Національний університет «Одеська політехніка»
Інститут комп'ютерних систем
Кафедра інформаційних систем

КУРСОВА РОБОТА

з дисципліни «Об'єктно-орієнтоване програмування»

Тема: «Розробка прогресивного калькулятора»

Студентка __2__ курсу AI-225 групи
Спеціальності 122 – «Комп’ютерні науки»
_____Майструк Н. М.
(прізвище та ініціали)
Керівник ст.викл, к.т.н. Годовіченко М.
_____
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____
Кількість балів: _____
Оцінка: ECTS _____

Члени комісії

_____ (підпис)	_____ (прізвище та ініціали)
_____ (підпис)	_____ (прізвище та ініціали)
_____ (підпис)	_____ (прізвище та ініціали)

Одеса – 2024

ЗМІСТ

ЗАВДАННЯ НА КУРСОВУ РОБОТУ 3
ВСТУП 5
1 ТЕОРЕТИЧНІ ВІДОМОСТІ ПРО КЛАСИ ТА ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ В JAVASCRIPT 6
1.1 Теоретичні відомості 6
2 ПРОГРАМНА РЕАЛІЗАЦІЯ ООП У ПРОЕКТІ 7
2.1 Програмна реалізація ООП 7
2.2 Опис структури проекту 8
3 ІНСТРУКЦІЯ КОРИСТУВАЧА 8
3.1 Інструкція користувача 9
ВИСНОВКИ 12
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 12

ЗАВДАННЯ НА КУРСОВУ РОБОТУ

- Розробка базових функцій калькулятора, таких як додавання, віднімання, множення та ділення.

- Підтримка введення десяткових чисел.

- Відображення результату на екрані калькулятора.

- Розробка дизайну інтерфейсу калькулятора, включаючи розташування кнопок, екран для відображення чисел та результатів.

- Забезпечення зручного та інтуїтивно зрозумілого інтерфейсу для користувача.

- Написання алгоритмів для виконання арифметичних операцій.

- Обробка виключних ситуацій, таких як ділення на нуль.

- Вибір відповідної архітектури та структури коду для калькулятора.

- Використання класів та об'єктів для організації коду (як в даному прикладі).

- Написання обробників подій для кнопок калькулятора.

- Забезпечення коректної обробки натискань на цифрові кнопки, оператори та кнопку "дорівнює".

- Додавання анімацій для покращення візуального сприйняття.

- Використання сучасних дизайнів для надання калькулятору привабливого вигляду.

АНОТАЦІЯ

Цей проект полягає у створенні простого, але функціонального веб-калькулятора, реалізованого за допомогою JavaScript, HTML та CSS. Головною метою було розробити інтуїтивно зрозумілий інтерфейс користувача, що дозволяє легко виконувати основні арифметичні операції: додавання, віднімання, множення та ділення.

Користувацький інтерфейс калькулятора розроблений з урахуванням сучасних дизайнерських тенденцій, включаючи використання градієнтів та плавних анімацій. Екран для відображення поточного введення та результатів обчислень забезпечує чітке й зрозуміле представлення інформації. Всі основні функції калькулятора, такі як введення чисел, вибір операцій та обчислення результатів, виконуються за допомогою JavaScript.

ABSTRACT

This project presents the development of a dynamic web-based calculator using JavaScript, HTML, and CSS. The main aim was to construct a user-friendly interface enabling users to perform basic arithmetic operations seamlessly, including addition, subtraction, multiplication, and division.

The calculator's interface incorporates modern design elements such as gradients and animations to enhance user experience. A designated display screen provides real-time feedback, ensuring clarity and accuracy throughout calculations. JavaScript is employed to manage all computation processes, facilitating efficient execution of operations.

ВСТУП

Ця курсова робота є результатом розробки веб-калькулятора, який є важливим інструментом для вирішення різноманітних завдань, пов'язаних з арифметикою, фінансами та іншими сферами життя. Основною метою проекту було створення інтуїтивного та легкого у використанні інтерфейсу, що дозволяє користувачам швидко та зручно виконувати різні арифметичні операції.

За допомогою JavaScript, HTML та CSS, було розроблено клас `Calculator`, який відповідає за всі аспекти роботи калькулятора. Цей клас забезпечує легке розширення та підтримку різних функцій, таких як додавання, віднімання, множення та ділення.

Особлива увага була приділена валідації введених даних та обробці можливих помилок, таких як спроба ділення на нуль або некоректні вхідні дані. Користувачеві надається інформативне повідомлення про помилку, яке допомагає зрозуміти, що сталося і як виправити ситуацію.

