|  |  |  |  |
| --- | --- | --- | --- |
| Министерство науки и высшего образования Российской Федерации | | | |
| ФГБОУ ВО «Брянский государственный инженерно-технологический  университет» | | | |
|  | | | |
| Кафедра «ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ» | | | |
|  | | | |
| ЛАБОРАТОРНАЯ РАБОТА №1  Разработка и оценка эффективности набора тестов ПО | | | |
| по дисциплине | | | |
| **Тестирование программного обеспечения** | | | |
|  | | | |
|  | | | |
|  | | | |
| Автор работы: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись) | Луговой Н.С.  (И.О. Фамилия студента) | |
| Группа: | ИВТ-401  (шифр группы) | № зачетной книжки 21-2.033 | |
| Преподаватель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | к.э.н. Доцент  (учен. степ, учен. звание) | О.Н. Юркова  (И.О. Фамилия) |
|  |  |  | |

Брянск 2024

## СОДЕРЖАНИЕ

[ЗАДАНИЕ 3](#_Toc176774927)

[ХОД ЛАБОРАТОНОЙ РАБОТЫ 5](#_Toc176774928)

[ЗАКЛЮЧЕНИЕ 15](#_Toc176774929)

## ЗАДАНИЕ

**Цель:** получить навыки в составлении набора тестов и оценить их полноту.

**Задание 1:** изучить статью

**http://dit.isuct.ru/Publish\_RUP/core.base\_rup/guidances/guidelines/test\_case\_81FD1D9F.html**

**Задание 2:** написать небольшую программу, которая представляет собой три поля, в которые вводятся целые числа – это длины сторон треугольника. После того, как нажата кнопка «Проверить», выводится сообщение о том, какой это треугольник: равнобедренный, равносторонний или разносторонний.

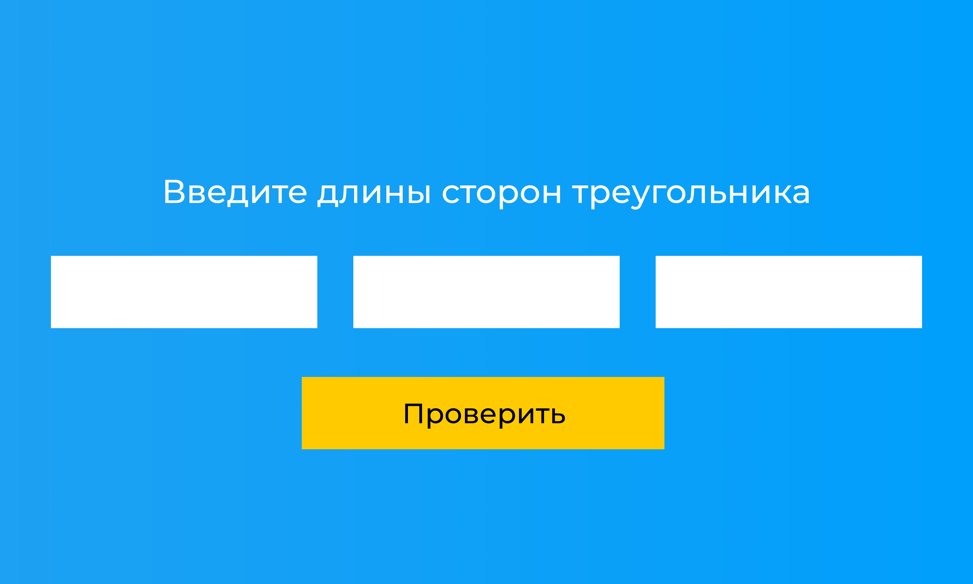
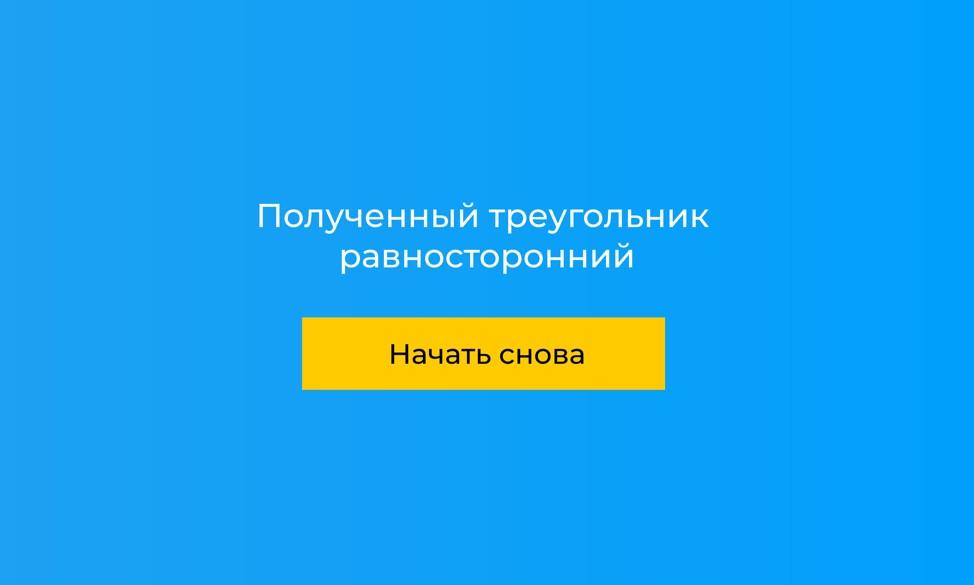
Рисунок 1 – Пример программы для тестирования

