Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Курский государственный университет»

Кафедра программного

обеспечения и администрирования

информационных систем

Направление подготовки

математическое обеспечение

и администрирование

информационных систем

Форма обучения очная

**Отчёт**

**по лабораторной работе №2**

дисциплина «Компьютерная графика»

Выполнил:

студент группы 313.1 Андреенко А.Д.

Проверил:

доц. кафедры ПОиАИС Сухотерин Е. А.

Курск, 2021

**Цель работы**: Изучение математических методов аффинных преобразований на плоскости и практическое освоение приёмов программной реализации аффинных преобразований.

**Задание**

На самостоятельно выбранном языке программирования создать программу, реализующую следующие преобразования фигуры (многоугольника) на плоскости в зависимости от варианта:

1. Перенос:
   1. вдоль оси OX (ввод расстояния переноса);
   2. вдоль оси OY (ввод расстояния переноса);
   3. вдоль вектора OP (указание точки P);
   4. вдоль заданной стороны фигуры (указание стороны, расстояние переноса равно длине стороны).
2. Масштабирование:
   1. вдоль оси OX (ввод масштабного коэффициента);
   2. вдоль оси OY (ввод масштабного коэффициента);
   3. вдоль вектора OP (указание точки P, ввод масштабного коэффициента);
   4. вдоль заданной стороны фигуры (указание стороны, ввод масштабного коэффициента).
3. Отражение:
   1. относительно оси OX;
   2. относительно заданной вершины фигуры (указание вершины);
   3. относительно прямой с направляющим вектором OP, проходящей через начало координат (указание точки P);
   4. относительно заданной стороны фигуры (указание стороны).
4. Поворот:
   1. вокруг начала координат (ввод величины угла поворота);
   2. вокруг заданной точки плоскости (указание точки, ввод величины угла поворота);
   3. вокруг заданной вершины фигуры (указание вершины, ввод величины угла поворота).

Требования к программе:

а) включать два окна (на одной форме или на разных), содержащих изображение фигуры до и после преобразования и управляющие элементы для ввода необходимой информации;

б) отображать координаты вершин исходной и преобразованной фигуры;

в) отображать итоговую матрицу преобразования и обеспечивать возможность внесения ручных изменений в эту матрицу и применения полученной матрицы к исходной фигуре;

г) иметь возможность отображать или скрывать координатные оси и масштабную сетку в окнах вывода фигур;

д) обеспечивать ввод фигуры для преобразования при помощи мыши посредством указания вершин фигуры;

е) для каждого преобразования обеспечивать возможность динамической визуализации многократного последовательного применения заданного преобразования к исходному многоугольнику (до остановки пользователем).

Работа выполняется по индивидуальному варианту №1, который представлен на рисунке 1.



Рисунок 1 – Вариант 1

**Разработка алгоритма**

Аффинное преобразование на плоскости — это линейное невырожденное преобразование плоскости в себя.

Афинное преобразование может быть представлено в матричном виде:

Здесь:

P = (x, y) – координаты исходной точки;

P` = (x`, y`) – координаты точки преобразования;

– матрицы преобразования, где .

Для упрощения реализации были введены однородные координаты.

Однородные координаты точки плоскости P с декартовыми координатами (x, y) — это тройка вида C = (a, b, c) = (x·h, y·h, h), где h — произвольное число, отличное от нуля. В данном случае, h был принят равным единице, то есть С = (x, y, 1).

В однородных координатах аффинные преобразования имеют одинаковую форму произведения вектора исходных координат на матрицу преобразования - C` = C · A.

Матрица C составного преобразования, заключающегося в последовательном выполнении n элементарных преобразований c матрицами определяется как произведение этих матриц слева направо в порядке выполнения:

В ходе выполнения данной работы было необходимо реализовать следующие афинные преобразования:

1. Перенос вдоль вектора OP;
2. Масштабирование вдоль оси OX;
3. Отражение относительно заданной вершины фигуры;
4. Поворот вокруг заданной точки плоскости;
5. Перенос вдоль вектора.

Параметром преобразования является вектор = , на который совершается перенос. Матрица преобразования T имеет вид:

.

1. Масштабирование вдоль координатных осей с заданными коэффициентами.