Цей проект є демонстрацією використання сучасних веб-технологій для створення інтерактивних інструментів, а також підкреслює важливість організації коду та валідації даних для забезпечення ефективної та надійної роботи програмного забезпечення.

1 ТЕОРЕТИЧНІ ВІДОМОСТІ ПРО КЛАСИ ТА ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ В JAVASCRIPT

1.1 Теоретичні відомості

У програмуванні, особливо в JavaScript, концепція класів та об'єктно-орієнтованого програмування (ООП) відіграють ключову роль у створенні структурованих та повторно використовуваних програмних рішень.

Припустимо, що ми розробляємо програму для управління автомобілями. Кожен автомобіль може мати свої властивості, такі як марка, модель, рік випуску та можливо стан палива. У зв'язку з цим, ми можемо створити клас "Car" (Автомобіль), який буде мати ці властивості та методи для їх обробки.

```
class Car {
  constructor(brand, model, year) {
    this.brand = brand;
    this.model = model;
    this.year = year;
    this.fuel = 100;
  }

  drive(distance) {
    const fuelConsumption = 10;
    const fuelNeeded = (distance / 100) * fuelConsumption;

    if (this.fuel >= fuelNeeded) {
      this.fuel -= fuelNeeded;
      console.log(`Подорожуючи ${distance} км, залишок палива: ${this.fuel}%`);
    } else {
      console.log(`Недостатньо пального для подорожі`);
    }
  }
  refuel(amount) {
    this.fuel += amount;
    console.log(`Бак поповнено. Загальний рівень палива: ${this.fuel}%`);
  }
}
const myCar = new Car("Toyota", "Camry", 2020);

myCar.drive(200);
myCar.refuel(50);
```

У цьому прикладі, клас "Car" визначає основні характеристики автомобіля та методи для управління ними. Кожен новий автомобіль, створений на основі цього класу, буде мати власні значення для марки, моделі та року випуску, а також зможе виконувати дії, такі як подорож і поповнення пального, за допомогою методів класу.

2 ПРОГРАМНА РЕАЛІЗАЦІЯ ООП У ПРОЕКТІ

2.1 Програмна реалізація ООП

У нашому проекті ми використали об'єктно-орієнтований підхід для реалізації класу `Calculator`, що відповідає за функціональність нашого веб-калькулятора. Цей клас містить різні методи для виконання арифметичних операцій та взаємодії з користувачем.

Один з головних принципів ООП, який ми використали, - це ідея інкапсуляції. Ми створили клас, який зберігає в собі стан (такий як поточні та попередні числа, результат операції, обрані оператори) і методи для роботи з цим станом. Наприклад, метод `setNum` призначений для зчитування введених користувачем чисел та їх збереження, метод `moveNum` відповідає за обробку обраних операторів, а метод `displayNum` реалізує логіку обчислення результату операції.

Ще одним важливим аспектом ООП, який ми використовуємо, є успадкування. Наш клас `Calculator` може бути розширений з іншими функціональними можливостями, наприклад, додавання нових арифметичних операцій чи покращення взаємодії з користувачем, без зміни базового коду.

Використовуючи ООП у нашому проекті, ми досягли чіткої структури програми, що сприяє зручності розробки, тестування та підтримки коду.

2.2 Опис структури проекту:

1. HTML-файл (index.html): Цей файл містить основну структуру веб-сторінки, включаючи розмітку для відображення калькулятора та інших елементів інтерфейсу.

2. CSS-файл (style.css): У цьому файлі знаходяться стилізаційні правила, які визначають зовнішній вигляд калькулятора і забезпечують його коректне відображення на різних пристроях і розмірах екрану.

3. JavaScript-файл (main.js): В цьому файлі містяться скрипти, які відповідають за логіку роботи калькулятора. Він реалізує клас `Calculator`, який забезпечує основний функціонал калькулятора, такий як додавання чисел, виконання арифметичних операцій та відображення результату.

Методи:

setNum(event): Ця функція встановлює поточне число, зчитуючи значення, яке вводить користувач, і відображаючи його на екрані калькулятора.

moveNum(event): Ця функція обробляє обрані оператори, зберігаючи поточне число і обраний оператор для подальшого використання при виконанні арифметичних операцій.

displayNum(): Ця функція виконує арифметичну операцію, засновану на обраних числах та операторі, і відображає результат на екрані калькулятора.

clearAll(): Ця функція очищає поточні значення, повертаючи калькулятор до початкового стану.

initialize(): Ця функція ініціалізує події для кнопок калькулятора, встановлюючи обробники подій для кліків на числа, оператори та кнопку "Дорівнює".

