**Лабораторна робота №2**

***Типи даних. Приведення типів. Регулярні вирази.***

***Робота з циклами у мові JavaScript.***

***Мета:*** вивчити відмінності у роботі операцій та операторів у мові JavaScript від інших мов; ознайомитись з базовими типами даних; вивчити особливості приведення типів, навчитися працювати з циклами.

**Хід Роботи:**

***Завдання 1.*** Виконайте наступні завдання для самоперевірки. Результати представити в звіті у вигляді скріншотів виконання скриптів.

Визначте, які значення будуть виведені на екран:

let a = 2;

let b = 3;

let c, d;

c = ++a; // a = 2 + 1 = 3, c = a = 3

d = b++; //d = b = 3, b = 3 + 1 = 4

c = (2+ ++a); //a = 3 + 1 = 4, c = 2 + 4 = 6

d = (2+ b++); //d = 2 + 4 = 6, b = 5

console.log(a); //a = 4

console.log(b); //b = 5

console.log(c); //c = 6

console.log(d); //d = 6

html:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Task1.1</title>

</head>

<body>

<script src="t1 jscode.js"></script>

</body>

</html>

js:

let a = 2;

let b = 3;

let c, d;

c = ++a;

d = b++;

c = (2+ ++a);

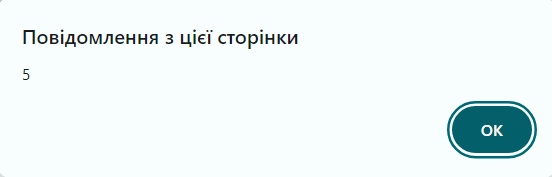
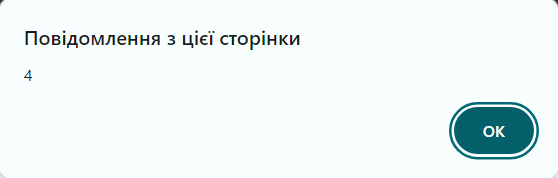
d = (2+ b++);

alert(a);

alert(b);

alert(c);

alert(d);



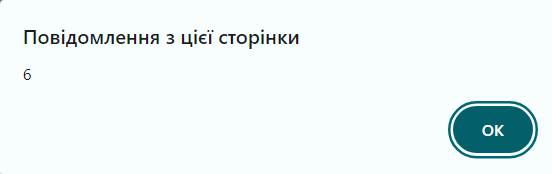
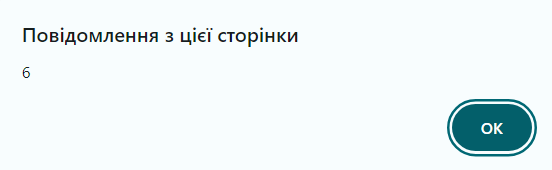


Рис. 1.1 Перевірка повідомленнями

Визначте, яке значення х буде отримано в результаті виконання коду:

let a = 5;

let x = a \*= (a \*= 2); // x = a\*(a\*2) = 5\*(5\*2) = 50

console.log(x); //50

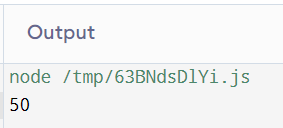


Рис. 1.2 Перевірка розрахунку

Які значення будуть виведені наступним кодом і чому?

* console.log(console.log("Hello, world")) //Hello, world та undefined, бо зовнішня функція console.log не має що повертати, тому undefined, фнутрішня функція спрацювала і вивела Hello world.

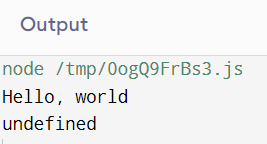


Рис. 1.3 Перевірка

* console.log(null || undefined) //undefined, бо undefined та null обидва мають значення false, виводиться крайнє значення.