Рисунок 2 – Пример программы для тестирования

**Задание 3:** составить набор тестов, то есть набор входных данных, которые, по вашему мнению, правильно проверят эту программу.

**Задание 4:** оценка эффективности проверки. Оцените ваш набор тестов.

**Задание 5:** необходимо протестировать веб-приложение (рис. 3).

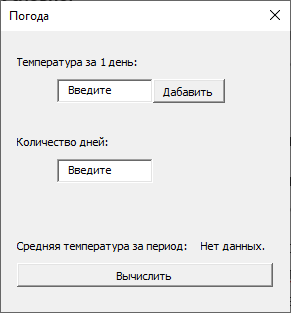


Рисунок 3 - Веб-приложение

Для полей ввода данных в приложении известны ограничения:

* Диапазон значений для поля *температура за 1 день*[-65; 65]
* Диапазон значений для поля *количество дней*[0; 1 000]

Проверка введенных значений и расчет выполняются на сервере. В случае успешной проверки данных и корректного вычисления делается запись в БД, например:

|  |  |  |
| --- | --- | --- |
| *id* | *temperature* | *day* |
| 1 | 25 | 1 |

**Задание 6:** составить набор входных комбинаций значений (в виде таблицы) необходимый и достаточный для проверки работы приложения.

Отчет по лабораторной работе должен включать данную таблицу.

Формат таблицы:

|  |  |  |
| --- | --- | --- |
| *Температура за 1 день* | *Количество дней* | *Ожидаемый результат вычислений*  *(Значение/Ошибка)* |
|  |  |  |

**Задание 7: о**писать проверки на внешний вид формы и возможные дополнительные проверки.

**По заданию 7 можно подготовить презентацию и выступить**

## ХОД ЛАБОРАТОНОЙ РАБОТЫ

1 Задание

Тестовый набор - это артефакт, в котором отражаются потребности заинтересованных лиц, и позволяющий проверить соответствие продукта этим потребностям.

Создание тестового набора важно по нескольким причинам.

