**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Національний університет «Львівська політехніка»**

**ІКНІ**

Кафедра **ПЗ**

**ЗВІТ**

до лабораторної роботи № 1

**на тему:** «Побудова двовимірних фігур засобами мови програмування»

**з дисципліни**:«Комп’ютерна графіка»

**Виконав:**

ст. гр. ПЗ – 27

Шрібак Б.С.

**Прийняв:**

асистент каф. ПЗ  
Ярема Н.П.

Львів

**2024**

**Тема**: Побудова двовимірних зображень.

**Мета**: Навчитись будувати двомірні зображення з допомогою графічних примітивів мови програмування.

**Теоретичні відомості**

Для виконання лабораторної роботи був вибраний HTML-елемент <Canvas>. Робота з елементом досить проста, вимагає базових знань HTML та JavaScript. Елемент є одним із найпоширеніших інструментів для візуалізації 2D-графіки в Інтернеті.

Однією з ключових операцій є отримання контексту полотна виведення за допомогою методу getContext(). Цей контекст надає методи та властивості для ефективного малювання в межах Canvas. Полотно спочатку порожнє і прозоре.

Були використані такі методи:

fillRect(x, y, width, height) – заповнений прямокутник;

strokeRect(x, y, width, height) – прямокутний контур;

clearRect(x, y, width, height) – очищення й прозорість прямокутної області. Контур малюють за допомогою таких методів:

beginPath() – створити контур;

closePath() – закрити контур;

stroke() – обведення контуру;

fill() – заповнення внутрішньої області;

lineTo(x,y) – проведення лінії з поточної точки до точки (x, y);

moveTo(x, y) – переміщення у точку (x, y) без проведення лінії;

arc(x, y, r, φ1, φ2) – малювання дуги кола з центром у точці (x, y), радіусом r і кутом у межах від φ0 до φ1.

**Завдання**

Написати програму згідно індивідуального варіанту вибраною мовою програмування з використанням її базових графічних примітивів. Програма має відповідати таким вимогам:

1. Відображення системи координат з початком у центрі області виведення з відповідними підписами та позначками (початок, одиничний відрізок, напрям, назва осей).

2. Задання фігур за введеними координати, що відповідають координатам відповідної побудованої декартової системи, а не координатам області виведення (Canvas).

3. Оптимальний ввід користувачем координат фігури з автоматичним обчисленням за можливості інших координат для уникнення зайвих обчислень користувачем.

4. Передбачити можливість некоректного введення даних.

5. Зручний інтерфейс користувача.

**Індивідуальний варіант(12)**

Побудувати декілька прямокутників за введеними координатами лівої верхньої та правої нижньої вершин лише у другій координатній чверті. Побудувати одну діагональ прямокутника (колір обирається користувачем) та заповнити двома випадковими кольорами два новоутворені трикутники. Забезпечити побудову описаних кіл навколо прямокутників.

**Текс програми:**

**index.html**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

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

  <title>Rectangle Drawing</title>

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

</head>

<body>

  <div class="main">

    <div id="canvas-container">

      <canvas id="canvas" width="1000" height="700"></canvas>

    </div>

    <div class="container">

        <h2>(x1; y1) - ліве верхня вершина</h2>

        <h2>(x2; y2) - права нижня вершина</h2><br><br>

        <label for="x1">X1:</label>

        <input type="number" id="x1">

        <label for="y1">Y1:</label>

        <input type="number" id="y1"> <br><br>

        <label for="x2">X2:</label>

        <input type="number" id="x2">

        <label for="y2">Y2:</label>

        <input type="number" id="y2"> <br><br>

        <button id="drawRectButton" onclick="funcDrawButton()">Draw Rectangle</button> <br><br>

