Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего

образования

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ИРКУТСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

ОТЧЕТ

к лабораторной работе по дисциплине:

|  |
| --- |
| **Технологии разработки программных комплексов** |
| Тестирование |

наименование темы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы | ИСТб-21-1 |  |  |  | Соколов Э.Д. |
|  | шифр группы |  | подпись |  | Фамилия И.О. |
| Проверил |  |  |  |  | Бахвалова З.А. |
|  | должность |  | подпись |  | Фамилия И.О. |

Иркутск 2024 г.

**Содержание**

[1 Постановка задачи 3](#_Toc161907890)

[2 Требования 3](#_Toc161907891)

[2.1 Функциональные требования 3](#_Toc161907892)

[2.2 Нефункциональные требования 3](#_Toc161907893)

[2.3 Требования к пользовательскому интерфейсу 3](#_Toc161907894)

[3 Структурный контроль 4](#_Toc161907895)

[4 Стратегия «белого ящика» 7](#_Toc161907911)

[5 Стратегия «чёрного ящика» 10](#_Toc161907912)

[6 Тестирование пользовательского интерфейса 15](#_Toc161907913)

1 Постановка задачи

Дан равносторонний шестиугольник, внутри которого вписана шестиконечная звезда, вершины которой опираются на стороны шестиугольника. Требуется рассчитать площадь заштрихованной области, изображённой на рисунке 1.1.

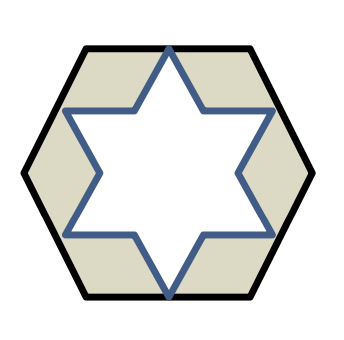


Рисунок 1.1 – Рисунок шестиугольника со вписанной шестиконечной звездой

2 Требования

Требуется создать приложения на языке JavaScript, которое будет рассчитывать площадь заштрихованной области.

2.1 Функциональные требования

Программа должна:

* строить геометрические фигуры и выводить их на экран,
* считать площадь заштрихованной области фигуры,
* изменять размеры фигуры в зависимости от введенных размеров
* масштабировать полотно для отрисовки в зависимости от размеров окна

2.2 Нефункциональные требования

Программа должна быть реализована на платформе языка JavaScript с использованием Bootstrap для создания пользовательского интерфейса и Canvas для отрисовки фигур;

* 1. Требования к пользовательскому интерфейсу
* Для ввода значения размера фигур необходимо поле ввода напротив надписи: "Размер фигур (10-300), пиксели".
* Для запуска отрисовки фигур используется кнопка "Отрисовать".
* Результаты отрисовки выводятся автоматически под кнопкой "Отрисовать".
* Площадь искомой красной зоной выводится под полем для ввода рзмера фигуры
* При изменении размера окна браузера автоматически пересчитывается размер холста и отрисовываются фигуры.
* На данный момент нет необходимости в переключении на другой тип расчета или вводе дополнительных параметров.

3 Структурный контроль

Таблица 1 – Тестирование программы с использованием структурного контроля

|  |  |  |  |
| --- | --- | --- | --- |
| **№ вопроса** | **Строки, подлежащие проверке** | **Результат проверки** | **Вывод** |
| 1.1 | 30-51 файла index.html | Все переменные инициализированы корректно | Все переменные инициализированы |
| 1.2 | 53-71 файла index.html | Все функции отрисовки фигур реализованы правильно | Отрисовка фигур работает корректно |
| 1.3 | 86-97 файла index.html | Проверка корректности ввода данных для размера фигур реализована верно | Доступны только допустимые значения для ввода размера фигур |
| 2.1 | 69-70 файла index.html | Программа выводит результаты расчета площади красной и синей зон с требуемой точностью | Расчеты соответствуют заданным требованиям точности |
| 3.1 | 78-81 файла index.html | Программа завершается только по нажатию на кнопку "Отрисовать" | Программа завершается по требованию пользователя |
| 4.1 | 15-18 файла index.html | Порядок ввода данных соответствует указанным требованиям | Порядок ввода данных отвечает требованиям |
| 4.2 | Нет | Нет неиспользуемых подпрограмм | Все подпрограммы используются корректно |
| 4.3 | Нет | Глобальные и локальные переменные имеют различные имена | Имена переменных уникальны и не пересекаются |