* Тестовый набор можно использовать как основу для проектировании и реализации собственно тестов. Время, потраченное на рассмотрение тестового набора, позволяет лучше понять требования к проекту и реализации и сэкономить время на задачах, относящихся к проектировании и реализации.
* Некоторые тесты могут быть особенно сложными или объемными. Такие тесты лучше хорошо изучить заранее, прежде чем их реализовывать, и для этого тестовый набор и артефакты проекта теста могут быть очень полезными.
* Объем тестирования обычно пропорционален числу тестов. К самому процессу тестирования можно будет относиться с большей уверенностью, если объем тестирования будет обоснован с помощью ряда тестовых наборов.
* Мерой полноты тестирования может является учет какого-либо ряда определяющих факторов. Этими факторами могут быть число выделенных тестовых наборов, число реализованных и /или выполненных тестов для каждого тестового набора, а также усилия, потраченные на исследование каждого тестового набора.
* Масштаб и сложность тестирования в какой-то степени можно считать пропорциональными числу тестовых наборов. Создание тестовых наборов позволяет с большей уверенностью оценить усилия, затраченные на тестирование.
* Варианты проектирования и разработки тестов и требуемые для тестирования ресурсы также зависят от числа и сложности тестовых наборов.

Будет полезным, однако, принять во внимание следующие соображения, касающиеся тестовых наборов:

* Не всякий тест является настолько сложным, чтобы ради него создавать артефакт тестового набора, который подлежит обзору и обновлению. Для простых тестов достаточно краткого описания того, что следует сделать. Очень многие тесты попадают именно в эту категорию. Затраты времени на документирование большого числа простых тестовых наборов могут оказаться излишними.
* Иногда в ходе тестирования обнаруживается, что тесты опирались на ошибочные идеи. Это означает, что тестовые наборы, основанные на этих идеях, будут отброшены. Фактически это будет означать, что вся работа по документированию тестового набора пойдет насмарку, а все отчеты, основанные на нем, потребуется пересмотреть. Поэтому отчеты об охвате тестов лучше основывать на более достоверных источниках, чем тестовые наборы, а тестовые наборы применять внутри коллектива по необходимости.

Тестовые наборы часто создаются на основе типа теста или требования для теста и могут заметно отличаться друг от друга. Эмпирические правила для выделения тестовых наборов сводятся к следующему:

* подтверждение того, что требование удовлетворено (тестовый набор с положительным результатом)
* демонстрация того, что требование может быть удовлетворено **только** в указанных условиях, или отрицательный тест. Этот тестовый набор отражает неприемлемые, нештатные или неожиданные условия или данные, с которыми может встретиться программное обеспечение.

**Создание тестового набора для тестирования производительности**

Основным источником для тестовых наборов, относящихся к производительности, являются вспомогательные спецификации, в которых указаны нефункциональные требования. См. [Рабочий продукт: вспомогательные спецификации](http://dit.isuct.ru/Publish_RUP/core.base_rup/workproducts/rup_supplementary_specification_F5ACAA22.html). При создании тестовых наборов для тестирования производительности придерживайтесь следующих рекомендаций:

* создайте хотя бы один тестовый набор для каждого положения вспомогательной спецификации, относящегося к критериям производительности. Критерии производительности могут выражаться как время на одну транзакцию, число транзакций или пользователей, или процентные соотношения.
* создайте хотя бы один тестовый набор для каждого критически важного варианта использования. Критические варианты использования перечислены в указанных документах и в документе анализа рабочей нагрузки и должны оцениваться с помощью показателей производительности.

Как и для тестовых наборов для функциональных тестов, как правило, одному сценарию использования или требованию будет соответствовать более одного тестового набора. Обычно определяют несколько тестовых наборов, например, один для допороговой производительности (среднее число транзакций), второй - для порога (большое число транзакций) и третий - для надпорогового значения (пиковое число транзакций).

Поимо указанных критериев производительности выделите особые условия, которые влияют на время отклика, а именно:

* Размер базы данных - количество записей
* Типичные варианты нагрузки и транзакций
  + тип и число одновременно работающих пользователей
  + тип, число, частота и продолжительность транзакций
* среда (аппаратное обеспечение, сеть, конфигурация программного обеспечения)

Тестовые наборы для тестов производительности могут также быть выражены в табличной форме, подобно той, что использовалась для функциональных тестов.

