**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение**

**Высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Математической кибернетики и информационных технологий

**Отчет по лабораторной работе №5**

По дисциплине «СиАОД»

На тему:

«Рекурсия. Фракталы»

Выполнил: студент группы БВТ1801

Задоркин Максим Александрович

Руководитель:

Кутейников Иван Алексеевич

Москва 2020

**Цель работы:**   
Изучить основные фракталы и способы построение их через рекурсию.

**Задача (Вариант 10):**  
Реализовать генерацию заданного типа фрактала с применением рекурсивных функций. Добавить возможность задания глубины фрактала. Оценить глубину рекурсии. Построить таблицу зависимости времени построения от глубины фрактала.

Тип фрактала: Салфетка Серпинского

**Программа:**

Программа реализована при помощи языка javascript, domAPI и технологии svg. Так как использованы последние стандарты ecmascript рекомендуется запускать приложение в последних версиях браузера.

Все файлы кода можно просмотреть и скачать с моего github: <https://github.com/MaximZar/SaDPA/tree/master/fractalNapkin>

Опробовать приложение можно прямо по следующей ссылке: <https://maximzar.github.io/SaDPA/fractalNapkin/src/index.html>

1. **Код программы**

Код содержит несколько абстракций:

makePoint – создание точки координат

makeTrianglePoints – создание объекта, содержащего 3 точки равностороннего треугольника

makeTriangle – создание svg треугольника

dive – рекурсивная функция. Используется итеративный процесс, результат функции исчисляется сразу, функция ничего не возвращает.

drawFractal – главная функция, где содержится основа фрактала и запуск dive рекурсии. Данная функция привязана к событию нажатия клавиши «Нарисовать»

Глубина рекурсии равна глубине фрактала. Этого можно наблюдать в функции dive.

**const** color = '#ffcc00' *//главный цвет*

**const** triangleColor = '#222'; *//цвет вырезанных треугольников*

**const** makePoint = (x, y) **=>** ({x, y});

**const** makeTrianglePoints = (startPoint, height, length) **=>** ({

pointLeft: startPoint,

pointRight: { x: startPoint.x + length, y: startPoint.y },

pointBottom: { x: startPoint.x + (Math.round(length / 2)), y: startPoint.y + height }

});

**const** makeTriangle = (startPoint, height, length, color) **=>** {

**const** triangle = document.createElement('polygon');

**const** points = makeTrianglePoints(startPoint, height, length);

**const** point1 = Object.values(points.pointLeft).join(' ');

**const** point2 = Object.values(points.pointRight).join(' ');

**const** point3 = Object.values(points.pointBottom).join(' ');

triangle.setAttribute('points', `${point1} ${point2} ${point3}`);

triangle.setAttribute('fill', color);

**return** triangle;

};

**const** dive = (fractal, svg, points, depth, maxDepth, length, color) **=>** {

**if** (depth > maxDepth) **return**;

*//строим все 3угольники на этом уровне*

**const** newPoints = points.flatMap(point **=>** {

*// point: создание треугольника*

*// \**

*// o \**

*// \*\*\*\*\**

**const** height = length - length \* (1 - Math.sqrt(3) / 2);

**const** triangle = makeTriangle(point, height, length, color);

svg.append(triangle);

*// пересоздаем новые точки на месте этой старой для будущей итерации*

*// потом здесь будет 3 треугольника*

*/// \**

*/// 0 \**

*/// \* \**

*/// 1 2 \**

*/// \*\*\*\*\*\*\**

**const** { x, y } = point;

**const** leftBottomPoint = makePoint(x - length / 4, y + height / 2); *// 0*

**const** leftTopPoint = makePoint(x + length / 4, y - height / 2); *// 1*

**const** rightBottomPoint = makePoint(leftBottomPoint.x + length, leftBottomPoint.y); *// 2*

**return** [leftBottomPoint, leftTopPoint, rightBottomPoint];

});

dive(fractal, svg, newPoints, depth + 1, maxDepth, length / 2, color);

};

