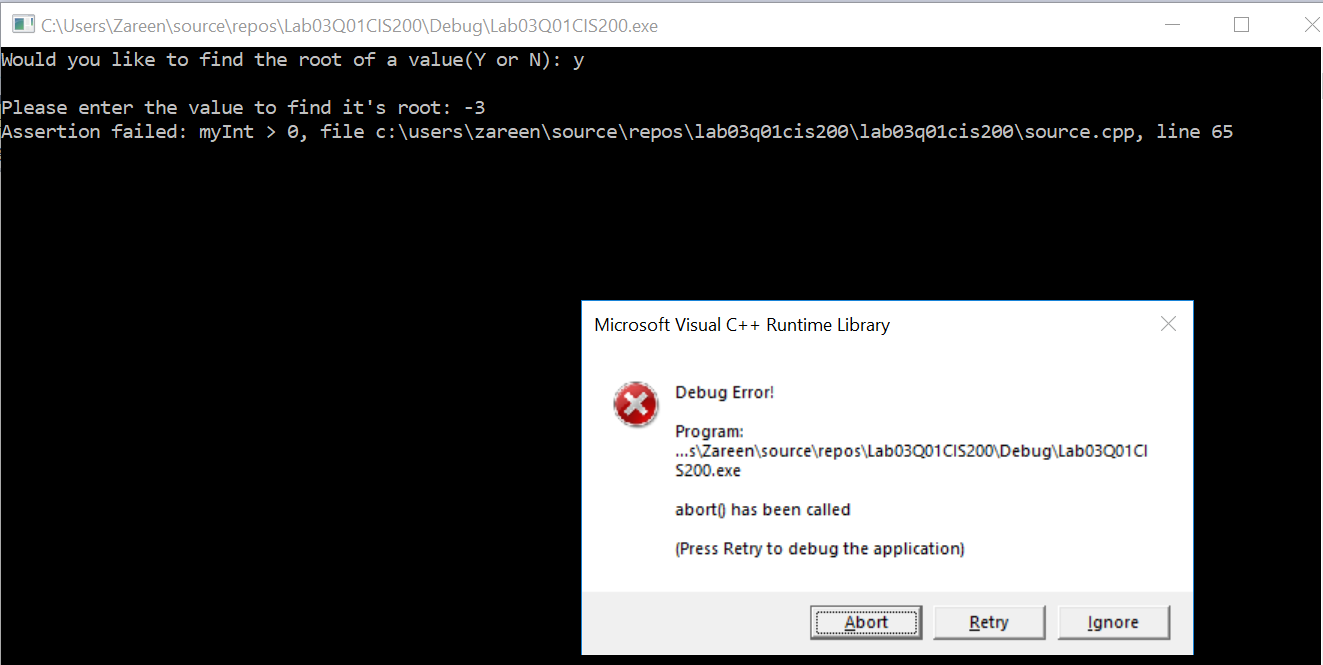
1. **Choice = Y, Value = 3**



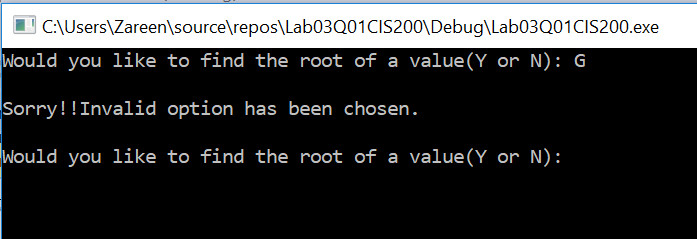
1. **Choice = y, Value = 0**



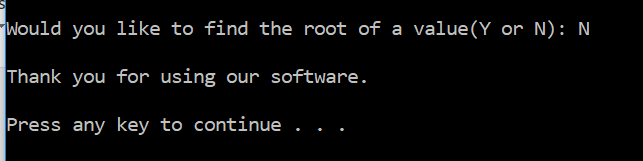
1. **Choice = y, Value = -3**



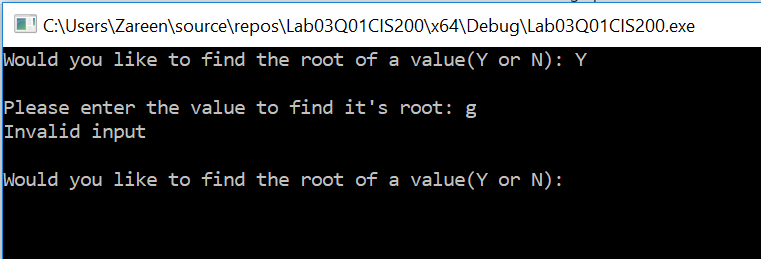
1. **Choice = G**



1. **Choice = N**



1. **Invalid Value:**



# Program 2:

## Initial Test Plan:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Invalid | When user enter invalid file name | Filename = “empty.txt” | Error message |  |  |
| 2 | Valid | File name is correct but file has no data | Filename = “program2.txt” | Error message and stop program through assertion |  |  |
| 3 | Invalid | File does not have integer values | Filename = “program2.txt”  Data: g e f | Invalid message |  |  |
| 4 | Invalid | File has more than 20 numbers (too many data) | Filename = “program2.txt”  Data: 12 4 2 5 6 7 2 4 6 8 3 4 12 2 4 5 6 7 8 12 45 23 | Error message saying too many datas |  |  |
| 5 | Valid | File exists and have sufficient data | Filename = “program2.txt”  Data: 12 4 2 5 6 7 2 4 6 8 3 4 | Print total numbers and display all those numbers |  |  |

## Source Code:

/\*Author: Nahrin Sharna

Creation Date: 02/01/2019

Modification Date: 02/03/2019

Purpose: To check the input file, display total numbers and store the numbers into an array

\*/

#include <iostream>