#### Создание тестового набора для тестирования защиты и прав доступа

Субъекты и варианты использования описывают взаимодействие между внешними пользователями системы и действиями, которые выполняет система, приводящими к результату, наблюдаемому субъектом. Сложные системы используются многими субъектами, и очень важно разработать тестовые наборы, в которых участвуют только необходимые субъекты. Это особенно необходимо в случаях, когда поток варианта использования зависит от типа субъекта.

Например, поток событий для варианта использования банкомата клиентом банка, выпустившего карту, будет отличаться от потока для клиента стороннего банка или банка, который вообще не включен в данную систему электронных транзакций.

#### Создание тестового набора для тестирования конфигурации

В распределенных системах может поддерживаться много допустимых сочетаний программного обеспечения и аппаратного обеспечения. В ходе тестирования должна быть проверена работа целевого объекта тестирования в различных конфигурациях, в разных операционных системах, браузерах, на разных процессорах. Кроме того, при тестировании требуется проверить работоспособность различных сочетаний компонентов, например, не конфликтуют ли версии DLL, устанавливаемые приложением с версиями этих же DLL, с которыми работает другое приложение.

Для того чтобы создать тестовые наборы для конфигурации, придерживайтесь следующих рекомендаций:

* Создайте хотя бы один тестовый набор для каждой критически важной конфигурации. Для этого определите обязательные программные и аппаратные конфигурации в среде целевого объекта тестирования и определите их приоритеты, чтобы наиболее важные были протестированы в первую очередь:
  + Поддержка принтера
  + Сетевые соединения - локальная и глобальная сеть
  + Конфигурации сервера - драйверы сервера, аппаратное обеспечение сервера
  + Прочее установленное на рабочем столе и/или сервере программное обеспечение
  + Все версии программного обеспечения
* Создайте хотя бы один тестовый набор для каждой конфигурации, в которой вероятны неполадки. В их число входят:
  + Аппаратное обеспечение с наименьшей производительностью.
  + Сопутствующее программное обеспечение, для которого ранее уже были выявлены неполадки, связанные с совместимостью.
  + Клиенты, работающие с сервером по самым медленным сетевым каналам.
  + Ситуации недостатка ресурсов (медленный процессор, недостаточно памяти или места на диске и пр.)

#### Создание тестового набора для тестирования установки

В ходе тестирования установки проверяется, что установка целевого объекта тестирования выполняется правильно во всех возможных сценариях установки. Сценариями могут быть первая установка целевого объекта тестирования или установка его новой версии в систему, где уже есть старая версия. Также необходимо проверить, будет ли программа установки целевого объекта тестирования работать правильно при возникновении неполадок, например, при нехватке места на диске.

Тестовые наборы должны отражать сценарии установки для программного обеспечения, а именно:

* Установка с носителя, например, с дискеты, CD-ROM или файлового сервера.
* Новая установка.
* Полная установка.
* Выборочная установка.
* Обновление системы.

Программы установки для клиент-серверных систем содержат специальные тестовые наборы. В отличие от систем, устанавливаемых на один компьютер, программа установки содержит часть сервера и часть клиента. Поэтому при тестировании необходимо проверить установку всех компонентов целевого объекта тестирования, включая клиент, промежуточные компоненты и сервер.

#### Создание тестового набора для прочих нефункциональных тестов

В идеале все требования для создания тестовых наборов должны быть отражены в модели вариантов использования, проектной модели и вспомогательных спецификациях. Однако часто приходится обращаться еще и к приведенным здесь материалам.

Примеры:

* Тестовые наборы для операционных тестов (проверка длительности работы системы между сбоями).
* Тестовые наборы для обнаружения узких мест в производительности, способности обрабатывать большие объемы данных или для поиска условий, приводящих к сбою системы.