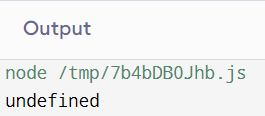


Рис. 1.4 Перевірка

* console.log(null || 2 || undefined) //2, бо 2 є єдиним значенням true

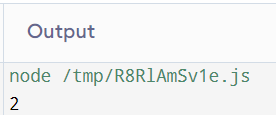


Рис. 1.5 Перевірка

* console.log(1 && null && 2) //null, бо в логічному операторі and якщо одне значення дорівнює null, то відповідь буде null

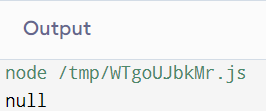


Рис. 1.6 Перевірка

* console.log(1 && console.log("Hello")) //Hello та undefined, бо зовнішня console.log повертає false, позаяк у логічному and якщо одне дорівнює false, результат буде false.

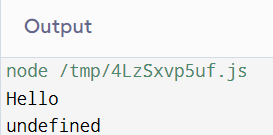


Рис. 1.6 Перевірка

Пояснити які значення і яких типів будуть давати наступні вирази:

console.log("" + 1 + 0); //10, string

console.log("" - 1 + 0); //-1, number

console.log(true + false); //1 + 0 = 1, number

console.log(6 / "3");//2, number

console.log("2" \* "3");//6, number

console.log(4 + 5 + "px");//9px, string через частину px

console.log("$" + 4 + 5); //$45, string через частину $

console.log("4" - 2); //2, number

console.log("4px" - 2); //nan, number

console.log(7 / 0); //infinity, number через ділення на нуль

console.log(" -9\n" + 5); //-9\n 5, string значення є рядком

console.log(" -9\n" - 5); //-14, number

console.log(5 && 2); //2, number, останнє значення

console.log(2 && 5); //5, number, останнє значення

console.log(5 || 0); //5, number, більше значення

console.log(0 || 5); //5, number, більше значення

console.log(null + 1); //1, number

console.log(undefined + 1); //nan, number

console.log(null == "\n0\n");//false, boolean

console.log(+null == +"\n0\n");//true, boolean

Василькові потрібно написати функцію toCamelCase(), яка приймає значення імені змінної в стилі “Snake case”, та повертає це ж ім’я в стилі «Lower camel case». Використати регулярні вирази.

Для прикладу: Значення, що передається в функцію:

Результат роботи функції

my\_variable

myVariable

new\_brand\_product

newBrandProductfunction toCamelCase(snakeCase) {

    return snakeCase.replace(/\_([a-z])/g, function(match, char) {

        return char.toUpperCase();

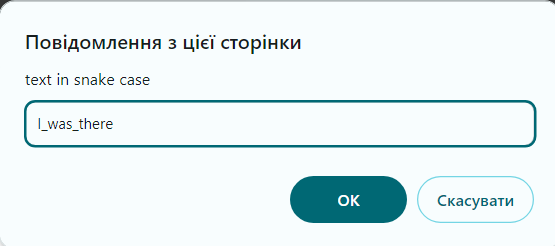
    });

}

let str = prompt('text in snake case', '');

str = toCamelCase(str);

alert(`text in camel case: ${str}`);

****

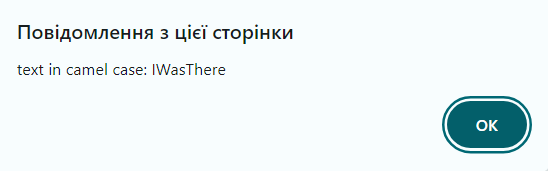
****

Рис. 1.7 Робота функції

Оленці в свою чергу потрібно написати функцію toSnakeCase(), яка працює як обернена до функції toCamelCase(). Допоможіть їй це зробити. Використати регулярні вирази.

function toSnakeCase(camelCase) {

    return camelCase.replace(/[A-Z]/g, function(match) {

        return '\_' + match.toLowerCase();

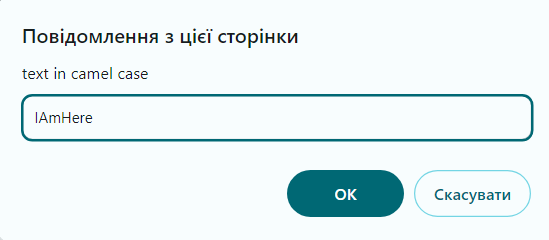
    });

