МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Інститут **КНІТ**
Кафедра **ПЗ**

ЗВІТ

До лабораторної роботи № 1
З дисципліни: *“Комп’ютерна графіка”*
На тему: *“Побудова двовимірних зображень”*

Лектор:
доц. каф. ПЗ
Левус Є.В.

Виконала:
ст. гр. ПЗ-26
Пелих О.Р.

Прийняв:
доц. каф. ПЗ
Ярема Н.П.

« ____ » _____ 2025 р.
∑=____ _____

Львів – 2025

Тема роботи: Побудова двовимірних зображень.
Мета роботи: Навчитись будувати двомірні зображення з допомогою графічних примітивів мови програмування.

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.

ХІД ВИКОНАННЯ

ТЕХНОЛОГІЇ:

HTML – використовується для створення структури веб-сторінки та розміщення елементів, таких як форми введення та полотно (canvas) для малювання графіки.
CSS – відповідає за стилізацію сторінки, зокрема оформлення форм та розташування елементів.
JavaScript (Canvas API) – використовується для малювання графічних об'єктів. Методи на кшталт getContext, beginPath, moveTo, lineTo, stroke та fill допомагають керувати зображенням на полотні.
Переваги обраних технологій:
HTML, CSS та JavaScript є основними інструментами для розробки веб-сторінок, що гарантує їхню підтримку у всіх сучасних браузерах. Canvas API є простим у використанні та дозволяє ефективно працювати з графікою.
Основні функції та методи в коді:
drawDec() – будує координатну пряму.
draw() – отримує введені дані, перевіряє їх коректність і, якщо все правильно, будує коло та вписаний у нього шестикутник.

ВАРІАНТ 6

6. Побудувати декілька рівносторонніх трикутників за введеними координатами вершин однієї сторони та з можливістю вибору кольору заливки та вигляду вершин трикутника (у вигляді квадратиків чи кружечків).