В большинстве случаев эти тестовые наборы создаются на основе или посредством объединения уже имеющихся тестовых наборов.

Создание тестового набора для тестирования приемки продукта

Приемка продукта - это заключительное тестирование перед развертыванием программного обеспечения. Цель приемки состоит в том, чтобы проверить готовность программного обеспечения и пригодность его для окончательной передачи пользователям. Приемка продукта часто включает не только тестирование работы программного обеспечения, но и передачу всех рабочих продуктов заказчикам, включая обучение, документацию и пакеты.

Создание тестовых наборов для рабочих продуктов программного обеспечения фактически уже описано выше. В зависимости от того, насколько формально должна выполняться приемка продукта, тестовые наборы могут быть полным комплектом уже указанных выше тестовых наборов или их частью. В любом случае перед началом тестирования должны быть согласованы тестовые наборы и критерии приемки продукта.

Оценка рабочих продуктов, отличных от программного обеспечения, в значительной степени зависит от того, какой это рабочий продукт. Обратитесь к рекомендациям и справочным таблицам каждого из рабочих продуктов, отличных от программного обеспечения, за информацией о том, каким образом выполняется их оценка.

Создание проверочных тестовых наборов для регрессионного тестирования

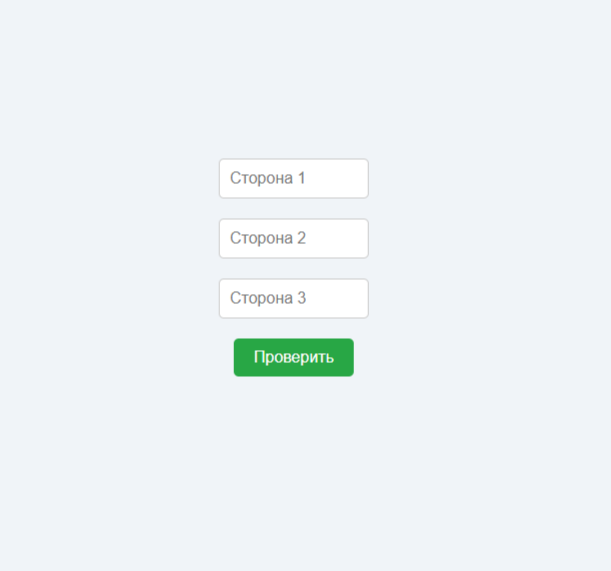
В регрессионном тестировании сравниваются две версии целевого объекта тестирования с целью поиска различий как потенциальных неполадок. При этом предполагается, что более поздняя версия должна работать не хуже прежней, и проверяется, не возникли ли неполадки в результате изменений.

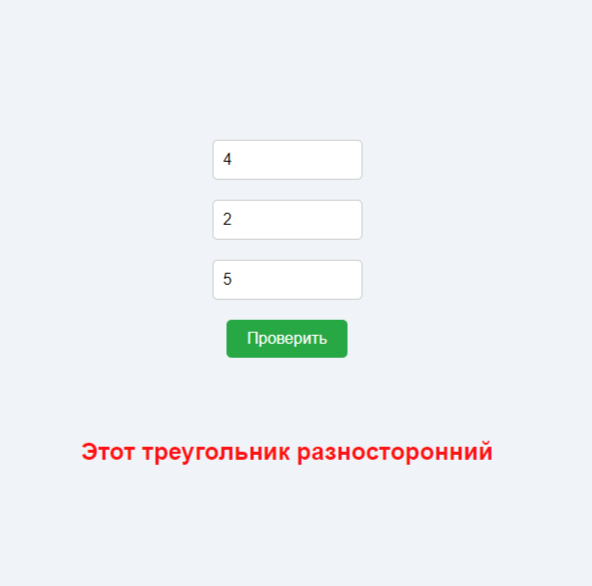
В идеале в последующих итерациях желательно использовать все тестовые наборы с предыдущей итерации. Ниже приведены рекомендации, которые помогают определить и реализовать тестовые наборы, которые повышают ценность регрессионного тестирования, и в то же время снизить расходы на его обеспечение:

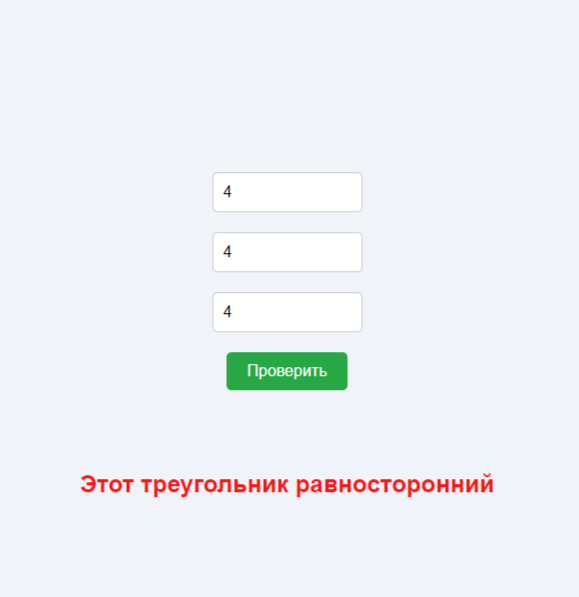
* Определите тестовый набор, в котором проверяются только критически важные элементы данных (требующиеся для воссоздания тестируемого условия)
* Убедитесь, что каждый тестовый набор соответствует уникальному набору входных данных или уникальной последовательности событий, которые должны вызывать уникальное поведение тестируемого продукта.
* Исключите избыточные или равнозначные тестовые наборы
* Выделите группы тестовых наборов, в которых есть одинаковые состояния целевого объекта тестирования и тестовых данных

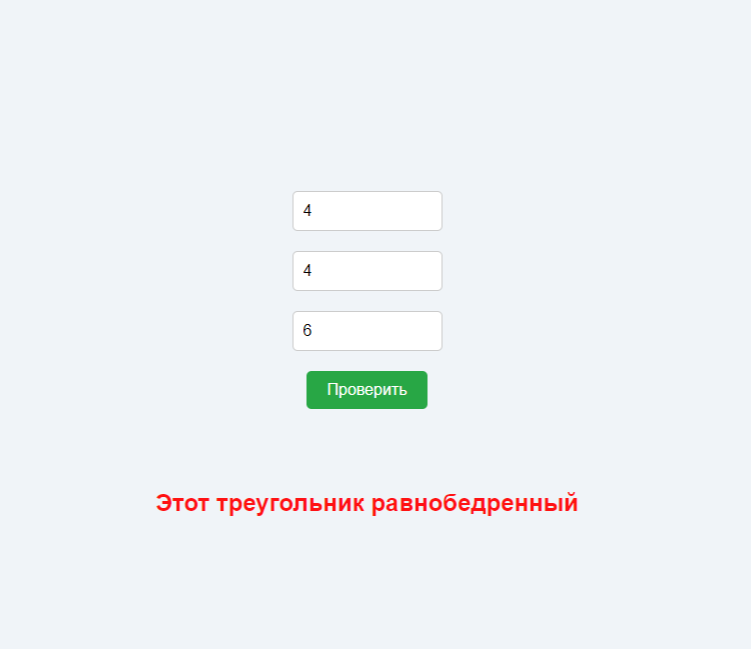
Задание 2

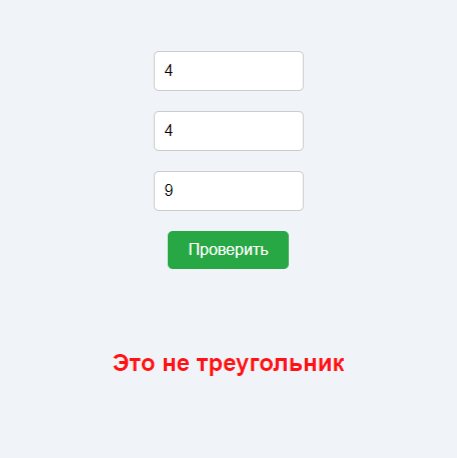
Была написана программа с использование JS в которой определяется вид треугольника изходя из данных о его сторонах.