Преобразование имеет два параметра: коэффициент масштабирования вдоль оси OX и коэффициент масштабирования вдоль оси OY. Матрица преобразования имеет вид:

.

1. Отражение относительно заданной вершины фигуры.

Параметром преобразования является вершина фигуры . Данное преобразование можно представить в виде последовательности трех элементарных преобразований:

1. Перенос каждой точки фигуры на вектор -. Таким образом вершина окажется в начале координат;
2. Масштабирование с матрицей , то есть отражение относительно начала координат;
3. Перенос каждой точки фигуры на вектор .
4. Поворот вокруг заданной точки плоскости.

Параметром преобразования являются точка плоскости и угол поворота φ. Данное преобразование можно представить в виде последовательности трех элементарных преобразований:

1. Перенос каждой точки фигуры на вектор -. Таким образом фигура окажется в том же положении относительно начала координат, в котором она была относительно точки до переноса;
2. Поворот каждой точки фигуры на угол φ относительно начала координат с матрицей преобразования R(φ):
3. Перенос каждой точки фигуры на вектор .

Поскольку в однородных координатах все афинные преобразования точки выполняются единообразно, алгоритм выполнения произвольного афинного преобразования заданного многоугольника для различных преобразований отличается только в части формирования матрицы преобразования. Блок-схема данного алгоритма представлена на рисунке 2.

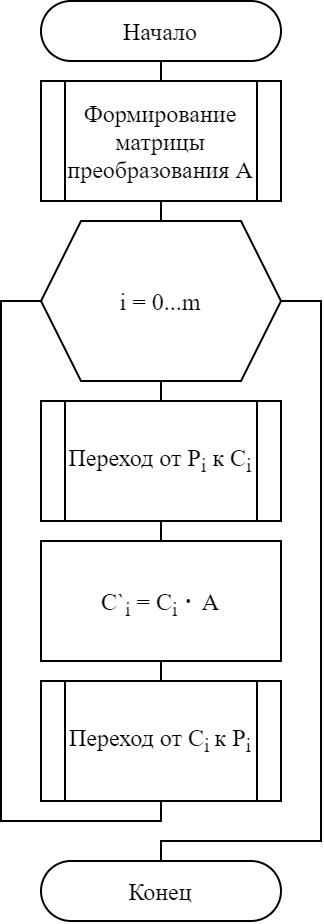


Рисунок 2 – Блок-схема алгоритма реализации произвольного афинного преобразования m-угольника. Здесь: , – декартовы и однородные координаты i-й вершины исходного многоугольника; – однородные и декартовы координаты образа исходного многоугольника; m – количество вершин многоугольника.

В качестве средств программной реализации для данной лабораторной работы был выбран язык программирования JavaScript и графическая библиотека p5.js. Выбор данных средств реализации обоснован наличием опыта работы с ними и простотой их использования.

**Текст программы**

Программная реализация данных афинных преобразований приведена ниже:

translate() {

let x = this.translation.x;

let y = this.translation.y;

const translationMatrix = [

[1, 0, 0],

[0, 1, 0],

[x, y, 1]

];

this.resultMatrix = math.multiply(this.resultMatrix, translationMatrix);

this.updateTransformed();

this.updateTable();

}

scale() {

let x = this.scaling.x;

let y = this.scaling.y;

const scalingMatrix = [

[x, 0, 0],

[0, y, 0],

[0, 0, 1]

];

this.resultMatrix = math.multiply(this.resultMatrix, scalingMatrix);

this.updateTransformed();

this.updateTable();

}

reflect() {

// saving previous values to later restore them

const [scaleX, scaleY] = [this.scaling.x, this.scaling.y];

const [translateX, translateY] = [this.translation.x, this.translation.y];

this.translation.x = -this.reflection.x;

this.translation.y = -this.reflection.y;

this.translate();

this.scaling.x = -1;

this.scaling.y = -1;

this.scale();

this.translation.x = this.reflection.x;

this.translation.y = this.reflection.y;

this.translate();

[this.scaling.x, this.scaling.y] = [scaleX, scaleY];

[this.translation.x, this.translation.y] = [translateX, translateY];

}

rotate() {

const [translateX, translateY] = [this.translation.x, this.translation.y];

const angle = -transformed.radians(this.rotation.angle);

const rotationMatrix = [

[Math.cos(angle), -Math.sin(angle), 0],

[Math.sin(angle), Math.cos(angle), 0],

[0, 0, 1]