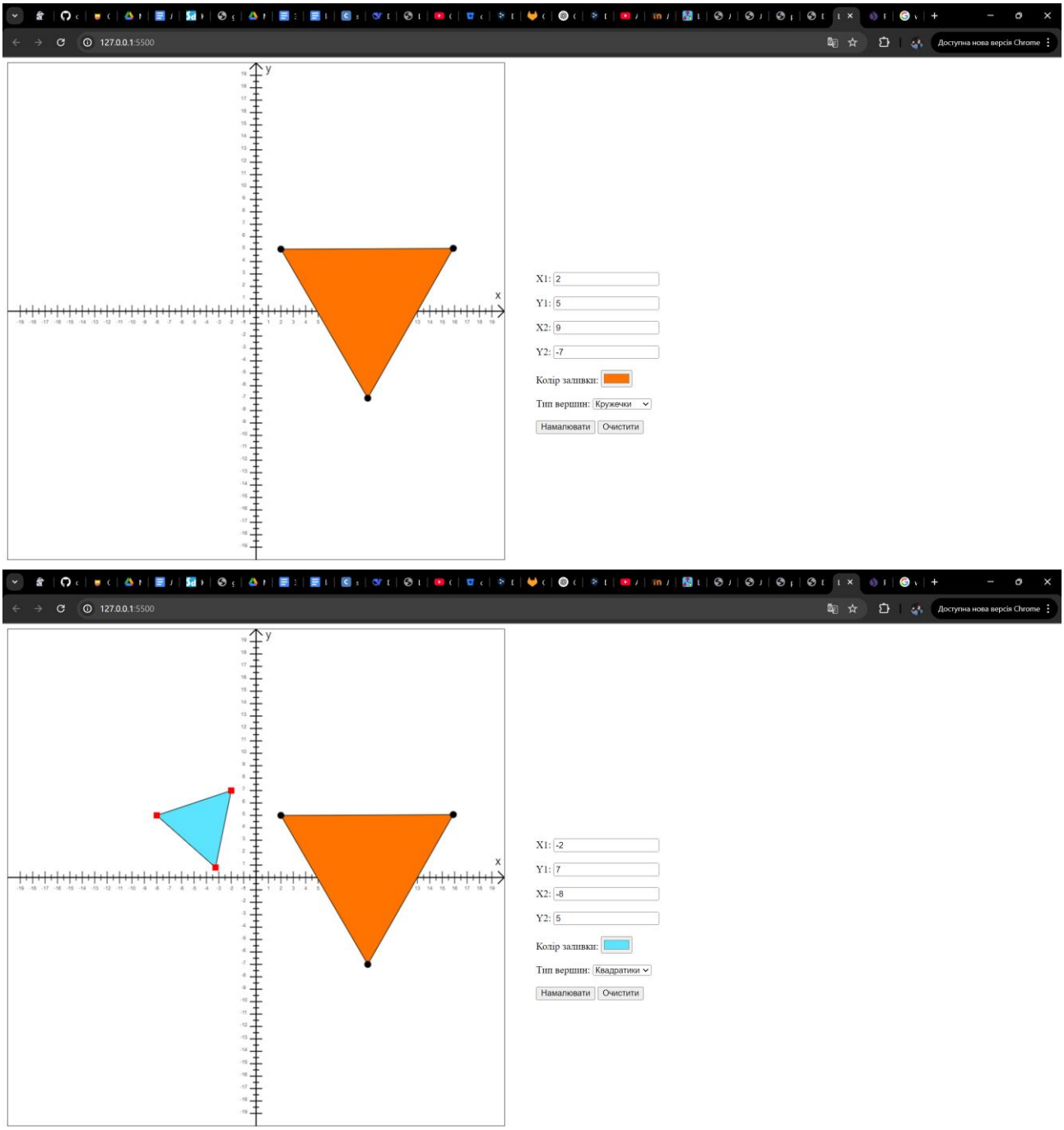

Рис.1-2. Приклад виконання програми

КОД ПРОГРАМИ

Index.html

```
<!DOCTYPE html>
<html lang="en-UK">
  <head>
    <link rel="stylesheet" href="./css/style.css" />
    <meta charset="UTF-8" />
    <meta name="author" content="Olha Pelykh" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Lab1 Pelykh</title>
  </head>
  <body>
    <canvas id="myCanvas" width="800" height="800"></canvas>
    <div id="data">
      <label for="x1">X1:</label>
      <input type="number" id="x1" name="x1" /><br /><br />
      <label for="y1">Y1:</label>
      <input type="number" id="y1" name="y1" /><br /><br />
      <label for="x2">X2:</label>
      <input type="number" id="x2" name="x2" /><br /><br />
      <label for="y2">Y2:</label>
      <input type="number" id="y2" name="y2" /><br /><br />
      <label for="fillColor">Колір заливки:</label>
      <input type="color" id="fillColor" name="fillColor" /><br /><br />
      <label for="vertexType">Тип вершин:</label>
      <select id="vertexType">
        <option value="circle">Кружечки</option>
        <option value="square">Квадратики</option></select
      ><br /><br />
      <button id="draw">Намалювати</button>
      <button id="clear">Очистити</button>
    </div>
    <script type="text/javascript" src="./js/script.js"></script>
  </body>
</html>
```

Style.css

```
canvas {
  border: 1px solid black;
}

body {
  display: flex;
}

#data {
  margin-left: 3%;
  margin-top: 20%;
}
```