  
Рисунок 1

  
Рисунок 2

  
Рисунок 3

  
Рисунок 4

  
Рисунок 5

Задание 3

Был составлен набор тестовых данных, которые проверят корректность работы написанной программы.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Сторона 1 | Сторона 2 | Сторона 3 | Условие | Ожидаемый результат |
| 1 | 3 | 3 | 3 | Все стороны равны | Равносторонний |
| 2 | 5 | 5 | 8 | Две стороны равны | Равнобедренный |
| 3 | 7 | 10 | 5 | Все стороны разные | Разносторонний |
| 4 | 1 | 2 | 3 | Сумма двух сторон не больше третьей | Это не треугольник |
| 5 | 6 | 6 | 10 | Две стороны равны | Равнобедренный |
| 6 | 8 | 15 | 20 | Сумма двух сторон не больше третьей | Это не треугольник |
| 7 | 7 | 7 | 7 | Все стороны равны | Равносторонний |
| 8 | 6 | 8 | 10 | Все стороны разные | Разносторонний |
| 9 | 2 | 2 | 4 | Сумма двух сторон не больше третьей | Это не треугольник |
| 10 | 0 | 5 | 3 | Сторона не может быть равна 0 | Это не треугольник |
| 11 | -3 | 4 | 5 | Сторона не может быть меньше 0 | Это не треугольник |
| 12 | 1000 | 1000 | 1000 | Все стороны равны | Равносторонний |
| 13 | 5000 | 5000 | 8000 | Две стороны равны | Равнобедренный |
| 14 | 100000 | 150000 | 200000 | Все стороны разные | Разносторонний |

Таблица 1

Задание 4

Созданный набор тестов покрывает разнообразные случаи и сценарии работы программы, и вот подробная оценка:

* 1. Проверка всех типов треугольников:

Равносторонний треугольник: Включены несколько тестов, что эффективно проверяет корректность определения равностороннего треугольника.

Равнобедренный треугольник: обеспечивают проверку работы программы для равнобедренных треугольников.

Разносторонний треугольник: проверяют корректность определения разностороннего треугольника.

2. Проверка некорректных значений:

Не существующий треугольник: проверяют случаи, когда треугольник не может существовать из-за нарушения неравенства треугольника.

Некорректные значения сторон: Тесты с нулевыми и отрицательными значениями проверяют обработку невалидных данных.

3. Тесты на большие значения:

Большие значения: Тесты с большими числами обеспечивают проверку работы программы с крупными значениями и показывают, что программа корректно обрабатывает большие входные данные.

4. Полнота набора тестов:

Набор покрывает:

Все типы треугольников (равносторонний, равнобедренный, разносторонний).

Сценарии, когда треугольник не существует из-за неравенства треугольника.

Некорректные данные (негативные значения, ноль).

Большие значения.

Вывод:

Набор тестов является достаточно полным и эффективным для проверки основной функциональности программы. Он покрывает все критические сценарии и типы треугольников, а также проверяет корректность обработки невалидных данных и большие значения. Не числовое значение в поле ввести не удастся, т.к. заранее задан тип данных число.

Задание 5

Было протестировано написанное приложение на составленных тестовых данных.

