Writing Test Cases II

Team 16:

Raghav Gupta

Chelsea Jaculina

Bala Nyan Kwaw

Mike Lee

Yihua Li

May 6th, 2018

Professor Badari

CMPE 187

# Table of Contents

[**Table of Contents**](#_fvgztqx5fz61) **1**

[**Overview**](#_98626yde1lui) **2**

[Introduction](#_v8p54qc9bvlq) 2

[Purpose](#_swv4r9obd9id) 2

[**Oracle**](#_5u470bcj8tu8) **2**

Test Oracle 2

[Environment](#_hxp4hhr1f9w7) 2

[Testing Techniques](#_t900sory5c5w) 3

[**Control Flow Testing**](#_fnb1o5bc1blo) **4**

[**Domain Testing**](#_8cebhn3ggjvr) **5**

[Inputs](#_twb9ilwu258p) 5

[Output](#_7xfe2srenesk) 5

[Test Cases](#_9hiblnygl9q) 5

[Input Domain Diagram](#_27f2eociscxc) 7

[**Equivalent Class Testing**](#_k1jetht8ninc) **8**

[**Decision Tables**](#_jkbskanzqlc3) **9**

[**Conformance Testing**](#_6ubw2ipcd19h) **10**

[**Feature Testing**](#_ns7wqckz34l2) **11**

[**Lessons Learned**](#_l7uh6ydbadvf) **12**

# 

# Overview

## Introduction

To introduce our project we will be reusing the triangle program that we tested in the very beginning of this class. The background of the project is that it will check if the inputs that has been entered by the user are valid or not. The program will have the ability to check if any string, character, or negative numbers, and reject them as an input. If the input is not valid, the user will be prompt again to enter 3 new valid positive integers. The overall program will check if the 3 input values will form a triangle or not, and let the user know if the values create an equilateral, scalene, or an isosceles triangle.

## Purpose

The purpose of this project is to see if we can apply as many techniques to our simple triangle program that we started off with at the beginning of the semester.

# Oracle

### Test Oracle

## Environment

The hardware used for programming and testing is an iPhone running the operating system of iOS 11.3. The program used for creating the State diagram was draw.io (a plugin by Google) and Code2flow ( <https://code2flow.com/app> ) for verification of the diagram through a third party software.

## 

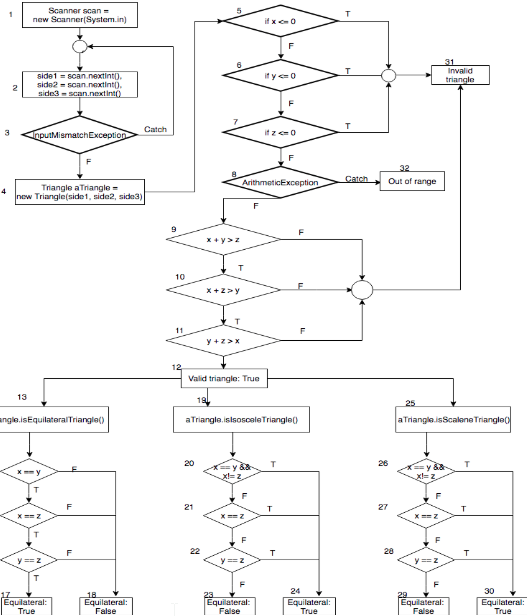
# Testing Techniques

1. Mutation Testing (*Not Chosen*)
2. Control Flow
3. Data Flow *(Not Applicable)*
4. Domain Testing
5. Equivalent Class Testing
6. Decision Tables
7. Conformance Testing
8. Feature Testing (*Not Applicable*)

# 

# Control Flow Testing

## Control Flow Diagram



**Figure 1**. Control Flow Diagram of Triangle Program

# Domain Testing

## Inputs

The inputs of the triangle program will be provided by the user in which they will enter three values for each side of the triangle. The user can input any integer value between 1 to 2147483647, which is the positive range for an integer in Java.

## Output

The output of this program will consist of two parts. First, the program will validate if the given lengths can build a valid triangle or not (String of True/False). Secondly, if the given lengths construct a valid triangle, the program will determine what type of triangle it is depending on the value of the lengths (True/False of “Equilateral”, “Isosceles”, “Scalene”).

