Министерство образования и науки Российской Федерации

|  |  |
| --- | --- |
|  | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования | |
| «Новосибирский государственный технический университет» | |
| Факультет автоматики и вычислительной техники  Кафедра АСУ | |

ОТЧЁТ

по лабораторной работе №2

Тестирование по методу «Черного ящика»

по дисциплине: Технология программирования

Проверил:

к.т.н., доцент

Астапчук В.А.

Выполнили:

Студенты гр. АВТ-612, АВТФ

Савинцев А.С,

Ларионова Д.Н,

Новосибирск

2018

1. Задание

Разработать программу определения вида треугольника, заданного длинами его сторон: равносторонний, равнобедренный, прямоугольный, разносторонний.

1. Разработка тестов по методике «черного ящика»
   1. Эквивалентное разбиение

2.1.1 Исходные данные

|  |  |  |
| --- | --- | --- |
| Входное условие | Допустимые классы эквивалентности | Недопустимые классы эквивалентности |
| Тип вводимых данных | 3 вещественных числа в диапазоне  0 < X < 2147483647 (1) | Нецифровой символ (4),  X<=0 (5), X>=2147483647 (6) |
| 3 целых числа в диапазоне  0 < X < 2147483647 (2) |
| Комбинация вещественных и целых чисел в диапазоне  0 < X < 2147483647 (3) |

2.1.2 Тесты

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Назначение теста | Значения исходных данных | Ожидаемый результат | Реакция программы | Вывод |
| 1 | Проверка для вещественных чисел | 1.5  54.3  67.25 | Разносторонний | Разносторонний | Тест пройден успешно |
| 2 | Проверка для целых чисел | 3  4  3 | Равнобедренный | Равнобедренный | Тест пройден успешно |
| 3 | Проверка для комбинации целых и вещественных чисел | 123.5  55  62 | Разносторонний | Разносторонний | Тест пройден успешно |
| 4 | Проверка для нецифровых символов | 9  sorokDva  8 | Ошибка. Введеные значения не яляются числами. | Ошибка. Введеные значения не яляются числами. | Тест пройден успешно |
| 5 | Проверка на выход за минимум | -5  6  0 | Ошибка. Введеные значения выходят из допустимого диапазона снизу. | Ошибка. Введеные значения выходят из допустимого диапазона снизу. | Тест пройден успешно |
| 6 | Проверка на выход за максимум | 2  2147483647  4 | Ошибка. Введеные значения выходят из допустимого диапазона сверху. | Ошибка. Введеные значения выходят из допустимого диапазона сверху. | Тест пройден успешно |

* 1. Анализ граничных значений

2.2.1 Тесты

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Назначение теста | Значения исходных данных | Ожидаемый результат | Реакция программы | Вывод |
| 7 | Проверка на нулевые условия | 0  0  0 | Ошибка. Введеные значения выходят из допустимого диапазона снизу. | Ошибка. Введеные значения выходят из допустимого диапазона снизу. | Тест пройден успешно |
| 8 | Проверка на погрешность | 0.001  0.001  0.001 | Равносторонний | Равносторонний | Тест пройден успешно |
| 9 | Проверка на выход за максимум | 2147483647  6  88 | Ошибка. Введеные значения выходят из допустимого диапазона сверху. | Ошибка. Введеные значения выходят из допустимого диапазона сверху. | Тест пройден успешно |

* 1. Анализ причинно-следственных связей

2.3.1 Таблица истинности

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A<=0 | B<=0 | C<=0 | A=B=C | A=B≠C,  A≠B=C,  A=C≠B | A2+B2=C2 | Результат |
| 1 | X | X | X | X | X | Не треугольник |
| X | 1 | X | X | X | X | Не треугольник |
| X | X | 1 | X | X | X | Не треугольник |
| 0 | 0 | 0 | 1 | 0 | 0 | Равносторонний |
| 0 | 0 | 0 | 0 | 1 | X | Равнобедренный |
| 0 | 0 | 0 | 0 | 0 | 1 | Прямоугольный |
| 0 | 0 | 0 | 0 | 0 | 0 | Разносторонний |

2.3.2 Тесты

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Назначение теста | Значения исходных данных | Ожидаемый результат | Реакция программы | Вывод |
| 10 | Проверка для неположительных  значений | 0  -5  -6 | Ошибка. Введеные значения выходят из допустимого диапазона снизу. | Ошибка. Введеные значения выходят из допустимого диапазона снизу. | Тест пройден успешно |
| 11 | Проверка на равносторонний треугольник | 3  3  3 | Равносторонний | Равносторонний | Тест пройден успешно |
| 12 | Проверка на равнобедренный треугольник | 6  6  4 | Равнобедренный | Равнобедренный | Тест пройден успешно |
| 13 | Проверка на прямоугольный треугольник | 4  5  3 | Прямоугольный | Прямоугольный | Тест пройден успешно |
| 14 | Проверка на разносторонний треугольник | 6  12  42 | Разносторонний | Разносторонний | Тест пройден успешно |

3 Вывод

Тестирование по стратегии «чёрного ящика» происходит без рассмотрения внутреннего устройства тестируемого объекта.

