МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ПРИКАРПАТСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТЕФАНИКА

Кафедра комп’ютерних наук та інформаційних систем

# Лабораторна робота № 1

# з курсу «Розробка WEB-додатків (TypeScript)»

на тему:

«Застосування TypeScript для розв’язання прикладних задач за допомогою консолі браузера»

# Виконав:

студент групи КН-41 Кадемський О. В.

«18» лютого 2025 р.

(підпис)

# Перевірив:

ст. викладач кафедри КНІС к.т.н. Ізмайлов А. В.

«\_\_» 2025р.

(оцінка, підпис)

м. Івано-Франківськ 2025

**Мета роботи:** Навчитись застосовувати TypeScript для розв’язання прикладних задач за допомогою консолі браузера.

# Теоретичні відомості:

TypeScript (TS) – це JavaScript (JS) із синтаксисом для типів. TS є строго типізованою мовою та є інструментом для розробників, який покликаний допомогти створювати якісніший JS-код.

TS компілюється у JS і фактично є розширенням (superset) JS. Процес компіляції TS полегшує та стабілізує процес JS-розробки, оскільки дозволяє виявляти та виправляти помилки, які, у загальному випадку, довелося б виявляти уручну при традиційному процесі розробки у JS.

Для встановлення TS слід встановити Node Package Manager (npm). Після цього можливе встановлення TS.

Глобальне встановлення TS: npm i typescript -g

Для швидкого тестування успішності встановлення TS слід:

• у порожній папці підготувати мінімальну html-сторінку до якої

підключатиметься скрипт;

• створити файл для TS, наприклад main.ts;

• внести у main.ts довільний код, наприклад:

let username = ‘John’;

console.log(username);

• у консолі (PowerShell, Bash, тощо) запустити команду TS (tsc - TypeScript Command): tsc main.ts

# Хід роботи:

**1.2.1** У створеному на GitHub репозиторії для курсу створити нову підпапку для завдання лабораторної:



**1.2.2** У всіх наступних завданнях слід чітко вказувати типи змінних та

аргументів функцій.

**1.2.3** Написати JavaScript-функцію triangle (скомпільовану із TypeScript), яка розв'язує прямокутний трикутник (знаходить всі його сторони та обидва гострі кути) за двома заданими елементами і їх «типами» (катет, прилеглий кут, гіпотенуза і т.д.) і яка може бути запущена у консолі для певних значень аргументів:

console.info("Інструкції із використання:")

console.info("Використання: triangle(value1, type1, value2, type2)");

console.info("Допустимі типи: 'leg', 'hypotenuse', 'adjacent angle', 'opposite angle', 'angle'");

console.info("Приклади:");

console.info("triangle(4, 'leg', 8, 'hypotenuse')");

console.info("triangle(8, 'hypotenuse', 4, 'leg')");

type TAngle = 'leg' | 'hypotenuse'| 'adjacent angle' | 'opposite angle' | 'angle';

function triangle(val1: number, type1: TAngle, val2: number, type2: TAngle): string {

    const validTypes: TAngle[] = ['leg', 'hypotenuse', 'adjacent angle', 'opposite angle', 'angle'];

    const angleTypes: TAngle[] = ['adjacent angle', 'opposite angle', 'angle'];

    // Type Validation

    if (!validTypes.includes(type1) || !validTypes.includes(type2)) {

        console.error("Не правильний тип. Дивіться інструкції!.");

        return "failed";

    }

    if ((type1 === 'angle' && type2 !== 'hypotenuse') || (type2 === 'angle' && type1 !== 'hypotenuse')) {

        console.log("Тип 'angle' повинен бути заданий в парі з 'hypotenuse'.");

        return "failed";

    }

    if (type1 === 'hypotenuse' && type2 === 'hypotenuse') {

        console.error("Неможливо вказати два значення гіпотенузи.");

        return "failed";

    }

    if (angleTypes.includes(type1) && angleTypes.includes(type2)) {

        console.error("Неправильна комбінація. Вказано два кути.");

        return "failed";

    }

    // Value Validations

    const value1: number = Number(val1);

    const value2: number = Number(val2);

    if (isNaN(value1) || isNaN(value2) || value1 <= 0 || value2 <= 0) {

        console.error("Значення повинні бути додатніми числами.");

        return "failed";

    }

    const checkAngle = (type: TAngle, value: number): boolean => {

        if (angleTypes.includes(type) && (value <= 0 || value >= 90)) {

            console.error("Кути повинні бути від 0 до 90 градусів.");

            return false;