#include <string>

#include<fstream>

#include<cassert>

using namespace std;

//function prototypes

void readIntFile(ifstream &, int [], const int, int&);

void printArray(int [], int);

void emptyFile(ifstream& , int);

const int SIZE = 20;

void emptyFile(ifstream& in,int x) {

//assert when the file is empty

assert(x == in.eof());

}

int main() {

string fileName;

ifstream ins;

int a[SIZE];

int length, numbers;

cout << "Please enter the file name: ";

cin >> fileName;

ins.open(fileName.c\_str());

if (!ins.is\_open())

{

cout << "Sorry!! The file does not exist." << endl;

}

if (ins.peek() == EOF) {

cout << "Sorry!! The file is empty." << endl;

emptyFile(ins, ins.peek());

}

else {

readIntFile(ins, a, SIZE, length);

if (length < SIZE) {

printArray(a, length);

cout << endl;

}

}

system("pause");

return 0;

}

void readIntFile(ifstream &x, int intArray[], const int size, int& length) {

length = 0;

int numbers;

while (!x.eof())

{

x >> numbers;

if (x.fail()) {

cout << "Invalid Data." << endl;

x.clear();

x.ignore(256, '\n');

}

else {

intArray[length] = numbers;

length++;

}

}

if (length < SIZE)

cout << "Total Integer Numbers: " << length << endl;

else

cout << "Sorry!! Too many numbers for the array." << endl;

}

void printArray(int a[], int length) {

for (int i = 0; i < length; i++) {

cout << a[i] << " ";