        <div class="diagonal">

          <div>

            <button id="drawDiagonalButton">Draw Diagonal</button> <br><br>

          </div>

          <div>

            <label for="color">Color:</label>

            <input type="color" id="color"> <br><br>

          </div>

        </div>

        <button id="randomTrianglesButton">Randomize Triangles</button> <br><br>

        <button id="drawCircleButton">Draw Circumcircle</button> <br><br>

      </div>

  </div>

  <script src="script.js"></script>

</body>

</html>

**style.css**

#canvas-container {

    width: 1000px;

    height: 700px;

    border: 1px solid #ccc;

  }

  #canvas {

    width: 100%;

    height: 100%;

  }

  .axis {

    stroke: #000;

    stroke-width: 1px;

  }

  .tick {

    stroke: #aaa;

    stroke-width: 0.5px;

  }

  .label {

    font-family: Arial, sans-serif;

    font-size: 10px;

    fill: #000;

  }

  canvas {

    width: 1000px;

    height: 700px;

    border: 1px solid #000;

  }

.main {

    display: flex;

    column-gap: 100px;

}

#drawDiagonalButton {

    margin-top: 5px;

}

.diagonal {

    display: flex;

    column-gap: 20px;

}

.container {

    padding-top: 80px;

}

.circle {

    display: none;

  }

**script.js**

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

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

canvas.width = 1000;

canvas.height = 800;

const width = canvas.width;

const height = canvas.height;

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

ctx.beginPath();

ctx.moveTo(0, height / 2);

ctx.lineTo(width, height / 2);

ctx.moveTo(width / 2, 0);

ctx.lineTo(width / 2, height);

ctx.strokeStyle = '#000';

ctx.stroke();

ctx.fillText(0, width / 2 - 10, height / 2 + 10);

const step = 40;

for (let i = -width / 2; i < width / 2; i += step) {

  ctx.beginPath();

  ctx.moveTo(width / 2 + i, height / 2 - 5);

  ctx.lineTo(width / 2 + i, height / 2 + 5);

  ctx.strokeStyle = 'black';

  ctx.stroke();

  ctx.beginPath();

  ctx.moveTo(width / 2 - 5, height / 2 - i);

  ctx.lineTo(width / 2 + 5, height / 2 - i);

  ctx.strokeStyle = 'black';

  ctx.stroke();

  if (i % 20 === 0) {

    ctx.fillStyle = '#000';

    ctx.font = '10px Arial';

    ctx.fillText(i.toString(), width / 2 + i - 5, height / 2 + 15);

    ctx.fillText(i.toString(), width / 2 - 25, height / 2 - i + 5);

  }

function funcDrawLine(x1, y1, x2, y2) {

  ctx.beginPath();

  ctx.moveTo(x1, y1);

  ctx.lineTo(x2, y2);

  ctx.strokeStyle = 'rgb(97, 97, 97)';

  ctx.stroke();

  ctx.closePath();

}

ctx.font = '20px Arial';

ctx.fillStyle = 'rgb(97, 97, 97)';

funcDrawLine(500, 0, 490, 15);

funcDrawLine(500, 0, 510, 15);

ctx.fillText("y", 520, 15);

funcDrawLine(1000, 400, 990, 390);

funcDrawLine(1000, 400, 990, 410);

ctx.fillText("x", 985, 425);

  function funcDrawButton() {

    const x1 = parseInt(document.getElementById('x1').value) + width / 2;

    const y1 = -1 \* parseInt(document.getElementById('y1').value) + height / 2;

    const x2 = parseInt(document.getElementById('x2').value) + width / 2;

    const y2 = -1 \* parseInt(document.getElementById('y2').value) + height / 2;

    if (isNaN(x1) || isNaN(y1) || isNaN(x2) || isNaN(y2))

    {

      alert('Невірні дані. Поля повинні бути заповнені числами.');

    }

    else if(x1 < 0 || y1 < 0 || x2 > 1000 || y2 > 800)

    {

      alert('Невірні координати. Ви виходите за межі області площини.');

    }

    else if (x1 > 500 || y1 > 400 || x2 > 500 || y2 > 400)