];

this.translation.x = -this.rotation.x;

this.translation.y = -this.rotation.y;

this.translate();

this.resultMatrix = math.multiply(this.resultMatrix, rotationMatrix);

this.translation.x = this.rotation.x;

this.translation.y = this.rotation.y;

this.translate();

[this.translation.x, this.translation.y] = [translateX, translateY];

}

**Тестирование программы**

Тестирование программы представлено на рисунках 3 –

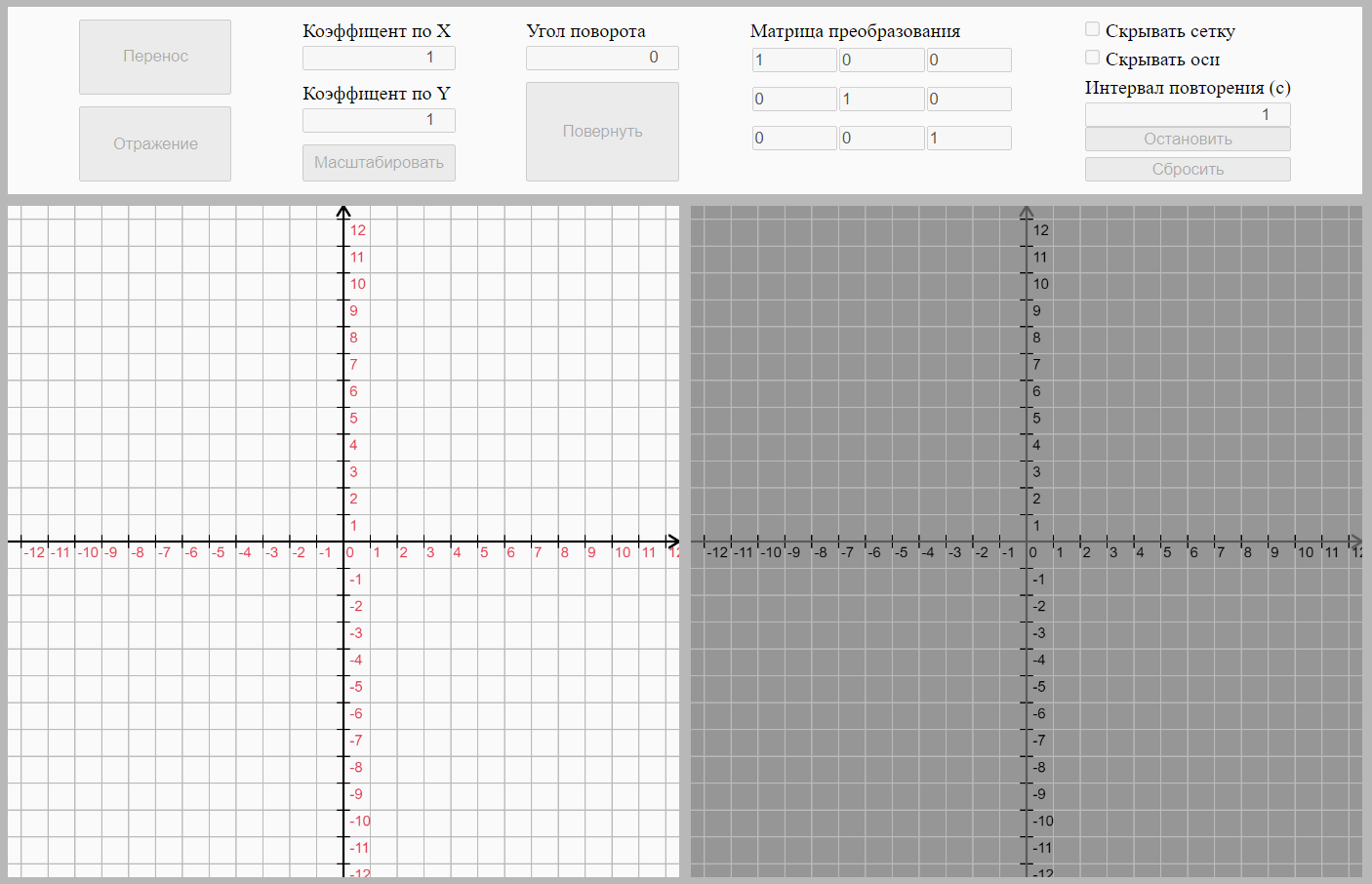


Рисунок 3 – Начальный вид формы

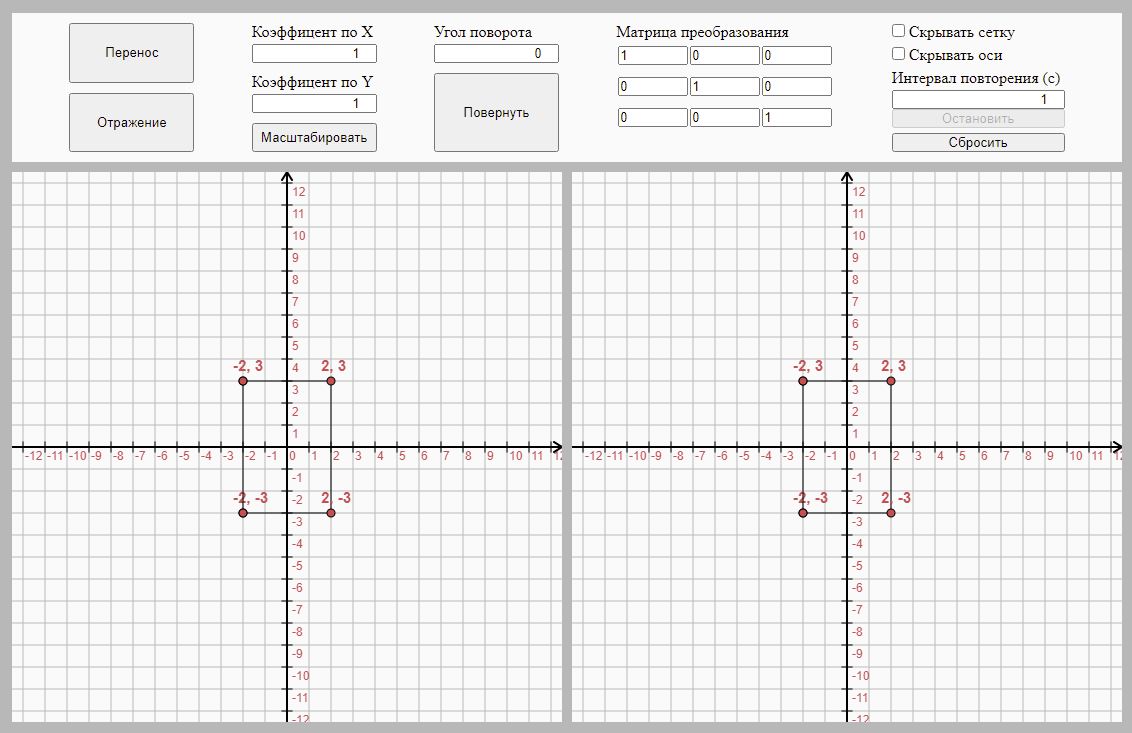