Задание 6

Составил набор входных комбинаций значений в виде таблицы для проверки приложения.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Сторона 1 | Сторона 2 | Сторона 3 | Ожидаемый результат вычислений |
| 1 | 3 | 3 | 3 | Равносторонний |
| 2 | 5 | 5 | 8 | Равнобедренный |
| 3 | 7 | 10 | 5 | Разносторонний |
| 4 | 1 | 2 | 3 | Это не треугольник |
| 5 | 6 | 6 | 10 | Равнобедренный |
| 6 | 8 | 15 | 20 | Это не треугольник |
| 7 | 7 | 7 | 7 | Равносторонний |
| 8 | 6 | 8 | 10 | Разносторонний |
| 9 | 2 | 2 | 4 | Это не треугольник |
| 10 | 0 | 5 | 3 | Это не треугольник |
| 11 | -3 | 4 | 5 | Это не треугольник |
| 12 | 1000 | 1000 | 1000 | Равносторонний |
| 13 | 5000 | 5000 | 8000 | Равнобедренный |
| 14 | 100000 | 150000 | 200000 | Разносторонний |

Таблица 2

Все комбинации данных были проверены и выдали ожидаемый результат, написанная программа работает корректно.

Задание 7

Внешний вид написанной программы представлен на Рисунке 1.Каждому HTML элементу принадлежит свой стиль, они расположены по середине экрана , для более удобного восприятия глазу, кнопка проверки результата имеет зелёный цвет, а полученный результат появляется чуть ниже выделенный красным шрифтом. Тест на адаптивность – при помощи CSS стилей при любом разрешении экрана элементы всегда смещаются в середину.

## ЗАКЛЮЧЕНИЕ

Лабораторная работа выполнена, получены новые навыки составления тестовых данных и их оценки, проверки написанных программ.

## ПРИЛОЖЕНИЕ

Листинг файла index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Треугольник</title>

<script src="script.js" defer></script>

<link rel="stylesheet" href="style.css">

</head>

<body>

<div class="input-container">

<input type="number" id="a" placeholder="Сторона 1">

<input type="number" id="b" placeholder="Сторона 2">

<input type="number" id="c" placeholder="Сторона 3">

<button id="checkButton">Проверить</button>

</div>

<div class="result-container">

<h1 id="result"></h1>

</div>

</body>

</html>

Листинг файла style.css

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: Arial, sans-serif;

background-color: #f0f4f8;

color: #333;

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

height: 100vh;

}

.input-container {

display: flex;

flex-direction: column;

align-items: center;

margin-bottom: 30px;

}

input {

padding: 10px;

margin: 10px;

border-radius: 5px;

border: 1px solid #ccc;

font-size: 16px;

width: 150px;

}

input:focus {

outline: none;

border-color: #007bff;

box-shadow: 0 0 5px rgba(0, 123, 255, 0.5);

}

button {

padding: 10px 20px;

margin-top: 10px;

border: none;

background-color: #28a745;

color: white;

font-size: 16px;

border-radius: 5px;

cursor: pointer;

transition: background-color 0.3s;

}

button:hover {

background-color: #218838;

}

.result-container {

margin-top: 50px;

text-align: center;

}

h1 {

font-size: 24px;

color: #ff0000ec;

}

Листинг файла scrypt.js

const button = document.getElementById('checkButton');

button.addEventListener('click', checkTriangleType);

function checkTriangleType() {

const a = parseInt(document.getElementById('a').value);

const b = parseInt(document.getElementById('b').value);

const c = parseInt(document.getElementById('c').value);

const resultElement = document.getElementById('result');

if (a + b <= c || a + c <= b || b + c <= a) {

resultElement.textContent = 'Это не треугольник';

return;

}

let triangleType;

if (a === b && b === c) {

triangleType = 'Этот треугольник равносторонний';

} else if (a === b || a === c || b === c) {

triangleType = 'Этот треугольник равнобедренный';

} else {

triangleType = 'Этот треугольник разносторонний';

}

resultElement.textContent = triangleType;

}