    {

      alert('Невірні координати. Прямокутник має бути розміщений в 2 координатній чверті.');

    }

    else if (x1 > x2 || y2 < y1)

    {

      alert('Невірні координати. x1 має бути меншим за x2 і y1 має бути більшим за y2. Натисніть OK для перезавантаження сторінки.');

    }

    else {

      ctx.beginPath();

      ctx.strokeStyle = 'black';

      ctx.rect(x1, y1, x2 - x1, y2 - y1);

      ctx.stroke();

      ctx.closePath();

    }

    }

    const drawDiagonalButton = document.getElementById('drawDiagonalButton');

    drawDiagonalButton.addEventListener('click', () => {

      const x1 = parseInt(document.getElementById('x1').value) + width / 2;

      const y1 = -1\* parseInt(document.getElementById('y1').value) + height / 2;

      const x2 = parseInt(document.getElementById('x2').value) + width / 2;

      const y2 = -1 \* parseInt(document.getElementById('y2').value) + height / 2;

      const color = document.getElementById('color').value;

      ctx.beginPath();

      ctx.moveTo(x1, y1);

      ctx.lineTo(x2, y2);

      ctx.strokeStyle = color;

      ctx.stroke();

      ctx.closePath();

    });

    function getRandomColor() {

      const r = Math.floor(Math.random() \* 256);

      const g = Math.floor(Math.random() \* 256);

      const b = Math.floor(Math.random() \* 256);

      return `rgb(${r}, ${g}, ${b})`;

    }

    randomTrianglesButton.addEventListener('click', () => {

      const x1 = parseInt(document.getElementById('x1').value) + width / 2;

      const y1 = -1\* parseInt(document.getElementById('y1').value) + height / 2;

      const x2 = parseInt(document.getElementById('x2').value) + width / 2;

      const y2 = -1 \* parseInt(document.getElementById('y2').value) + height / 2;

      const x4 = x2;

      const y4 = y1;

      const x3 = x1;

      const y3 = y2;

      const triangle1 = [[x1, y1], [x4, y4], [x2, y2]];

      const triangle2 = [[x1, y1], [x3, y3], [x2, y2]];

      const color1 = getRandomColor();

      const color2 = getRandomColor();

      ctx.beginPath();

      ctx.moveTo(triangle1[0][0], triangle1[0][1]);

      ctx.lineTo(triangle1[1][0], triangle1[1][1]);

      ctx.lineTo(triangle1[2][0], triangle1[2][1]);

      ctx.closePath();

      ctx.fillStyle = color1;

      ctx.fill();

      ctx.beginPath();

      ctx.moveTo(triangle2[0][0], triangle2[0][1]);

      ctx.lineTo(triangle2[1][0], triangle2[1][1]);

      ctx.lineTo(triangle2[2][0], triangle2[2][1]);

      ctx.closePath();

      ctx.fillStyle = color2;

      ctx.fill();

    });

    const drawCircleButton = document.getElementById('drawCircleButton');

    drawCircleButton.addEventListener('click', () => {

      const x1 = parseInt(document.getElementById('x1').value) + width / 2;

      const y1 = -1\* parseInt(document.getElementById('y1').value) + height / 2;

      const x2 = parseInt(document.getElementById('x2').value) + width / 2;

      const y2 = -1 \* parseInt(document.getElementById('y2').value) + height / 2;

      const color = document.getElementById('color').value;

      const centerX = (x1 + x2) / 2;

      const centerY = (y1 + y2) / 2;

      const radius = Math.sqrt((x1 - x2) \* (x1 - x2) + (y1 - y2) \* (y1 - y2)) / 2;

      ctx.beginPath();

      ctx.strokeStyle = 'rgb(97, 97, 97)';

      ctx.arc(centerX, centerY, radius, 0, 2 \* Math.PI);

      ctx.stroke();

    });