Рисунок 4 – Результат работы приложения после ввода фигуры

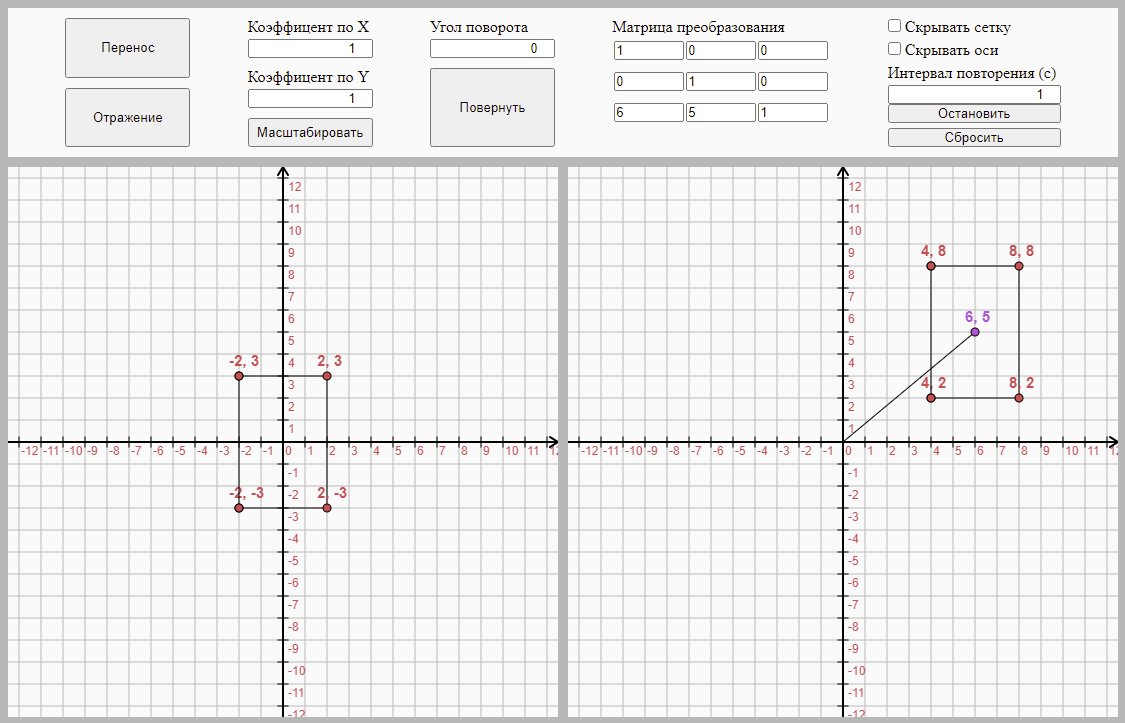


Рисунок 5 - Результат работы приложения после применения переноса

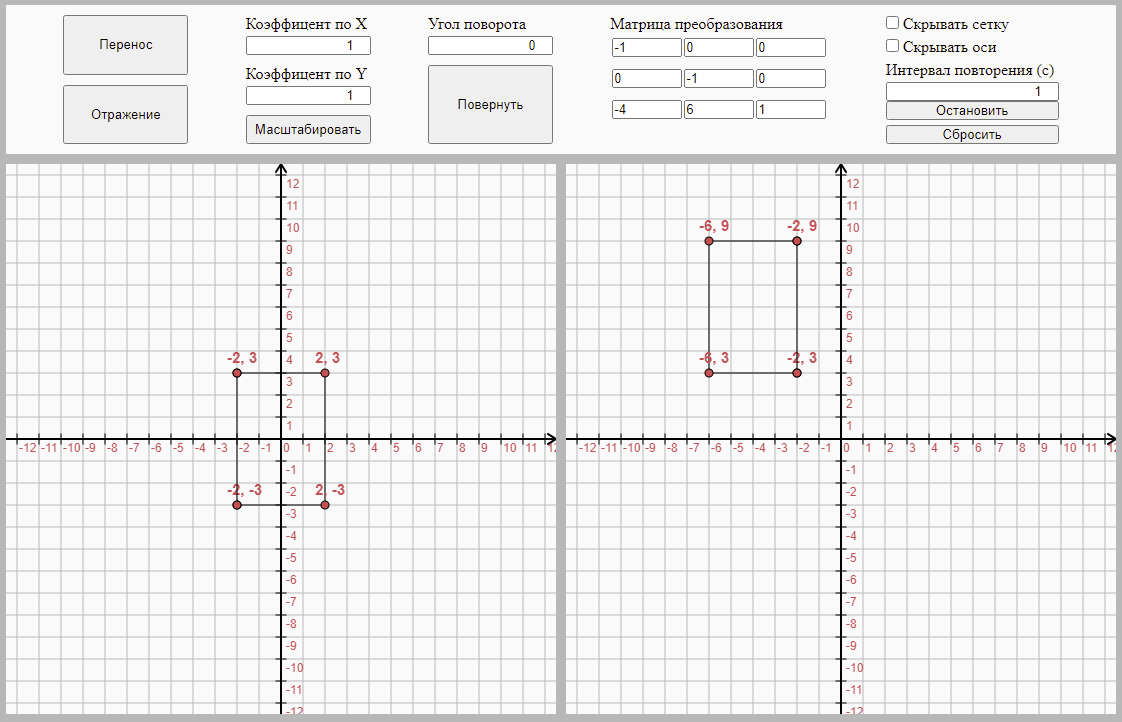


Рисунок 6 - Результат работы приложения после применения отражения относительно верхней левой вершины