3 ІНСТРУКЦІЯ КОРИСТУВАЧА

3.1 Інструкція користувача

Після переходу на сайт користувач бачить перед собою калькулятор(рис. 3.1).

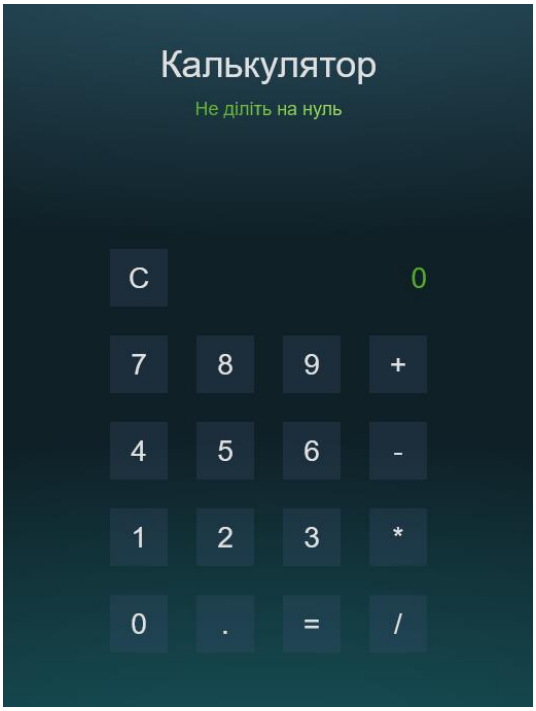

Рисунок 3.1 – Стартовий екран

Користувач може робити будь-які розрахунки: додавання,віднімання,множення,ділення(рис 3.2)

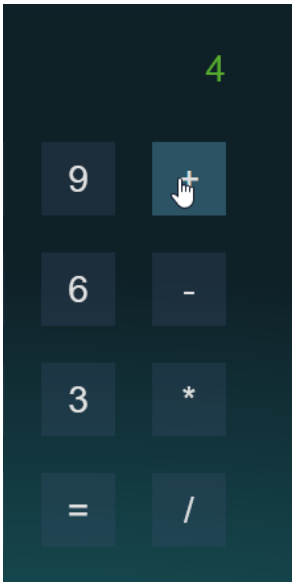

Рисунок 3.2 – Усі реалізовані дії

Кожен раз при натисканні на цифру чи дію вона відображається анімацією на екрані(рис 3.3)