}

str = prompt('text in camel case', '');

str = toSnakeCase(str);

alert(`text in snake case: ${str}`);



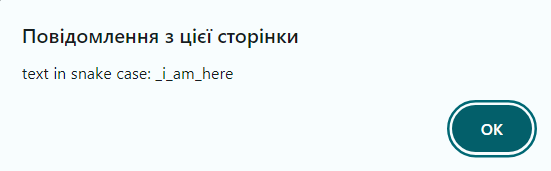


Рис. 1.8 Робота функції

Андрійко повинен в тексті замінити всі дати формату yyyy/mm/dd на формат dd.mm.yyyy. Текст може бути як завгодно великий і одна з вимог - використати пошук і заміну з використанням регулярних виразів. Текст вводить користувач.

function replaceDates(text) {

    return text.replace(/\b(\d{4})\/(\d{2})\/(\d{2})\b/g, function(match, year, month, day) {

        return day + '.' + month + '.' + year;

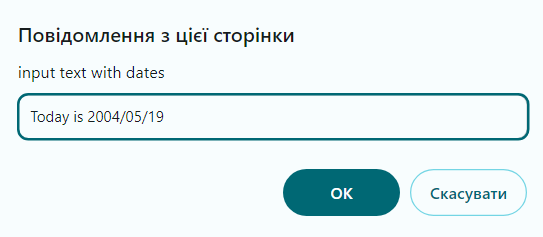
    });

}

str = prompt('input text with dates', '');

str = replaceDates(str);

alert(`text with dates in different format: ${str}`);



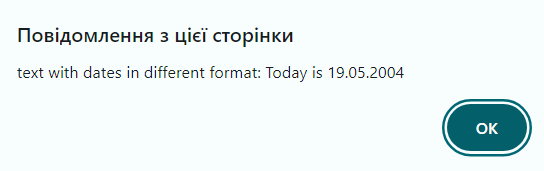


Рис. 1.9 Робота функції

Виконати завдання на закріплення роботи з циклами.

* Вивести всі числа від 1 до 100, які кратні зазначеному користувачем числу.

let n = parseInt(prompt('input number'));

let res = "";

for (let i = 1; i <= 100; i++) {

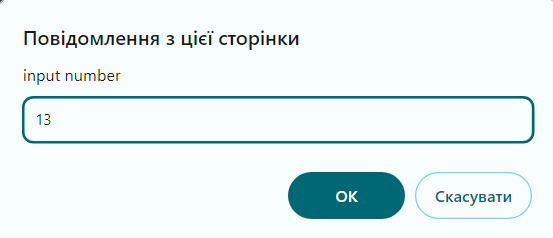
    if (i % n === 0) {

        res += i + " ";

    }

}

alert(res);



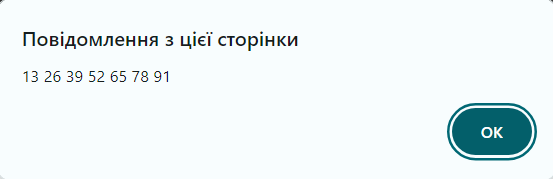


Рис. 2.1 Робота циклу

* Вивести кожен 4-й елемент із зазначеного користувачем діапазону. Користувач вказує мінімальне та максимальне значення діапазону.

let min = parseInt(prompt('min:'));

let max = parseInt(prompt('max:'));

res = "";

for (let i = min; i <= max; i++) {

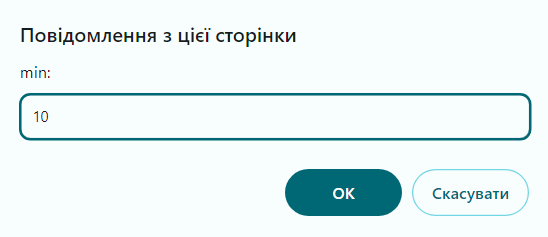
    if ((i - min) % 4 === 0) {

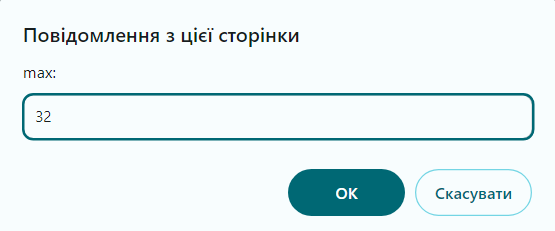
        res += i + " ";