}

**Результати виконання програми**

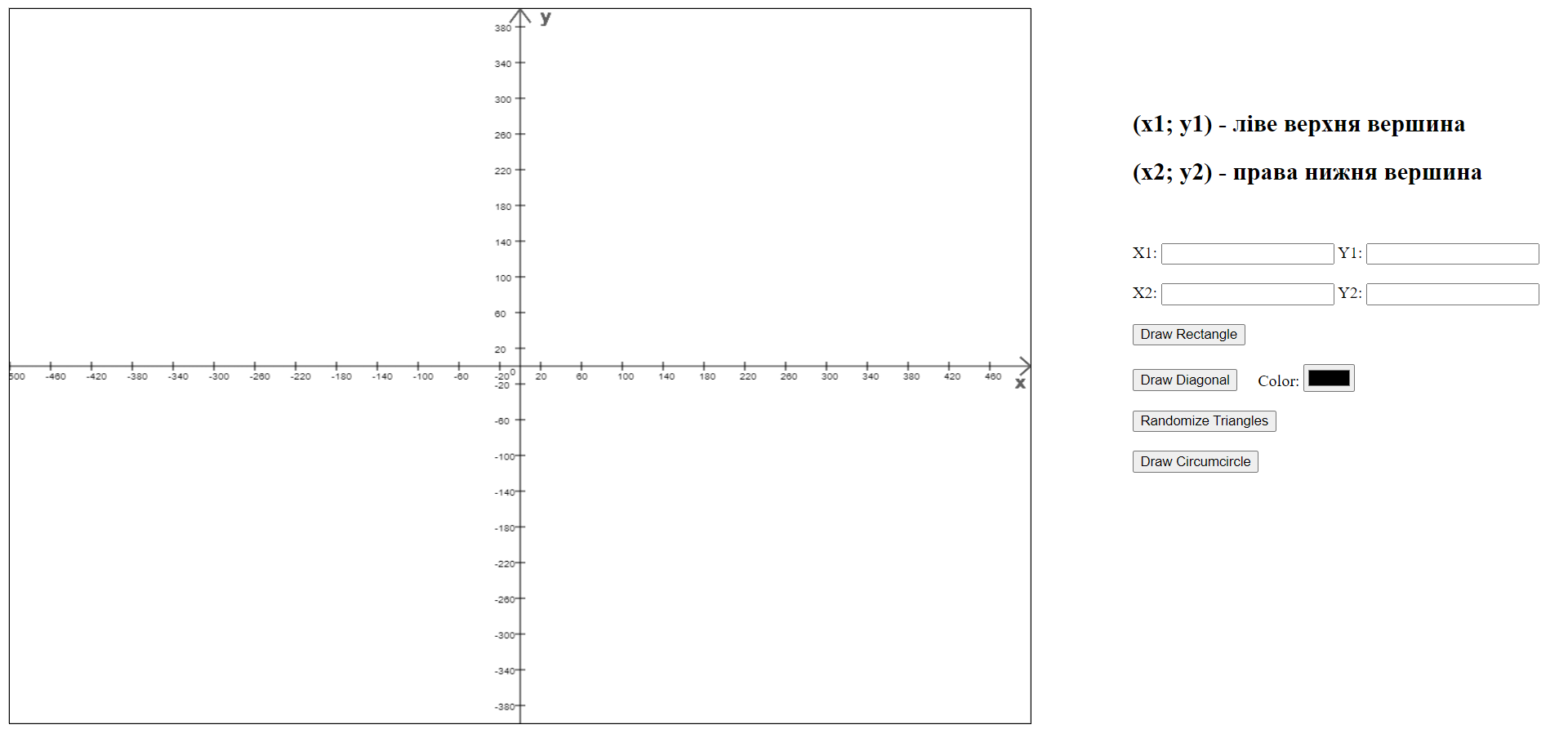
****

Рис. 1. Головне меню програми