**Вывод:** В целом, структурный контроль – это, полезный метод тестирования программного обеспечения, который может помочь программисту создать качественный код и значительно сократить количество ошибок, но необходимо помнить, что это только один из способов тестирования и он не является абсолютным решением для обнаружения ошибок.

Плюсы структурного контроля как метода тестирования:

* Эффективное обнаружение ошибок: Использование метода структурного контроля может помочь обнаружить ошибки, связанные с некорректным использованием операторов и блоков кода.
* Повышение качества кода: Результатом использования метода структурного контроля может быть улучшение качества кода, так как этот подход помогает идентифицировать некорректно написанные операторы, циклы и блоки.
* Повышение понимания кода: Использование структурного контроля может помочь в понимании логики программного кода и улучшить его читаемость.

Минусы структурного контроля как метода тестирования:

* Ограниченная эффективность: Использование метода структурного контроля не гарантирует абсолютную надежность кода и может не помочь выявить определенные типы ошибок.
* Неэффективное использование времени: Использование метода структурного контроля может потребовать большого количества времени и ресурсов, которые могут быть необходимы для более важных задач.

4 Стратегия «белого ящика»

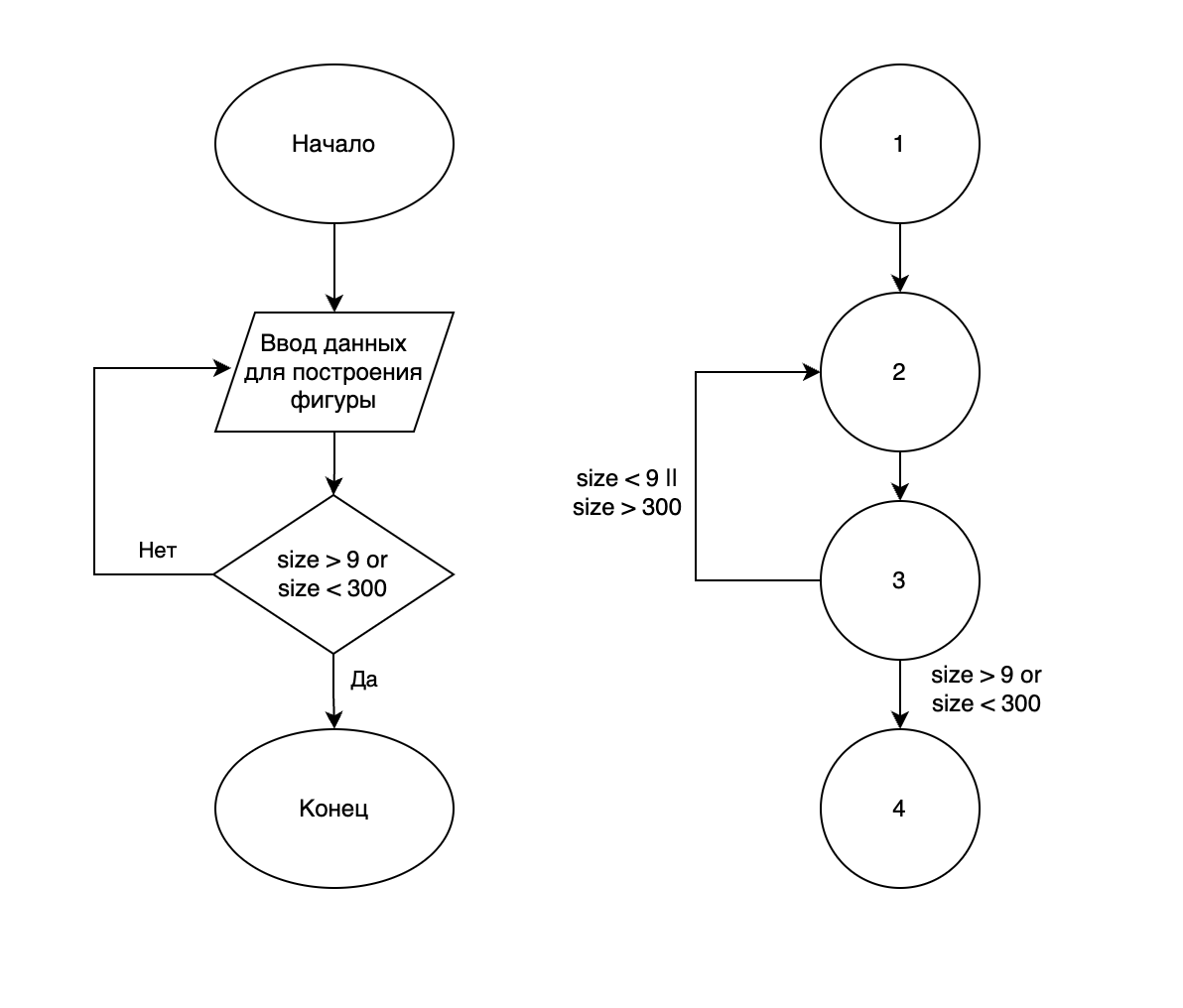


Рисунок 4.1 – Блок-схема решений при тестировании и её граф передачи управления

Таблица 4.1 – Тесты программы с использованием стратегии белого ящика

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Покрытие** | **Назначение теста** | **№ теста** | **Значение исходных данных** | **Ожидаемый результат** |
| Покрытие операторов | Проверка наличия операторов при вводе корректных числовых значений | 1 | Size = 100 | Путь 1 2 3 4 |