**const** drawFractal = () **=>** {

*// первая ступень: создаем основу для фрактала*

**const** length = 600; *//длина стороны треугольника*

**const** depth = 1; *// текущая глубина*

**const** maxDepth = document.querySelector('#depth-fractal').value; *// глубина, на которую нужно погрузиться*

**const** fractal = document.querySelector('#fractal');

**const** svg = document.createElement('svg');

*// Создание главного треугольника*

**const** mTriangle = document.createElement('polygon');

**const** height = length - Math.sqrt(3) \* length / 2;

**const** leftPoint = makePoint(0, length);

**const** rightPoint = makePoint(length, length);

**const** topPoint = makePoint(length / 2, height);

**const** trianglePoints = [...Object.values(leftPoint),

...Object.values(topPoint), ...Object.values(rightPoint)];

mTriangle.setAttribute('points', `${trianglePoints.join(' ')}`);

mTriangle.setAttribute('fill', color);

svg.append(mTriangle);

*//строим фрактал*

**const** points = [makePoint(length / 4, (length + height) / 2)]; *// пояснение в dive к этой точке*

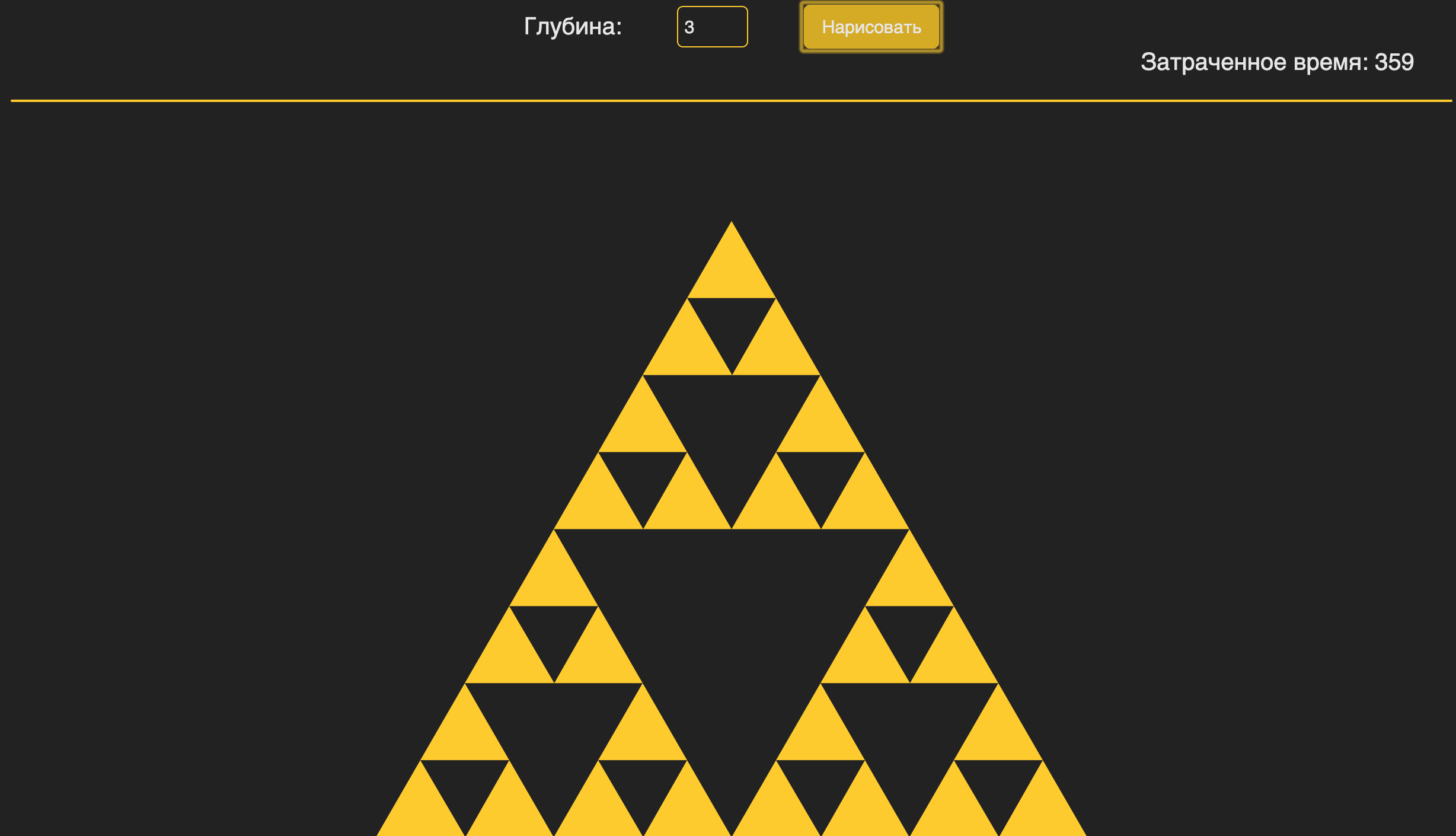
dive(fractal, svg, points, depth, maxDepth, length / 2, triangleColor);