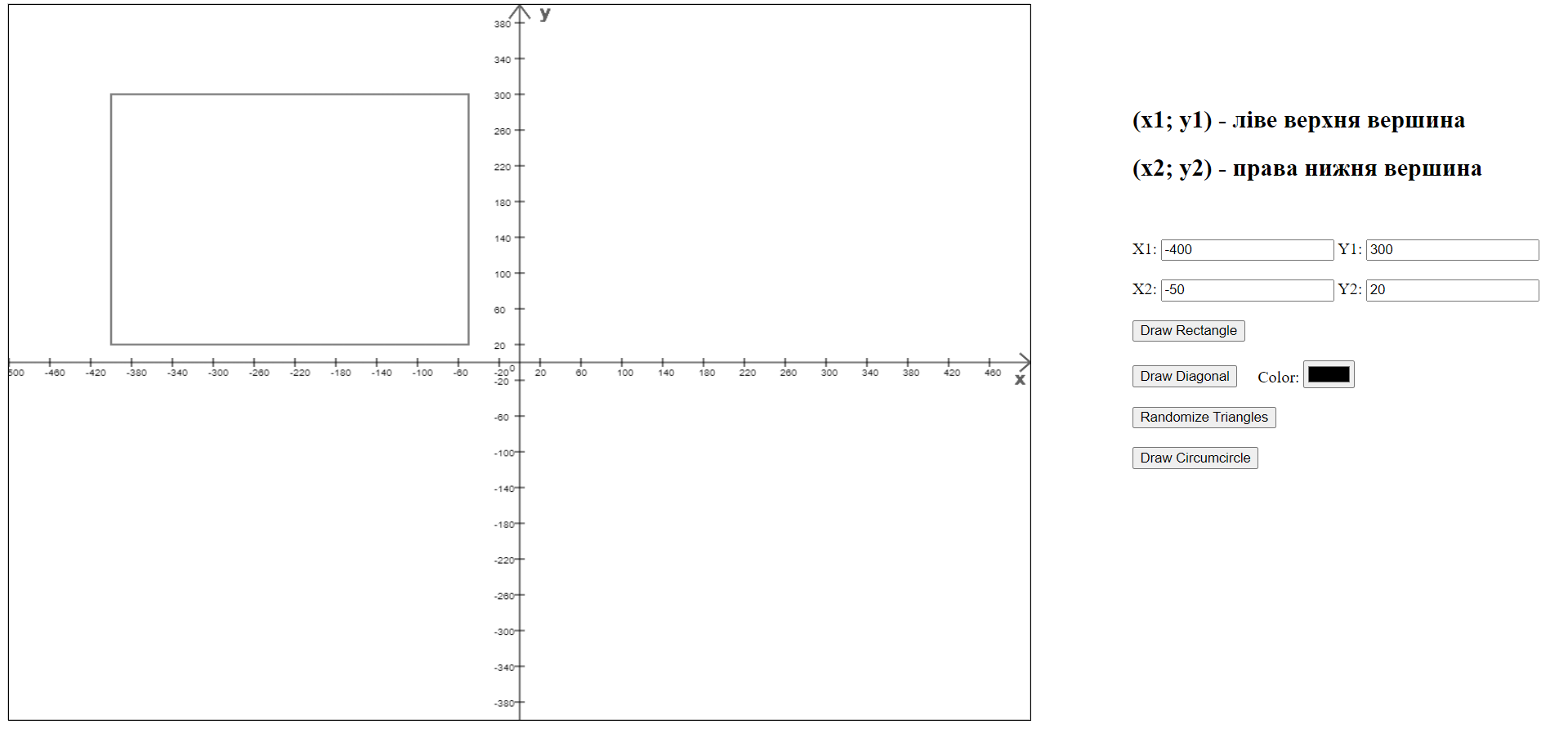


Рис. 2. Малювання прямокутника по заданим координатам.