| Проверка наличия операторов при вводе некорректных значений | 2 | Size = 0 | Путь 1 2 3 (3) |
| Покрытие решений | Проверка принадлежности введенных данных промежутку от 10 до 300 | 3 | Size = 100 | Путь 1 2 3 4 |
| Проверка не принадлежности введенных данных промежутку от 10 до 300 | 4 | Size = 0 | Путь 1 2 3 (3) |
| Покрытие условий | Проверка пути, при принадлежности введенных данных промежутку от 10 до 300 | 5 | Size = 100 | Путь 1 2 3 4 |
| Проверка пути, при не принадлежности введенных данных промежутку от 10 до 300 | 6 | Size = 0 | Путь 1 2 3 (3) |
| Покрытие решений/условий | Проверка корректности выполнении решения | 7 | Size = 100 | Путь 1 2 3 4 |
| Проверка корректности выполнения условия | 8 | Size = 0 | Путь 1 2 3 (3) |
| Комбинаторное покрытие | Проверка принадлежности введенных данных промежутку от 10 до 300 и корректного выполнения решения | 9 | Size =100 | Пусть 1 2 3 4 5 |

Таблица 4.2 – Тестирование с использованием стратегии белого ящика

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№ теста** | **Входные данные** | **Ожидаемый результат** | **Результат** | **Вывод по покрытию** |
| 1 | Size = 0 | Путь 1 2 3 (3) | Путь 1 2 3 (3) | Операторы условий работают |
| 2 | Size = 100 | Пусть 1 2 3 4 | Пусть 1 2 3 4 |

**Вывод:** Подводя итоги тестирования программы стратегией белого ящика, в данной случае все методы покрывают все возможные варианты условий, связано это с небольшим количеством и не сложной структурой условий. Данное тестирование позволяет проверить логику и внутреннюю структуру программы. Однако даже у такой стратегии есть недостатки, в том, что даже в программе такого уровня, количество тестов сравнительно большое. Также эта стратегия не дает проверку соответствует программа полным требованиям.