Script.js

```
const canvas = document.getElementById("myCanvas");
const ctx = canvas.getContext("2d");

drawDec();

function drawDec() {
  ctx.font = "18px Arial";
  ctx.fillStyle = "black";
  ctx.textAlign = "center";
  ctx.fillText("x", 790, 380);
  ctx.fillText("y", 420, 15);
  ctx.moveTo(0, 400);
  ctx.lineTo(800, 400);
  ctx.lineTo(790, 390);
  ctx.moveTo(800, 400);
  ctx.lineTo(790, 410);
  ctx.moveTo(400, 0);
  ctx.lineTo(390, 10);
  ctx.moveTo(400, 0);
  ctx.lineTo(410, 10);
  ctx.moveTo(400, 0);
  ctx.lineTo(400, 800);
  ctx.stroke();

  for (let i = 390, j = 20; j <= 780; j += 10) {
    if (j % 20 === 0) {
      ctx.moveTo(i, j);
      ctx.lineTo(i + 20, j);
    } else {
      ctx.moveTo(i + 5, j);
      ctx.lineTo(i + 15, j);
    }
  }
  ctx.stroke();

  for (let i = 20, j = 390; i <= 780; i += 10) {
    if (i % 20 === 0) {
      ctx.moveTo(i, j);
      ctx.lineTo(i, j + 20);
    } else {
      ctx.moveTo(i, j + 5);
      ctx.lineTo(i, j + 15);
    }
  }
  ctx.stroke();

  ctx.font = "8px Arial";
  for (
    let i = -((canvas.width - 40) / 20 / 2),
      xx = 20,
      xy = 420,
```

```
      yx = 380,
      yy = 20;
    i <= (canvas.width - 40) / 20 / 2;
    i++, xx += 20, yy += 20
  ) {
    if (i !== 0) {
      ctx.fillText(i.toString(), xx, xy);
      ctx.fillText((-i).toString(), yx, yy);
    }
  }
}

function draw() {
  const x1 = parseFloat(document.getElementById("x1").value);
  const y1 = parseFloat(document.getElementById("y1").value);
  const x2 = parseFloat(document.getElementById("x2").value);
  const y2 = parseFloat(document.getElementById("y2").value);
  const fillColor = document.getElementById("fillColor").value;
  const vertexType = document.getElementById("vertexType").value;

  const dx = x2 - x1;
  const dy = y2 - y1;
  const sideLength = Math.sqrt(dx * dx + dy * dy);

  const height = (Math.sqrt(3) / 2) * sideLength;

  const x3 = x1 + dx / 2 - dy * (height / sideLength);
  const y3 = y1 + dy / 2 + dx * (height / sideLength);

  ctx.beginPath(); /*починаю новий шлях для малювання*/
  ctx.moveTo(
    x1 * 20 + 400,
    400 - y1 * 20
  ); /*переміщує перо до першої вершини */
  ctx.lineTo(x2 * 20 + 400, 400 - y2 * 20); /*малює лінії до інших вершин*/
  ctx.lineTo(x3 * 20 + 400, 400 - y3 * 20);
  ctx.closePath(); /*замикає шлях, з'єднуючи останню точку з першою*/
  ctx.fillStyle = fillColor;
  ctx.fill();
  ctx.stroke();

  drawVertex(x1 * 20 + 400, 400 - y1 * 20, vertexType);
  drawVertex(x2 * 20 + 400, 400 - y2 * 20, vertexType);
  drawVertex(x3 * 20 + 400, 400 - y3 * 20, vertexType);
}

function drawVertex(x, y, type) {
  if (type === "circle") {
    ctx.beginPath();
    ctx.arc(
      x,
      y,
      5,
```

```
      0,
      2 * Math.PI
    ); /*малює коло з центром у (x, y), радіусом 5 пікселів, від кута 0 до 2π
(повне коло)*/
    ctx.fillStyle = "black";
    ctx.fill();
    ctx.stroke();
  } else if (type === "square") {
    ctx.fillStyle = "red";
    ctx.fillRect(
      x - 5,
      y - 5,
      10,
      10
    ); /*x - 5, y - 5 — зміщення координат, щоб центр квадрата був у точці (x, y)*/
  }
}

function clear() {
  ctx.clearRect(0, 0, canvas.width, canvas.height);
  drawDec();
}

document.getElementById("draw").addEventListener("click", draw);
document.getElementById("clear").addEventListener("click", clear);
```

ВИСНОВКИ

Під час виконання цієї лабораторної роботи я навчилася створювати двовимірні зображення, використовуючи графічні примітиви мови програмування. Робота з графікою та координатами допомогла мені краще зрозуміти систему координат, відображення фігур та їх параметри.

Також я реалізувала можливість вибору кольору заливки трикутників та налаштування вигляду їх вершин (квадратики або кружечки), що допомогло краще опанувати роботу з графічними примітивами та їх стилізацією.

У процесі розробки програми я вивчила та застосувала основні технології, такі як HTML, CSS і JavaScript. Найважчим для мене було зрозуміти систему пристрою виведення, та як перетворити її у звичну декартову.