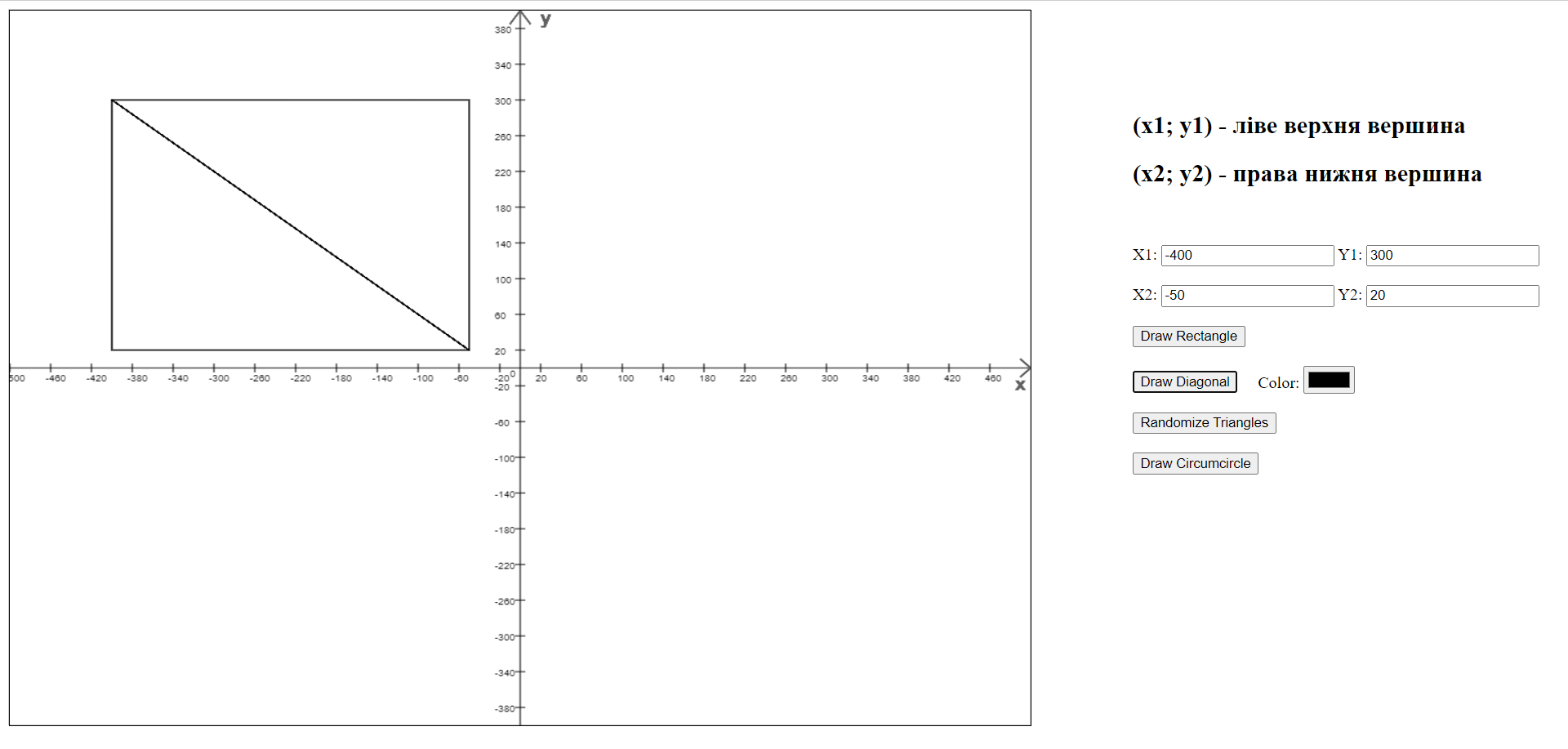


Рис. 3. Малювання діагоналі, кольором, який задав користувач.