}

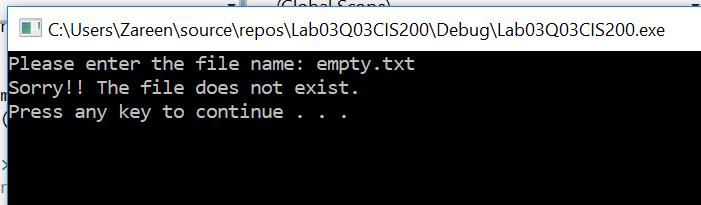
}

## Final Test Plan:

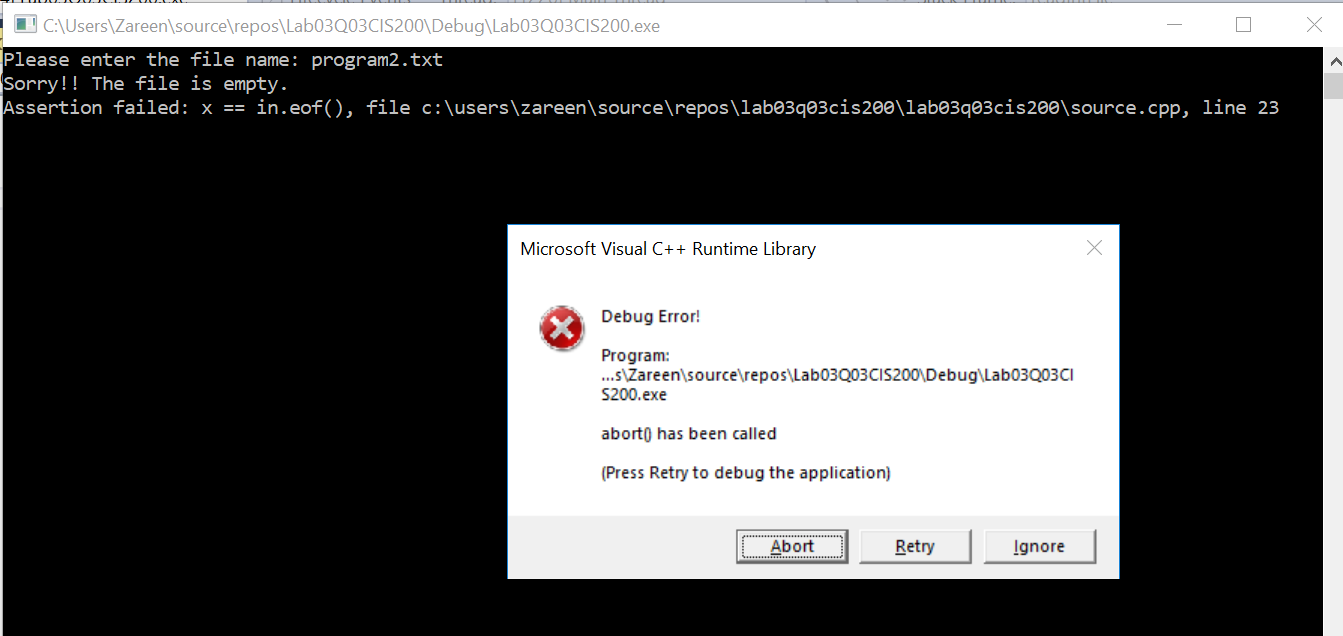
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Invalid | When user enter invalid file name | Filename = “empty.txt” | Error message | “Sorry!!The file does not exist.” | Pass |
| 2 | Valid | File name is correct but file has no data | Filename = “program2.txt” | Error message and stop program through assertion | “Sorry!! The file is empty.”  Assertion failed | Pass |
| 3 | Invalid | File does not have integer values | Filename = “program2.txt”  Data: g e f | Invalid message showing no integers | “Invalid Data.”  “Total Integer Numbers: 0” | Pass |
| 4 | Invalid | File has more than 20 numbers (too many data) | Filename = “program2.txt”  Data: 12 4 2 5 6 7 2 4 6 8 3 4 12 2 4 5 6 7 8 12 45 23 | Error message saying too many datas | “Sorry!! Too many numbers for the array.” | Pass |
| 5 | Valid | File exists and have sufficient data | Filename = “program2.txt”  Data: 12 4 2 5 6 7 2 4 6 8 3 4 | Print total numbers and display all those numbers | “Total Integer Numbers: 12  12 4 3 5 6 7 2 4 6 8 3 4” | Pass |