    }

}

alert(res);





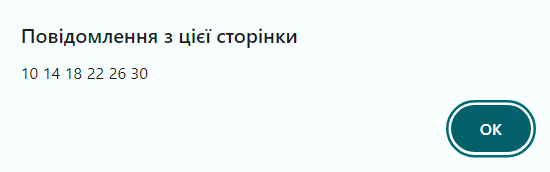


Рис. 2.2 Робота циклу

Дано два цілих числа A і B (A <B). Якщо користувач помилково ввів А> B, то поміняти ці числа місцями. Вивести всі числа від А до В. Знайти суму всіх цілих чисел від A до B включно. Знайти кількість виведених чисел.

let a = parseInt(prompt('a:'));

let b = parseInt(prompt('b:'));

if (a > b) {

    let temp = a;

    a = b;

    b = temp;

}

let sum = 0;

let count = 0;

res = "";

for (let i = a; i<=b; i++) {

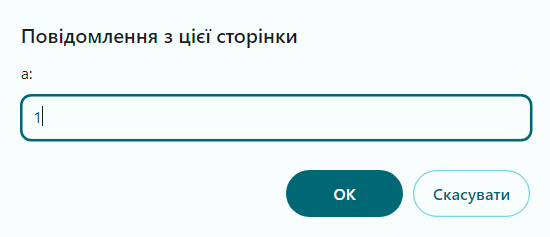
    res += i + " ";

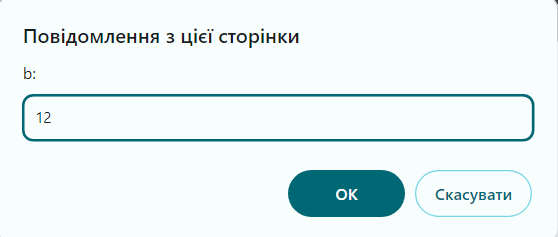
    sum += i;

    count ++;

}

alert(res + `\n s: ${sum} \n c: ${count}`);





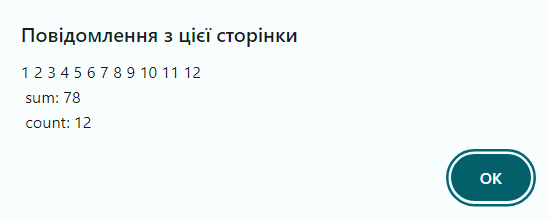


Рис. 2.3 Робота циклу та обрахунок

Користувач ввів число, на екран вивести всі числа від введеного до 0.

n = parseInt(prompt('number:'));

res = "";

for (let i = n; i >=0; i--) {

    res += i + " ";

}

alert(res);

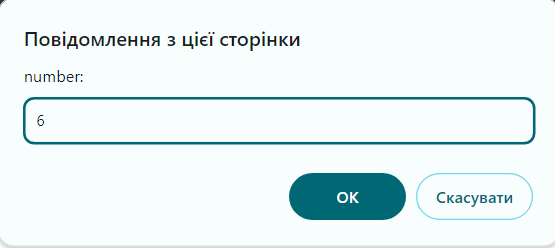
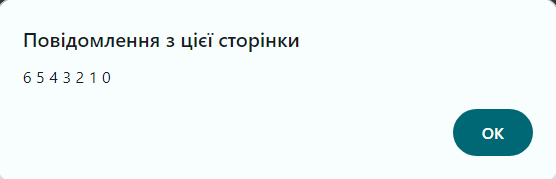
 

Рис. 2.4 Робота циклу

Запитати число та ступінь. Піднести число до зазначеного ступеня та вивести результат.

let n = parseInt(prompt('number:'));

let m = parseInt(prompt('power:'));