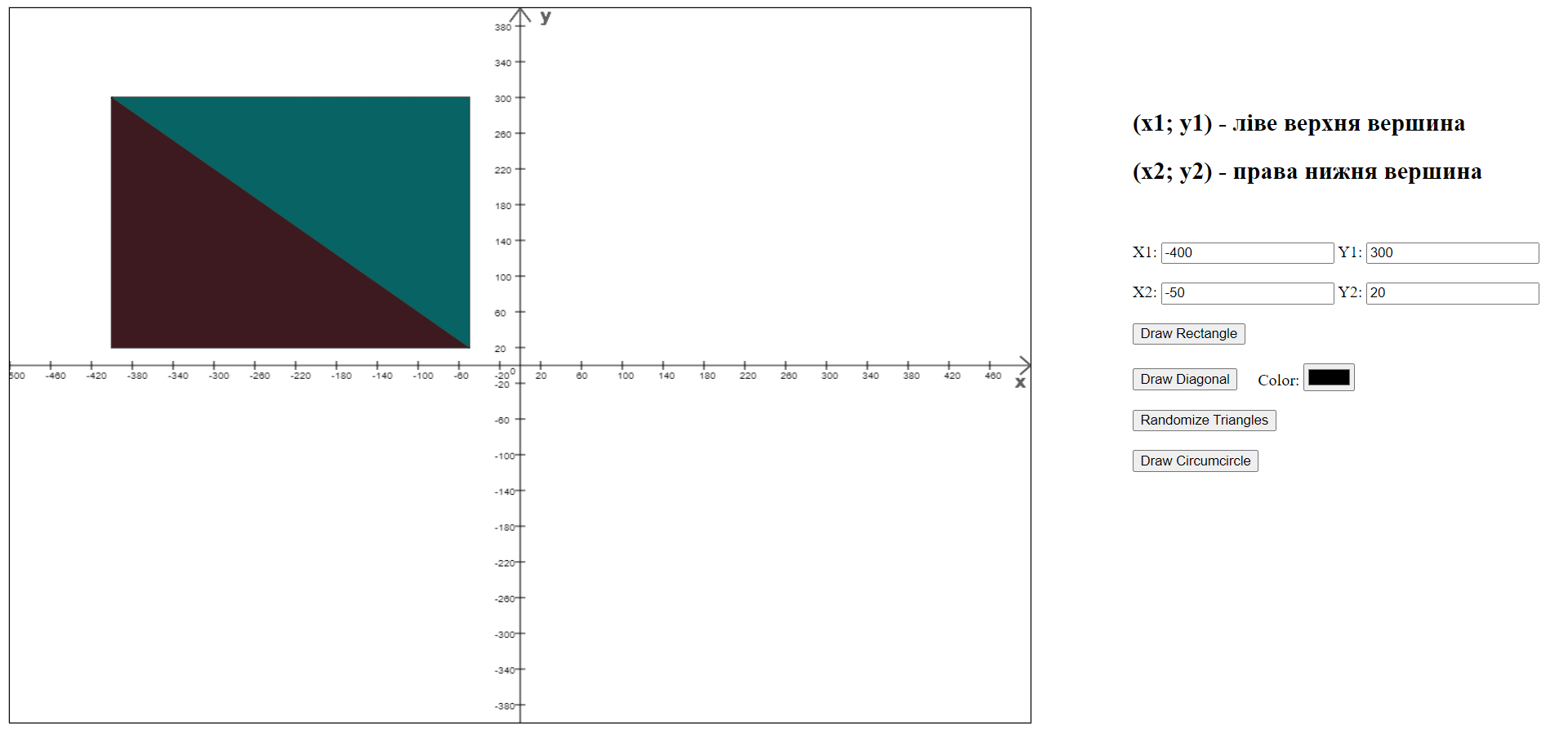


Рис. 4. Рандомне заповнення двох новоутворених трикутників.