* Метод эквивалентного разбиения обнаруживает ошибки при несоответствии

входных значений входным условиям. Недостатком является неполная проверка входных значений на всевозможные комбинации.

* Метод анализа граничных значений позволяет избежать появления ошибок при учете граничных значений входных данных. Недостатком является то, что метод не проверяет комбинации входных значений.
* Метод анализа причинно-следственных связей дает информацию о результате при различных комбинациях входных данных. Недостатком является громоздкость при достаточно большом количестве входных данных и плохое исследование граничных условий.

4 Код программы

#include <iostream>

#include <string>

using namespace std;

#define MAX\_VALUE 2147483647

#define MAX\_VALUE\_LENGTH 10

#define ErrNotNum "Ошибка. Введеные значения не яляются числами."

#define ErrAOutOfRangeMax "Ошибка. Введеные значения выходят из допустимого диапазона сверху."

#define ErrAOutOfRangeMin "Ошибка. Введеные значения выходят из допустимого диапазона снизу."

#define TriangleEquilateral "Равносторонний"

#define TriangleIsosceles "Равнобедренный"

#define TriangleRectangular "Прямоугольный"

#define TriangleVersatile "Разносторонний"

string determineAppearanceTriangleOnSides(string Astr, string Bstr, string Cstr)

{

// определим длины чисел

int Alength = Astr.length();

int Blength = Bstr.length();

int Clength = Cstr.length();

// проверка на нецифровые символы

if (!atof(Astr.c\_str()) && Astr != "0" || !atof(Bstr.c\_str()) && Bstr != "0" || !atof(Cstr.c\_str()) && Cstr != "0")

{

return ErrNotNum;

}

// замена , на .

for (int i = 0; i < Alength; i++)

{

if (Astr[i] == ',')

{

Astr[i] = '.';

break;

}

}

for (int i = 0; i < Blength; i++)

{

if (Bstr[i] == ',')

{

Bstr[i] = '.';

break;

}

}

for (int i = 0; i < Clength; i++)

{

if (Cstr[i] == ',')

{

Cstr[i] = '.';

break;

}

}

// проверка выхода за максимум по длине

bool isDotFindInA = false;

bool isDotFindInB = false;

bool isDotFindInC = false;

for (int i = 0; i < Alength; i++)

{

if (Astr[i] == '.')

{

isDotFindInA = true;

}

if (i > MAX\_VALUE\_LENGTH && !isDotFindInA)

return ErrAOutOfRangeMax;

}

for (int i = 0; i < Blength; i++)

{

if (Bstr[i] == '.')

{

isDotFindInB = true;

}

if (i > MAX\_VALUE\_LENGTH && !isDotFindInB)

return ErrAOutOfRangeMax;

}

for (int i = 0; i < Clength; i++)

{

if (Cstr[i] == '.')

{

isDotFindInC = true;

}

if (i > MAX\_VALUE\_LENGTH && !isDotFindInC)

return ErrAOutOfRangeMax;

}

// перевод длин из строк в числа

double A = atof(Astr.c\_str());

double B = atof(Bstr.c\_str());

double C = atof(Cstr.c\_str());

// проверка выхода за минимум

if (A <= 0 || B <= 0 || C <= 0)

{

return ErrAOutOfRangeMin;

}

// проверка выхода за максимум по числу

if (A >= MAX\_VALUE || B >= MAX\_VALUE || C >= MAX\_VALUE)

{

return ErrAOutOfRangeMax;

}

// вазврат вида треугольника

if (A == B && A == C)

{

return TriangleEquilateral;

}

else if (A == B || B == C || A == C)

{

return TriangleIsosceles;

}

else if (A\*A + B\*B == C\*C || A\*A + C\*C == B\*B || B\*B + C\*C == A\*A)

{

return TriangleRectangular;

}

else

{

return TriangleVersatile;

}

}

void main()

{

setlocale(LC\_CTYPE, "Russian");

cout << "Определить вид треугольника по его длинам." << endl;

cout << determineAppearanceTriangleOnSides("1.5", "54.3", "67.25") << endl;

cout << determineAppearanceTriangleOnSides("3", "4", "3") << endl;

cout << determineAppearanceTriangleOnSides("123.5", "55", "62") << endl;

cout << determineAppearanceTriangleOnSides("9", "sorokDva", "8") << endl;

cout << determineAppearanceTriangleOnSides("-5", "6", "0") << endl;

cout << determineAppearanceTriangleOnSides("2", "2147483647", "4") << endl;

cout << determineAppearanceTriangleOnSides("0", "0", "0") << endl;

cout << determineAppearanceTriangleOnSides("0.001", "0.001", "0.001") << endl;

cout << determineAppearanceTriangleOnSides("2147483647", "6", "88") << endl;

cout << determineAppearanceTriangleOnSides("0", "0", "0") << endl;

cout << determineAppearanceTriangleOnSides("-4", "-5", "-6") << endl;

cout << determineAppearanceTriangleOnSides("3", "3", "3") << endl;

cout << determineAppearanceTriangleOnSides("6", "6", "4") << endl;

cout << determineAppearanceTriangleOnSides("4", "5", "3") << endl;

cout << determineAppearanceTriangleOnSides("6", "12", "42") << endl;

system("pause");

}