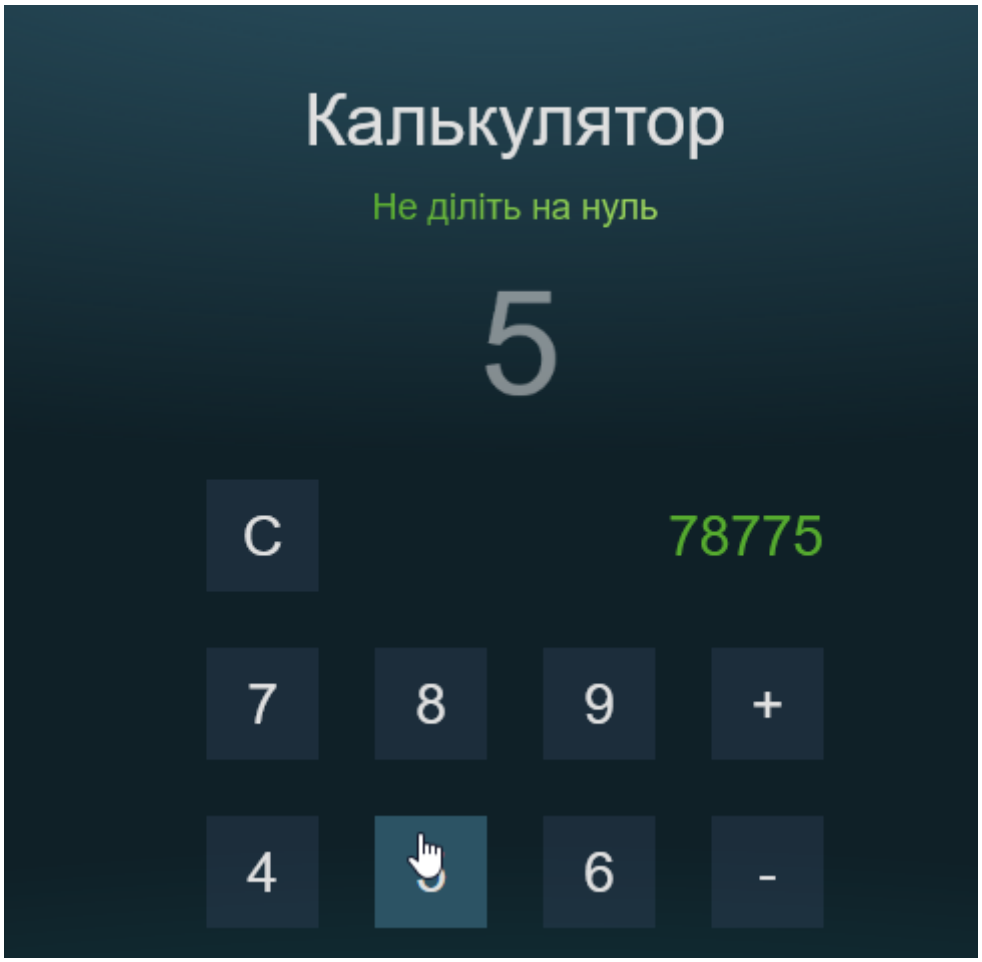

Рисунок 3.3 – Анімація дій

У калькуляторі реалізована валідації даних,тому якщо користувач буде робити неможливі дії,то ця помилка відобразиться(рис 3.4)

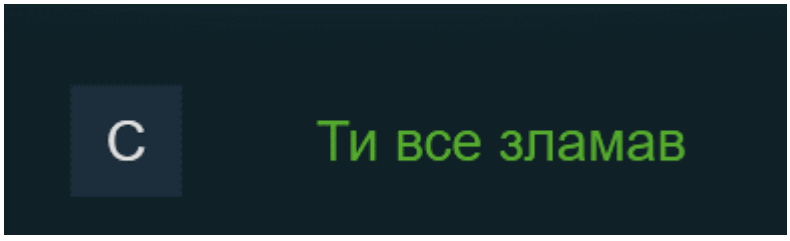

Рисунок 3.4 – Помилка дій

Також реалізована валідація ділення на нуль,у разі цього на вебсайті є невеличка пасхалка,яка припиняє роботу калькулятора і перезапускає його (рис 3.5)

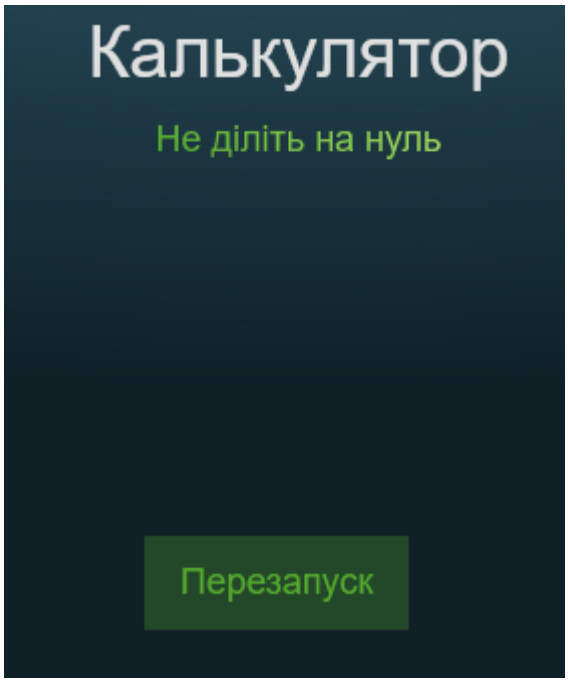

Рисунок 3.5 – Результат ділення на нуль

ВИСНОВКИ

У постановці завдання для цього проекту мої головні цілі були створити функціональний калькулятор за допомогою HTML, CSS та JavaScript, що був би легко зрозумілим та зручним у використанні. Я прагнула створити додаток, який міг би виконувати основні арифметичні операції та надавати користувачеві зручний інтерфейс для взаємодії.

Під час реалізації проекту я успішно створила функціонал калькулятора, який дозволяє вводити числа, обирати арифметичні операції та отримувати результати обчислень. Я використовувала об'єктно-орієнтований підхід у JavaScript, щоб структурувати код та зробити його більш зрозумілим та підтримуваним. Крім того, вдалося створити привабливий та інтуїтивно зрозумілий дизайн інтерфейсу.

У результаті цього проекту я отримала практичний досвід в розробці веб-додатків, покращила свої навички програмування на JavaScript та збагатила свій розуміння об'єктно-орієнтованого програмування. Я задоволена тим, що мої цілі були досягнуті, і проект відповідає вимогам, які я ставила перед собою.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

MDN Web Docs - https://developer.mozilla.org/en-US/
W3Schools - https://www.w3schools.com/
CSS-Tricks - https://css-tricks.com/
Smashing Magazine - https://www.smashingmagazine.com/
Dev.to - https://dev.to/
GitHub Discussions - https://github.com/discussions