5 Стратегия «чёрного ящика»

Таблица 5.1 – Классы эквивалентности

|  |  |  |
| --- | --- | --- |
| **Входные условия** | **Правильные классы эквивалентности (№)** | **Неправильные классы эквивалентности (№)** |
| Размер фигур (size) | 10 <= size (1)  300 > size (2)  Вещественное значение (3)  Разделителем может быть «.» или «,»(4) | Строка символов (5), Пустое значение (6), Меньше 10 (7)  Больше 300 (8) |

Таблица 5.2 – Тест программы с использованием стратегии чёрного ящика

|  |  |  |  |
| --- | --- | --- | --- |
| **№ теста** | **Назначение теста** | **Значение исходных данных** | **Ожидаемый результат** |
| 1 | Покрытие правильного класса эквивалентности №1, №2, №3 и №4 | size = 50 | Отрисовка фигур |
| 2 | Покрытие неправильного класса эквивалентности №5 | size = "abc" | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 3 | Покрытие неправильного класса эквивалентности №6 | size = "" | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 4 | Покрытие неправильного класса эквивалентности №7 | size = -20 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 5 | Покрытие неправильного класса эквивалентности №8 | size = 345 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |

Таблица 5.2.1 - Тестирование программы с использованием стратегии чёрного ящика

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№ теста** | **Значение исходных данных** | **Ожидаемый результат** | **Результат** | **Вывод** |
| 1 | size = 5 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." | Предупреждение | Неправильный класс эквивалентности № 1 покрыт |
| 2 | Size = 345 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." | Предупреждение | Неправильный класс эквивалентности № 2 покрыт |
| 3 | size = 50 | Отрисовка фигур | Отрисовка фигур | Правильный класс эквивалентности № 1 покрыт |
| 4 | size = "abc" | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." | Предупреждение | Неправильный класс эквивалентности № 1 покрыт |
| 5 | size = "" | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." | Предупреждение | Неправильный класс эквивалентности № 2 покрыт |
| 6 | size = -20 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." | Предупреждение | Неправильный класс эквивалентности № 3 покрыт |

Таблица 5.3 – Тест граничных условий

|  |  |  |  |
| --- | --- | --- | --- |
| **№ теста** | **Назначение теста** | **Значение исходных данных** | **Ожидаемый результат** |
| 1 | Проверить левую границу условия ввода размера фигур | size = 9 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 2 | size = 10 | Отрисовка фигур |
| 3 | Проверить правую границу условия ввода размера фигур | size = 301 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 4 | size = 300 | Отрисовка фигур |

Таблица 5.3.1 - Тестирование граничных условий

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Значение исходных данных | Ожидаемый результат | Результат | Вывод |
| 1 | size = 9 | Предупреждение | Предупреждение | Левая граница условия ввода размера фигур четко определена |
| 2 | size = 10 | Отрисовка фигур | Отрисовка фигур |
| 1 | size = 301 | Предупреждение | Предупреждение | Правая граница условия ввода размера фигур четко определена |
| 2 | size = 300 | Отрисовка фигур | Отрисовка фигур |

Таблица 5.4 – Тесты ожидаемых ошибок

|  |  |  |  |
| --- | --- | --- | --- |
| **№ теста** | **Назначение** | **Исходные данные** | **Ожидаемый результат** |
| 7 | Тестирование ввода нечислового значения | size = "пятьё" | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 8 | Тестирование ввода отрицательного значения | size = -21 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 9 | Тестирование ввода пустого значения | size = " " | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 10 | Тестирование ввода значения больше 300 | Size > 300 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 11 | Тестирование ввода значения меньше 10 | size < 10 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |