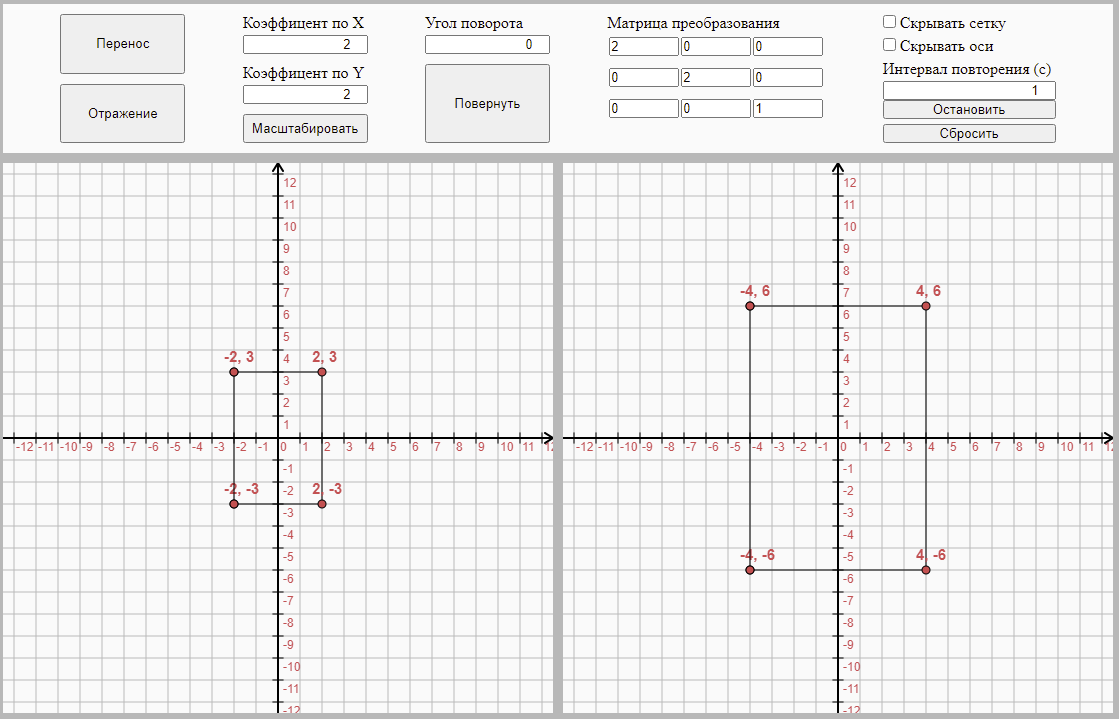


Рисунок 7 – Результат работы приложения после применения масштабирования

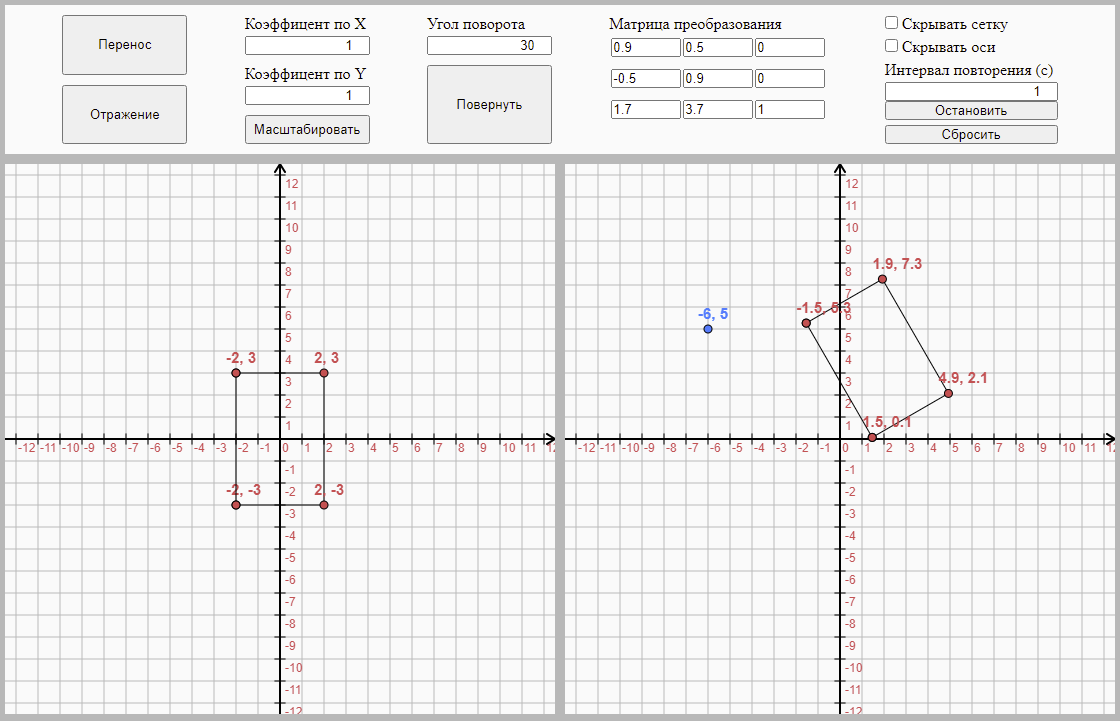


Рисунок 8 – Результат работы приложения после применения поворота

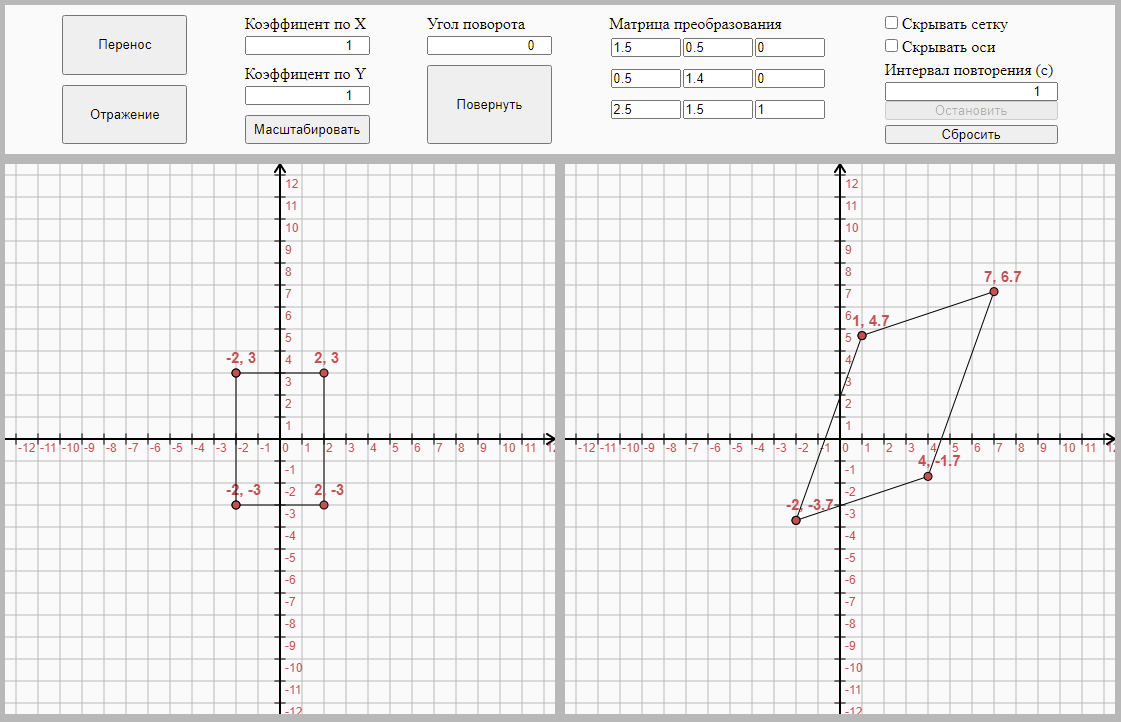


Рисунок 9 – Результат работы приложения после внесения произвольных изменений в матрицу преобразования

**Выводы**

В результате выполнения работы были изучены математические методов аффинных преобразований на плоскости и практически освоены приёмы программной реализации аффинных преобразований.