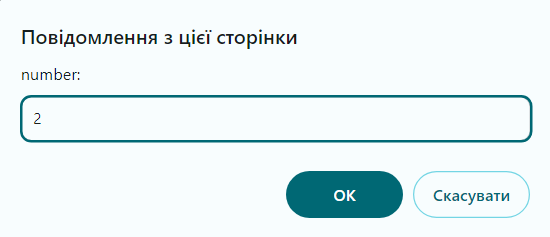
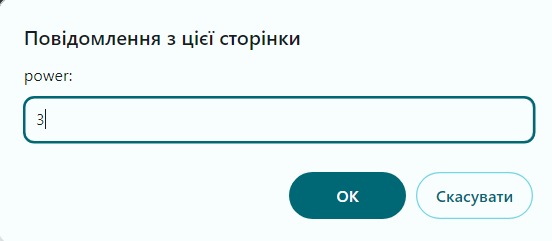
let res = 1;

for (let i = 0; i <m; i++) {

    res \*= n

}

alert(res);

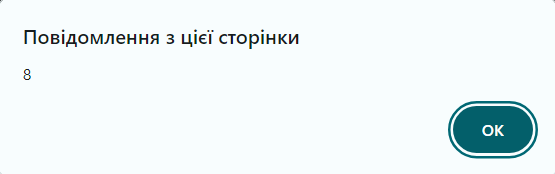


Рис. 2.5 Робота циклу

Користувач вводить числа з клавіатури. Знайти найбільше із введених чисел. Введення чисел закінчується тоді, коли буде введено два однакові числа поспіль.

let c;

n = null;

max = -Infinity;

res = ""

do {

    c = n;

    n = parseInt(prompt("n:"));

    if (n > max)

        max = n;

    res += n + " "

} while (n !== c)

alert(`string:${res}, max:${max}`);

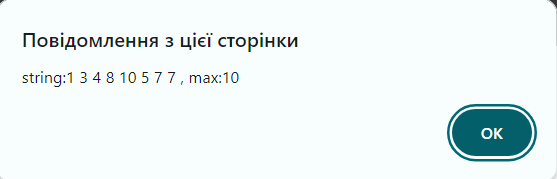


Рис. 2.6 Робота циклу

Зациклити виведення днів тижня таким чином: День тижня. Хочете побачити наступний день?» і так доти, доки користувач натискає OK.

let days = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"]

let ind = 0;

let cont = true;

while(cont) {

    cont = confirm(`Day: ${days[ind]}, continue?`);

    ind = (ind + 1) % days.length;

}

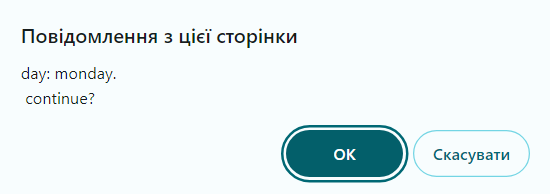


Рис. 2.7 Робота циклу

Вивести таблицю множення всіх чисел від 2 до 9. Кожне число необхідно помножити на числа від 1 до 10.

let table = ""

for(let i = 2; i <=9 ; i++){

    for(let j = 1; j <=10; j++){

        let mul = i \* j

        table += i + " \* " + j + " = " + mul + "\n"

    }

    table += "\n"

}

alert(table)

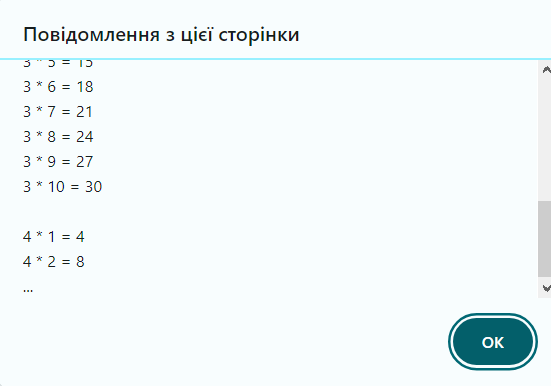


Рис. 2.8 Робота циклу

***Висновок:*** На цій лабораторній роботі ми вивчили відмінності у роботі операцій та операторів у мові JavaScript від інших мов; ознайомитись з базовими типами даних; вивчити особливості приведення типів, навчитися працювати з циклами.