Таблица 5.5 – Тестирование ожидаемых ошибок

|  |  |  |  |
| --- | --- | --- | --- |
| **№ теста** | **Назначение** | **Исходные данные** | **Полученный результат** |
| 7 | Тестирование ввода нечислового значения | size = "пятьё" | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 8 | Тестирование ввода отрицательного значения | size = -21 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 9 | Тестирование ввода пустого значения | size = " " | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 10 | Тестирование ввода значения больше максимального (больше 300) | size = 324 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |
| 11 | Тестирование ввода значения меньше минимально допустимого (меньше 10) | size = 9 | Предупреждение: "Пожалуйста, введите корректное положительное число для размера фигур." |

**Вывод:** Подводя итоги тестирования программы стратегией черного ящика, можно отметить возможность получения понимания поведения программы при использовании ее пользователем, а также узнать скрытые несоответствия с требованиями. Данная стратегия позволяет сразу понять соответствует ли она функциональным требованиям. Недостатками является возможность пропуска границ и переходов, которые не очевидны из спецификации.

6 Тестирование пользовательского интерфейса

Таблица 6.1 – Тестирование пользовательского интерфейса программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№ теста** | **Действие** | **Компонент** | **Реакция системы** | **Результат** |
| 1 | Ввод в поле ввода значения для размера фигур (макс. 300) | Поле ввода | Программа приняла заданное значение | Программа использовала заданные данные для отрисовки фигур |
| 2 | Нажатие на кнопку для отрисовки фигуры | Кнопка | Программа выполнила действие при нажатии кнопки | В поле для отрисовки отобразились фигуры заданного размера |
| 3 | Нажатие на кнопку для показа задания | Кнопка | Программа выполнила действие при нажатии кнопки | Появилось окно с описанием задания |
| 4 | Изменение размеров окна приложения | Окно | Размер окна был изменён под параметр пользователя | Изменение размера окна приложения |

Таблица 6.2 – Тесты функциональности пользовательского интерфейса программы

|  |  |  |  |
| --- | --- | --- | --- |
| **№**  **теста** | **Функция** | **Действие** | **Реакция системы** |
| 1 | Построение фигуры | Введите значение в поле ввода длина стороны шестиугольника = 100 | В поле для отображения фигуры, появятся шестиугольник с длиной стороны 100, с вписанной шестиконечной звездой |
| 2 | Расчёт площади построенной фигуры | Введите значение в поле ввода длина стороны шестиугольника = 100 | В поле отображения площади, изменяется значение площади на 4598.75 |
| 3 | Отображение информации о задании | Нажатие на кнопку теория | В поле для отображения фигуры, появится справочная информация по предметной области |

Таблица 6.2.1 – Тестирование функциональности пользовательского интерфейса программы

|  |  |  |  |
| --- | --- | --- | --- |
| **№**  **теста** | Входные данные | Ожидаемый результат | **Результат** |
| 1 | Введите значение в поле ввода длина стороны шестиугольника = 100 | Верно | Верно |
| 2 | Введите значение в поле ввода длина стороны шестиугольника = 100 | Верно | Верно |
| 3 | Нажатие на кнопку теория | Верно | Верно |

**7 Тесты ошибок**

|  |  |  |
| --- | --- | --- |
| **№**  **теста** | Входные данные | Результат |
| 1 | Изменение размеров окна до минимального при максимальном размере фигуры | Фигура не помещается в окно |