        }

        return true;

    };

    if (!checkAngle(type1, value1) || !checkAngle(type2, value2)) {

        return "failed";

    }

    if ((type1 === 'hypotenuse' && type2 === 'leg') || (type2 === 'hypotenuse' && type1 === 'leg')) {

        const hypotenuse: number = type1 === 'hypotenuse' ? value1 : value2;

        const leg: number = type1 === 'leg' ? value1 : value2;

        if (hypotenuse <= leg) {

            console.error("Гіпотенуза повинна бути довшою за катет.");

            return "failed";

        }

    }

    // Calculation

    let a: number | undefined;

    let b: number | undefined;

    let c: number | undefined;

    let alpha: number | undefined;

    let beta: number | undefined;

    const processType = (type: TAngle, value: number): void => {

        switch (type) {

            case 'leg':

                if (a === undefined) a = value;

                else b = value;

                break;

            case 'hypotenuse':

                c = value;

                break;

            case 'adjacent angle':

                alpha = value;

                break;

            case 'opposite angle':

                beta = value;

                break;

            case 'angle':

                alpha = value;

                break;

        }

    };

    processType(type1, value1);

    processType(type2, value2);

    try {

        if (a !== undefined && b !== undefined) { // Два катети

            c = Math.sqrt(a \*\* 2 + b \*\* 2);

            alpha = Math.atan(a / b) \* 180 / Math.PI;

            beta = 90 - alpha;

        } else if (a !== undefined && c !== undefined) { // Катет і гіпотенуза

            b = Math.sqrt(c \*\* 2 - a \*\* 2);

            alpha = Math.atan(a / b) \* 180 / Math.PI;

            beta = 90 - alpha;

        } else if (a !== undefined && alpha !== undefined) { // Катет і прилеглий кут (a, A)

            c = a / Math.cos(alpha \* Math.PI / 180);

            b = Math.sqrt(c \*\* 2 - a \*\* 2);

            beta = 90 - alpha;

        } else if (a !== undefined && beta !== undefined) { // Катет і протилежний кут (a, B)

            c = a / Math.sin(beta \* Math.PI / 180);

            b = Math.sqrt(c \*\* 2 - a \*\* 2);

            alpha = 90 - beta;

        } else if (c !== undefined && alpha !== undefined) { // Гіпотенуза і кут/прилеглий кут (c, A)

            b = c \* Math.sin(alpha \* Math.PI / 180);

            a = Math.sqrt(c \*\* 2 - b \*\* 2);

            beta = 90 - alpha;

        } else if (c !== undefined && beta !== undefined) { // Гіпотенуза і протилежний кут (c, B)

            a = c \* Math.sin(beta \* Math.PI / 180);

            b = Math.sqrt(c \*\* 2 - a \*\* 2);

            alpha = 90 - beta;

        } else if (alpha !== undefined && beta !== undefined) { // Два кута (A, B)

            throw new Error("Введено два кути.");

        } else {

            throw new Error("Неправильна комбінація типів.");

        }

        if (

            (a !== undefined && a <= 0) ||

            (b !== undefined && b <= 0) ||

            (c !== undefined && c <= 0) ||

            (alpha !== undefined && (alpha <= 0 || alpha >= 90)) ||

            (beta !== undefined && (beta <= 0 || beta >= 90))

        ) {

            throw new Error("Не коректно введені значення. \n І ні, 5.729577951308233e-18 не прокатить");

        }

        console.log(`a = ${a}`);

        console.log(`b = ${b}`);

        console.log(`c = ${c}`);

        console.log(`alpha = ${alpha}°`);

        console.log(`beta = ${beta}°`);

        return "success";

    } catch (error) {

        console.error(error instanceof Error ? error.message : "Помилка при розрахунках");

        return "failed";

    }

}

**1.2.4** Надати викладачеві посилання на зроблену веб-сторінку, розміщену на сервері GitHub Pages (посилання повинне бути у форматі: <https://username.github.io/userrepo/userdirectory>), а також, посилання на безпосередньо сам репозиторій (посилання повинне бути у форматі: <https://github.com/username/userrepo>):

<https://alexander0kd.github.io/TS-labs/lab1/dist/>

<https://github.com/Alexander0kd/TS-labs>

**Висновки:** в ході виконання даної лабораторної роботи ми навчились застосовувати TypeScript для розв’язання прикладних задач за допомогою консолі браузера.