## Screenshots:

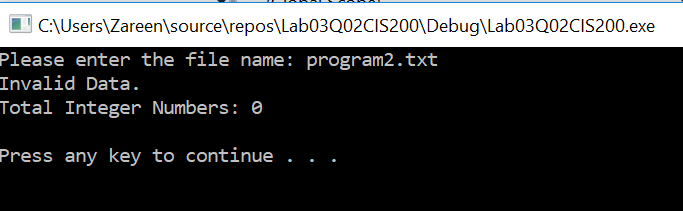
1. **Invalid File Name:**



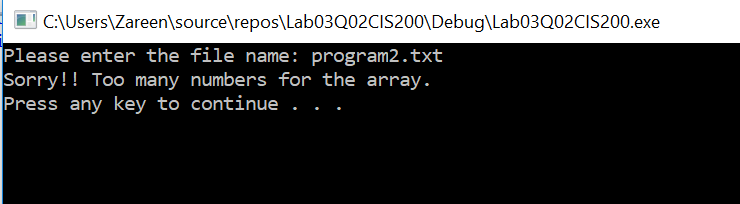
1. **Empty File:**



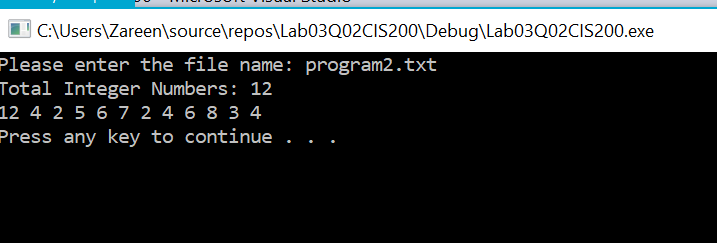
1. **Invalid Data:**



1. **Overloaded Array (Too many Data):**



1. **Valid File:**



# Program 3:

## Initial Test Plan:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Valid | String length is one | String = S | Print = “S” |  |  |
| 2 | Valid | String length is not one | String = Friend | Print = “dneirF” |  |  |

## Source Code:

/\*

Author:Nahrin Sharna

Creation Date: 02/02/2019

Modification Date: 02/02/2019

Purpose: To display a reverse string

\*/

#include <iostream>

#include <string>

using namespace std;

//function prototype

string reverseStringRecursively(string);

int main()

{

string str;

cout << "Enter the string to reverse : ";

cin >> str;

cout << "The reversed string is : "

<< reverseStringRecursively(str) << endl;

system("pause");

return 0;

}

string reverseStringRecursively(string s) {

if (s.length() == 1) {

return s;

}

else {

return reverseStringRecursively(s.substr(1, s.length())) + s.at(0);

}

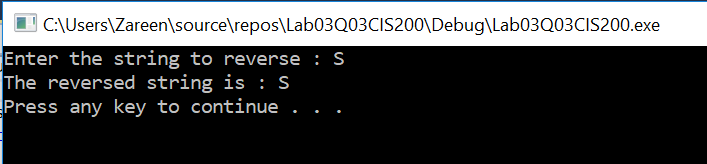
## }

## Final Test Plan:

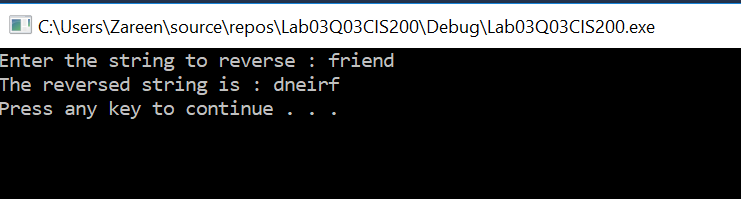
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Valid | String length is one | String = S | Print = “S” | “The reversed string is: S” | Pass |
| 2 | Valid | String length is not one | String = Friend | Print = “dneirF” | “The reversed string is: dneirF.” | Pass |

## Screenshots:

1. **String length is one:**



1. **String length is not one:**



# Program 4:

No test table necessary. List the three legs of the software development stool.

Ans: Three legs of the software development stool:

1. Unit testing

2.Functional testing

3.Performance testing