# Equivalent Class Testing

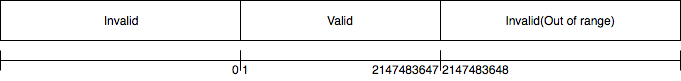
## Equivalent Classes

|  |  |
| --- | --- |
| **Triangle** | **Property** |
| Equilateral | Triangle has three equal sides and three equal angles.  Each angle is 60° |
| Isosceles | Triangle has **two** equal sides and two equal angles. |
| Scalene | Triangle has no congruent sides. In other words, each side must have a different length |

# Boundaries

|  |  |
| --- | --- |
| **Input** | **Boundaries** |
| Side A | 1 - 2,147,483,647 |
| Side B | 1 - 2,147,483,647 |
| Side C | 1 - 2,147,483,647 |

# 

[](https://www.draw.io/#G1jXtZNlA43WZoQaOSbhauZig2kK4geuLG)

**Figure 2**. Input Domain Diagram for Triangle Program

## Test Cases

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Purpose** | **Input** | **Expected Output** | **Actual Output** | **Pass/Fail** |
| 1 | Test first length domain | 0, 1, 1 | Invalid length | Invalid length | Pass |
| 2 | Test first length domain | 2147483648, 2, 2 | Invalid length | Invalid length | Pass |
| 3 | Test first length domain | 1, 2, 2 | Valid triangle,  Isosceles | Valid triangle,  Isosceles | Pass |
| 4 | Test second length domain | 1, 0, 1 | Invalid length | Invalid length | Pass |
| 5 | Test second length domain | 3, 2147483648, 2 | Invalid length | Invalid length | Pass |
| 6 | Test second length domain | 3, 1, 4 | Valid triangle,  Isosceles | Valid triangle,  Isosceles | Pass |
| 7 | Test third length domain | 2, 3, 0 | Invalid length | Invalid length | Pass |
| 8 | Test third length domain | 2, 3,  2147483648 | Invalid length | Invalid length | Pass |
| 9 | Test third length domain | 2, 3, 1 | Invalid triangle | Invalid triangle | Pass |
| 10 | Test valid triangle domain | 1, 1, 2 | Invalid triangle | Invalid triangle | Pass |
| 11 | Test valid triangle | 2, 4, 5 | Valid triangle,  Scalene | Valid triangle,  Scalene | Pass |
| 12 | Test valid equilateral triangle | 0, 0, 0 | Invalid length | Invalid length | Pass |
| 13 | Test valid equilateral triangle | 1, 1, 1 | Valid triangle,  Equilateral | Valid triangle,  Equilateral | Pass |
| 14 | Test Valid equilateral triangle | 2, 2, 3 | Valid triangle,  Isosceles(not equilateral) | Valid triangle,  Isosceles | Pass |
| 15 | Test Valid equilateral triangle | 3, 4, 5 | Valid triangle,  Scalene(not equilateral) | Valid triangle,  Scalene | Pass |
| 16 | Test valid isosceles | 2, 2, 4 | Invalid triangle | Invalid triangle | Pass |
| 17 | Test valid isosceles | 3, 3, 5 | Valid triangle,  Isosceles | Valid triangle,  Isosceles | Pass |
| 18 | Test Valid isosceles | 1, 1, 1 | Valid triangle,  Equilateral, | Valid triangle,  Equilateral | Pass |
| 19 | Test Valid isosceles | 3, 5, 7 | Valid triangle,  Scalene(not Isosceles) | Valid triangle,  Scalene | Pass |
| 20 | Test Valid scalene | 2, 2, 2 | Valid triangle,  Equilateral(not scalene) | Valid triangle,  Equilateral | Pass |
| 21 | Test Valid scalene | 3, 3, 5 | Valid triangle,  (not scalene) | Valid triangle, | Pass |
| 22 | Test Valid scalene | 3, 4, 5 | Valid triangle,  Scalene | Valid triangle,  Scalene | Pass |

# Decision Tables

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Conditions** | **Values** | **Rules or Combinations** | | | | | | | |  |  |  |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** |
| a < b + c? | **T, F, -** | **F** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** |
| b < a + c? | **T, F, -** | **-** | **F** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** |
| c < a + b? | **T, F, -** | **-** | **-** | **F** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** |
| A = B? | **T, F, -** | **-** | **-** | **-** | **T** | **T** | **T** | **T** | **F** | **F** | **F** | **F** |
| A = C? | **T, F, -** | **-** | **-** | **-** | **T** | **T** | **F** | **F** | **T** | **T** | **F** | **F** |
| B = C? | **T, F, -** | **-** | **-** | **-** | **T** | **F** |  | **F** |  | **F** |  | **F** |
| **Actions** |  |  |  |  |  |  |  |  |  |  |  |  |
| Not a Triangle | **T, F, -** | **X** | **X** | **X** |  |  |  |  |  |  |  |  |
| Scalene | **T, F, -** |  |  |  |  |  |  |  |  |  |  | **X** |
| Isosceles | **T, F, -** |  |  |  |  |  | **X** |  |  | **X** | **X** |  |
| Equilateral | **T, F, -** |  |  |  | **X** |  |  |  |  |  |  |  |
| Impossible | **T, F, -** |  |  |  |  | **X** | **X** |  | **X** |  |  |  |
| **Checksum** |  | 32 | 16 | 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

# Conformance Testing

## Description of the Problem

The goal of this assignment is to check if three integer input makes an isosceles, equilateral or scalene triangle.