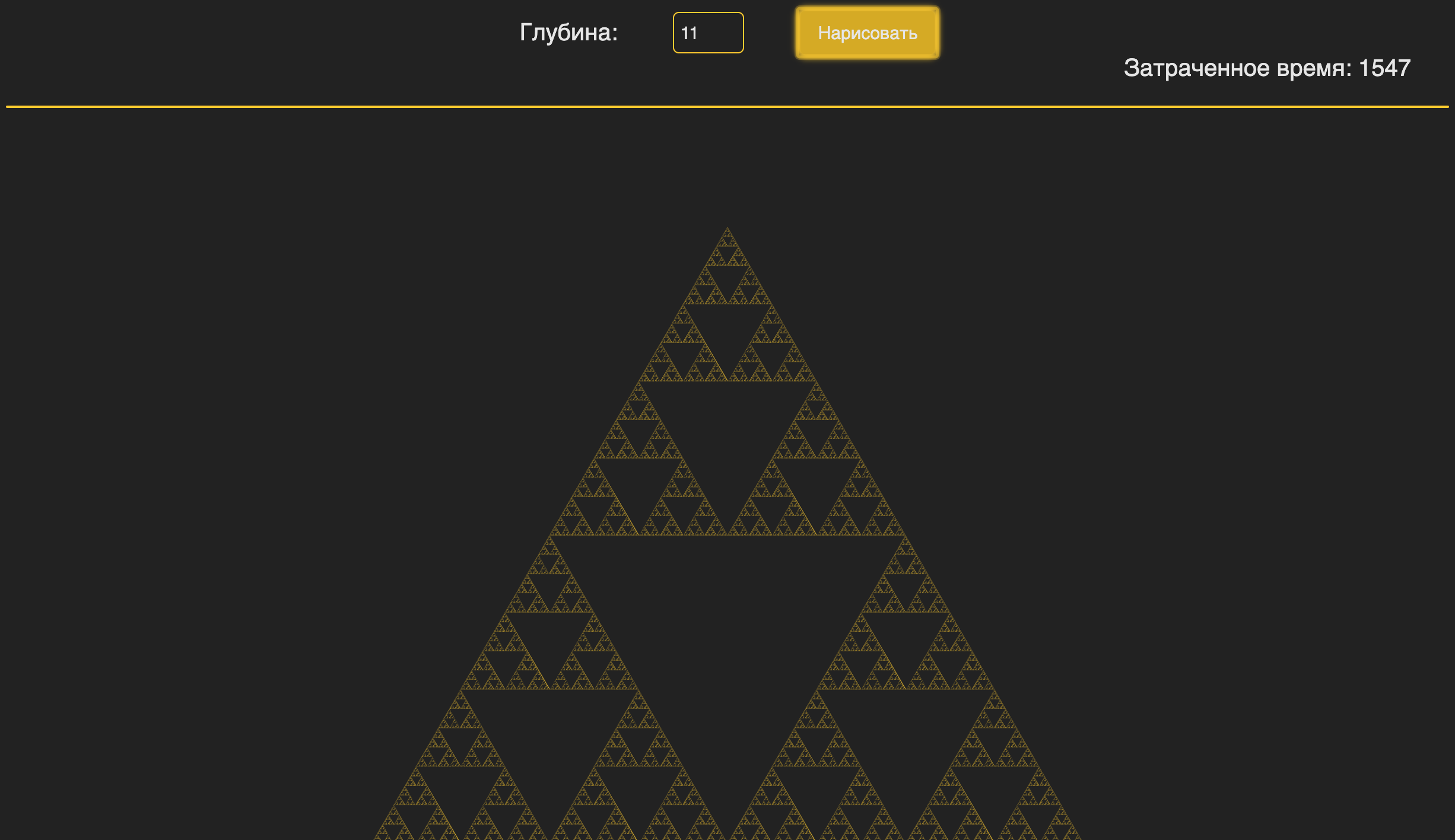
fractal.innerHTML = svg.innerHTML;