**8 Листинг кода**

**<!DOCTYPE html>**

**<html lang="ru">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>Shapes App</title>**

**<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">**

**<style>**

**#canvas-container {**

**height: calc(100vh - 50px); /\* Высота холста равна высоте экрана минус высота верхней панели \*/**

**}**

**canvas {**

**border: 1px solid black;**

**width: 100%;**

**height: 100%;**

**}**

**</style>**

**</head>**

**<body class="container-fluid">**

**<div class="row bg-light py-2">**

**<div class="col-md-4">**

**<label for="size" class="form-label">Размер фигур (макс. 300):</label>**

**<input type="number" id="size" class="form-control" min="10" value="100">**

**</div>**

**<div class="col-md-4">**

**<button onclick="drawShapes()" class="btn btn-primary mt-2">Отрисовать</button>**

**</div>**

**<div class="col-md-3">**

**<button onclick="showTheory()" class="btn btn-secondary mt-2">Теория</button>**

**</div>**

**<div class="col-md-4">**

**<p id="hexagonArea" class="mt-3"></p>**

**<p id="starArea"></p>**

**</div>**

**</div>**

**<div id="canvas-container" class="mt-3">**

**<canvas id="canvas"></canvas>**

**</div>**

**<script>**

**const canvas = document.getElementById('canvas');**

**const ctx = canvas.getContext('2d');**

**const sizeInput = document.getElementById('size');**

**function drawShapes() {**

**let size = parseInt(sizeInput.value);**

**// Проверка на отрицательные числа и некорректные символы**

**if (isNaN(size) || size <= 9 || size > 300) {**

**alert("Пожалуйста, введите корректное положительное число для размера фигур.");**

**return; // Прекратить выполнение функции**

**}**

**const hexagonSide = size / 2 / Math.cos(Math.PI / 6);**

**const centerX = canvas.width / 2;**

**const centerY = canvas.height / 2;**

**ctx.clearRect(0, 0, canvas.width, canvas.height);**

**// Рисуем шестиугольник**

**ctx.beginPath();**

**ctx.moveTo(centerX + hexagonSide, centerY);**

**for (let i = 1; i <= 6; i++) {**

**const angle = i \* Math.PI / 3;**

**ctx.lineTo(centerX + hexagonSide \* Math.cos(angle), centerY + hexagonSide \* Math.sin(angle));**

**}**

**ctx.closePath();**

**ctx.fillStyle = 'red';**

**ctx.fill();**

**ctx.strokeStyle = 'black';**

**ctx.stroke();**

**// Рисуем звезду**

**ctx.beginPath();**

**ctx.moveTo(centerX, centerY - size / 2);**

**for (let i = 1; i <= 12; i++) {**

**const angle = i \* Math.PI / 6 - 3 \* Math.PI / 6;**

**const radius = i % 2 === 0 ? size / 2 : size / (2 \* 1.732);**

**ctx.lineTo(centerX + radius \* Math.cos(angle), centerY + radius \* Math.sin(angle));**

**}**

**ctx.closePath();**

**ctx.fillStyle = 'blue';**

**ctx.fill();**

**ctx.strokeStyle = 'black';**

**ctx.stroke();**

**// Вычисляем и выводим площади фигур**

**const starArea = (5 / 4 \* size \*\* 2 \* Math.tan(Math.PI / 10)).toFixed(2);**

**const hexagonArea = (3 \* Math.sqrt(3) / 2 \* hexagonSide \*\* 2).toFixed(2) - starArea;**

**document.getElementById('hexagonArea').textContent = `Площадь красной зоны: ${hexagonArea}`;**

**}**

**function showTheory() {**

**alert("Программа для построения шестиугольника и звезды, вычисления их площади и отображения результата на холсте. Введите размер фигуры, нажмите 'Отрисовать' для построения. Максимальный размер - 300.");**

**}**

**// Установим начальный размер холста**

**resizeCanvas();**

**// Обработка изменений размера окна**

**window.addEventListener('resize', resizeCanvas);**

**// Функция изменения размера холста**

**function resizeCanvas() {**

**canvas.width = document.getElementById('canvas-container').offsetWidth;**

**canvas.height = document.getElementById('canvas-container').offsetHeight;**

**drawShapes();**

**}**

**</script>**

**</body>**

**</html>**