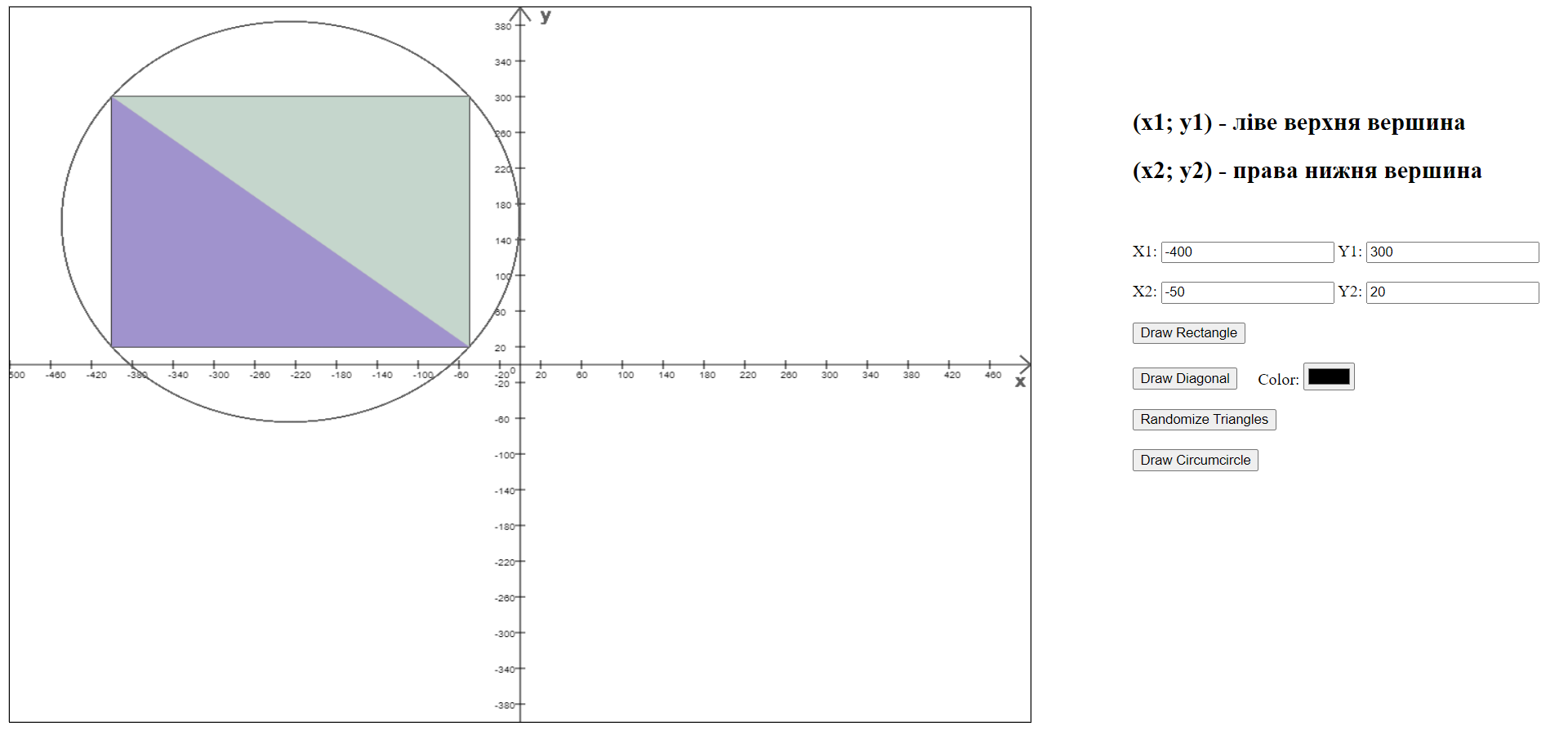


Рис. 5. Малювання описаного кола, навколо прямокутника.

**Висновки**

Під час виконання цієї лабораторної роботи було виконано усі пункти завдання. Мною було вивчено базові методи та алгоритми побудови 2d фігур за допомогою елемента HTML - <Canvas>. Працював над створенням системи координат, малюванню на ній об'єктів, обробкою введення некоректних даних, та можливістю вибору кольору трикутників.