Kevin Riggs Module 1 Assignment WriteUp

Requirements:
Your program should read three integer values.
The three values are interpreted as representing the integer lengths of the sides of a triangle.
The program prints a message that states whether the triangle is equilateral (three side equal), isosceles (two side equal) or scalene (no side equal).

Design:
The main design decision I made is I wanted to be able to use a text file, mainly for testing, or the command line for giving inputs.
To handle these cases I want to be able to handle a filename and I want to accept a generic input stream.

Two more design decisions I want to call attention to: First is to verify that inputs are integers. The second is to catch if the user is failing to input correctly or if some other problem is occurring.

Last design decision is around testing. In order to use the test files, pass two arguments; to test catching an unknown exception, pass one argument; regular usage by a user would be to pass no arguments.

Source Code:

```
//main.cpp
//#include <QCoreApplication>
#include <iostream>
#include <ios>
#include <limits>
#include "triangletest.h"
#include <string>


int main(int argc, char *argv[])
{
	//QCoreApplication a(argc, argv);
	//return a.exec();
	try{
		if(argv[2]){
			std::string fileName = "scalene.txt";
			TriangleTest a = TriangleTest(fileName);
			fileName = "isosceles.txt";
			a = TriangleTest(fileName);
			fileName = "equilateral.txt";
			a = TriangleTest(fileName);
			fileName = "BadFile";
			a = TriangleTest(fileName);

		} else if(argv[1]){
			// Test catching some exception(Will not reach this test, move to top of try statement)
			throw "error";
		} else{
			TriangleTest a = TriangleTest();
		}

	} catch(std::string p){
		std::cout << p << std::endl;
		exit(EXIT_SUCCESS);
	} catch(...){
		std::cout << "Some exception caught" << std::endl;
```

```
                exit(EXIT_FAILURE);
        }
}

//triangletest.h
#ifndef TRIANGLETEST_H
#define TRIANGLETEST_H
#include <iostream>
#include <ios>
#include <limits>
#include <string>

class TriangleTest
{
public:
        TriangleTest(std::string fileName);
        TriangleTest();
        int getSide(std::istream& input);
};

#endif // TRIANGLETEST_H


//triangletest.cpp
#include "triangletest.h"
#include <fstream>

TriangleTest::TriangleTest(std::string fileName)
{
        int side1 = 0, side2 = 0, side3 = 0;

        std::ifstream ifs;
        ifs.open(fileName, std::ifstream::in);
        side1 = getSide(ifs);
        side2 = getSide(ifs);
        side3 = getSide(ifs);

        if(side1 == side2 && side2 == side3)
        {
                std::cout << "Triangle is equilateral" << std::endl;
        }
        else if (side1 == side2 || side1 == side3 || side2 == side3)
        {
                std::cout << "Triangle is isosceles" << std::endl;
        }
        else
        {
                std::cout << "Triangle is scalene" << std::endl;
        }
}

TriangleTest::TriangleTest()
{
        int side1 = 0, side2 = 0, side3 = 0;

        side1 = getSide(std::cin);
        side2 = getSide(std::cin);
        side3 = getSide(std::cin);
```

```
        if(side1 == side2 && side2 == side3)
        {
                std::cout << "Triangle is equilateral" << std::endl;
        }
        else if (side1 == side2 || side1 == side3 || side2 == side3)
        {
                std::cout << "Triangle is isosceles" << std::endl;
        }
        else
        {
                std::cout << "Triangle is scalene" << std::endl;
        }
}

int TriangleTest::getSide(std::istream& input)
{
        int side = 0;
        int i = 0;
        std::cout << "Enter a postive integer" << std::endl;
        while(i < 3){
                if(!(input >> side)){
                        std::cout << side << std::endl;
                        input.clear();
                        input.ignore(std::numeric_limits<std::streamsize>::max(),'\n');
                }
                if(side > 0){
                        input.clear();
                        input.ignore(std::numeric_limits<std::streamsize>::max(),'\n');
                        return side;
                }
                else{
                        std::cout << "Bad input try again" << std::endl;
                }
                i++;
        }
        throw std::string("Too many bad inputs, quitting");
}


// Test input files
//equilateral.txt
1
1
1

//isosceles.txt
1
1
2

//scalene.txt
1
2
3

Test Results
// Using test files and bad input test, pass two arguments
C:\projects\build\build-TriangleTest-Desktop_Qt_5_13_1_MSVC2017_32bit-
Profile>release\TriangleTest.exe a a
```

```
Enter a postive integer
Enter a postive integer
Enter a postive integer
Triangle is scalene
Enter a postive integer
Enter a postive integer
Enter a postive integer
Triangle is isosceles
Enter a postive integer
Enter a postive integer
Enter a postive integer
Triangle is equilateral
Enter a postive integer
0
Bad input try again
0
Bad input try again
0
Bad input try again
Too many bad inputs, quitting

// Unknown exception test, pass one argument
C:\projects\build\build-TriangleTest-Desktop_Qt_5_13_1_MSVC2017_32bit-
Profile>release\TriangleTest.exe a
Some exception caught
```