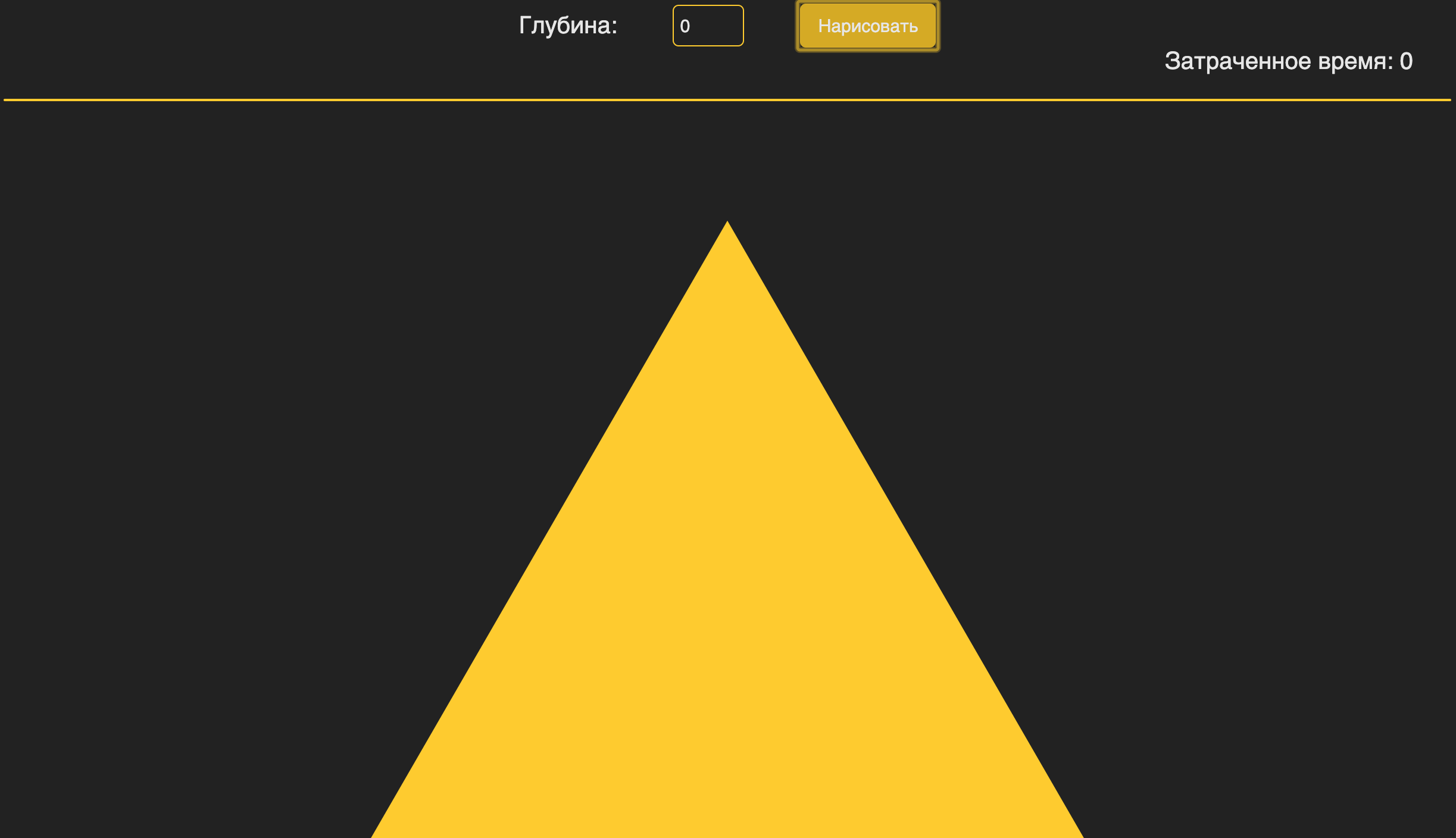
};

document.querySelector('#start-btn').addEventListener('click', drawFractal);

**Тестирование** **программы и скриншоты работы:**







Итоговая таблица тестирований:

|  |  |  |
| --- | --- | --- |
| **Глубина фрактала** | **Глубина рекурсии** | **Время создания** |
| 1 | 1 | 1 |
| 4 | 4 | 2 |
| 8 | 8 | 60 |
| 10 | 10 | 819 |
| 12 | 12 | 5206 |
| 13 | 13 | 19497 |

**Вывод:** из вышеперечисленных данных можно сделать вывод, что при увеличении глубины в разы увеличивается время, требуемое на создание фрактала.