## Business Rules

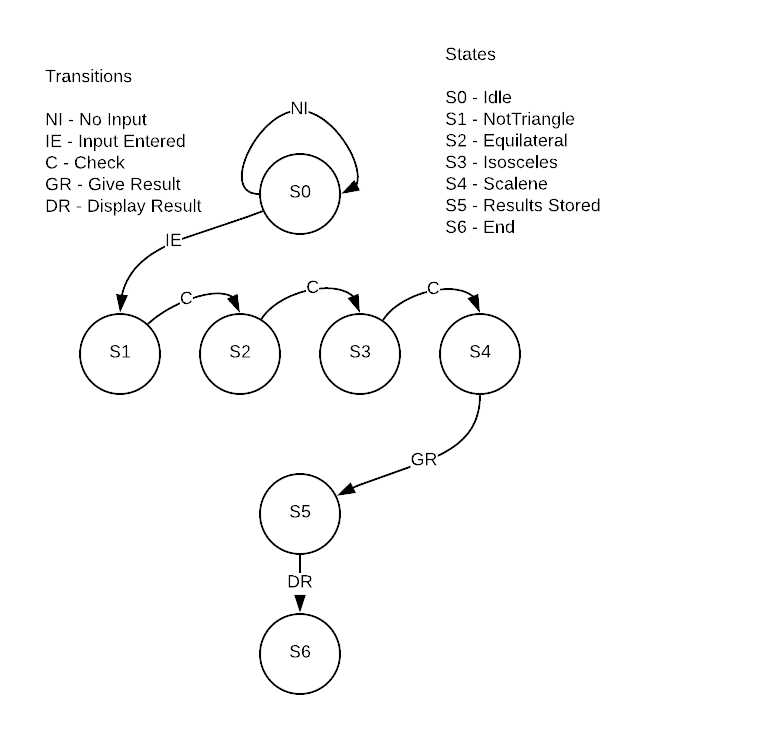
1. The program will take three integers x, y and z
2. The program will determine if a triangle is an equilateral, isosceles, scalene or a combination of one of them

## States and Transitions

|  |  |  |
| --- | --- | --- |
| Transitions | | |
| Symbol | Name | Definition |
| NI | No Input | Nothing is entered |
| IE | Integer Entered | Three integers entered |
| DR | Display Result | Result determined and printed on screen |

|  |  |  |
| --- | --- | --- |
| States | | |
| Symbol | Name | Definition |
| S0 | Idle | Program is waiting for input |
| S1 | NotTriangle | Inputs are not a triangle |
| S2 | Equilateral | Inputs are equilateral triangle |
| S3 | Isosceles | Inputs are isosceles triangle |
| S4 | Scalene | Inputs are scalene triangle |
| S5 | End | Program ends |

## Finite State Machine



**Triangle FSM**

## Transition Tours

|  |  |
| --- | --- |
| **Transition Tour #** | **Sequence** |
| 1 | **Idle** — Input Entered → **NotTriangle** — Check → **Equilateral** — Check → **Isoceles** — Check → **Scalene** — Give Results → **Results Stored** — Display Results → **End** |

## Exceptions

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Reason** | **Scenario** | **Input** | **Expected** | **Actual** | **Pass/Fail** | **Sequence** |
| 1. | Abnormal Input | Inputs are unnatural numbers that are not part of the domain | -1 | Invalid | Invalid | Pass | S0-S1 |
| 2. | Gigantic Input | Inputs are too large and computation takes forever | 9 x 1099 | Doesn’t end | Doesn’t end | Pass | S0-S1  S0-S2  S0-S3  S0-S4 |
| 3. | Undetermined Result | Inputs are mathematically uncomputable like zero divided by zero | 0,1,0 | Exception | Exception | Pass | S0-S4 |

# Lessons Learned

At the start of the class, the professor had mentioned how we will continue our first assignment as our last assignment. The goal of such a task was to implement our knowledge learned in class to gauge our readiness for testing real world applications.

In this particular project, our team learned how control flow testing, domain testing, equivalence class testing, boundary values, decision tables, and conformance testing were all techniques that were applicable to our simple triangle program.