ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ  
з дисципліни «Основи програмування на мові Javascript»  
студентки групи ПС-21-1  
Лисової Єлизавети Сергіївни  
кафедра комп’ютерних технологій, ДНУ  
2024/2025

1. Постановка задачі

Завдання 1  
На площині задано область. Напишіть програму, яка пропонує користувачеві "влучити" точкою в цю область, пропонуючи декілька спроб. Програма супроводжує спроби користувача коментарями: "тепліше", "холодніше", "нарешті ви влучили з N-ї спроби". Область на площині утворено за  
допомогою трьох заданих фігур таким чином: об'єднання перших двох фігур (фігури-1 і фігури-2) у перетині з третьою фігурою (фігурою-3).

Завдання 2  
Задано послідовність цілих чисел. Напишіть програму, яка за один (за можливості) перегляд послідовності, для елементів, що задовольняють заданому критерію, обчислює задану величину. Критерій і шукана величина визначені індивідуальним варіантом. Не використовуйте масиви, якщо в  
цьому немає потреби.

Завдання 3  
Напишіть програму, яка вводить послідовність цілих чисел a[0], a[1], ... , a[N-1] і упорядковує її згідно до варіанту завдання певним методом за певним критерієм у певному напрямку. Передбачте можливість файлового вводу і виводу даних за допомогою перенапрямлення вводу і виводу в командному рядку. Метод упорядкування реалізуйте у вигляді окремої функції (стандартні функції упорядкування використовувати не дозволяється).

2. Опис розв’язку

**Опис завдання**

Користувач повинен "влучити" точкою в задану область на площині, яка утворена об'єднанням двох вертикальних стрічок та перетином з горизонтальною стрічкою. Програма надає користувачеві декілька спроб та супроводжує кожну спробу коментарями: "тепліше", "холодніше", "нарешті ви влучили з N-ї спроби".

**Розв'язок**

1. **Визначення фігур**: Програма визначає три фігури: дві вертикальні стрічки та одну горизонтальну стрічку.
2. **Перевірка попадання**: Програма перевіряє, чи потрапляє точка в об'єднання двох вертикальних стрічок та перетин з горизонтальною стрічкою.
3. **Взаємодія з користувачем**: Програма запитує у користувача координати точки та надає відповідні коментарі про близькість до цілі.
4. **Виведення результату**: Після успішного попадання програма повідомляє, з якої спроби користувач влучив у ціль.

**Завдання 2: "Обчислення кількості найменших значень"**

**Опис завдання**

Користувач вводить послідовність цілих чисел. Програма обчислює кількість найменших значень серед чисел, сума власних дільників яких є парною.

**Розв'язок**

1. **Введення даних**: Програма запитує у користувача послідовність цілих чисел.
2. **Обчислення суми дільників**: Для кожного числа обчислюється сума його власних дільників.
3. **Перевірка парності**: Перевіряється, чи сума дільників є парною.
4. **Знаходження найменшого значення**: Серед чисел з парною сумою дільників знаходиться найменше значення.
5. **Обчислення кількості**: Обчислюється кількість чисел, які дорівнюють найменшому значенню.
6. **Виведення результату**: Програма виводить кількість найменших значень серед чисел з парною сумою дільників.

**Завдання 3: "Сортування послідовності за кількістю цифр"**

**Опис завдання**

Користувач вводить послідовність цілих чисел. Програма сортує послідовність за кількістю цифр у числі, використовуючи метод Selection Sort. Програма також підтримує файловий ввід та вивід даних.

**Розв'язок**

1. **Введення даних**: Програма запитує у користувача послідовність цілих чисел або зчитує її з файлу.
2. **Обчислення кількості цифр**: Для кожного числа обчислюється кількість цифр.
3. **Сортування методом Selection Sort**: Послідовність сортується за кількістю цифр у числі за допомогою методу Selection Sort.
4. **Виведення результату**: Відсортована послідовність виводиться на екран або записується у файл.

**Загальні аспекти для всіх завдань**

1. **Взаємодія з користувачем**: Усі програми взаємодіють з користувачем через консоль, запитуючи введення даних та виводячи результати.
2. **Файловий ввід/вивід**: Програми підтримують можливість читання даних з файлу та запису результатів у файл за допомогою перенапрямлення вводу та виводу в командному рядку.
3. **Реалізація власних алгоритмів**: Для кожного завдання реалізовано власні алгоритми, такі як перевірка попадання точки, обчислення суми дільників, сортування методом Selection Sort, що дозволяє уникати використання стандартних функцій упорядкування.

3. Вихідний текст програми розв’язку задачі (основні фрагменти з коментарями)

const readline = require('readline');

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout

});

// Визначення фігур

const figure1 = { x1: 1, x2: 3, y1: -Infinity, y2: Infinity };

const figure2 = { x1: 4, x2: 6, y1: -Infinity, y2: Infinity };

const figure3 = { x1: -Infinity, x2: Infinity, y1: 2, y2: 4 };

let attempts = 0;

function isPointInFigure(point, figure) {

return point.x >= figure.x1 && point.x <= figure.x2 && point.y >= figure.y1 && point.y <= figure.y2;

}

function isPointInTargetArea(point) {

const inFigure1Or2 = isPointInFigure(point, figure1) || isPointInFigure(point, figure2);

const inFigure3 = isPointInFigure(point, figure3);

return inFigure1Or2 && inFigure3;

}

function getUserInput() {

rl.question('Введіть координати точки (x, y): ', (input) => {

const [x, y] = input.split(',').map(Number);

const point = { x, y };

attempts++;

if (isPointInTargetArea(point)) {

console.log(`Нарешті ви влучили з ${attempts}-ї спроби!`);

rl.close();

} else {

const distance = Math.abs(point.x - 2.5) + Math.abs(point.y - 3);

if (distance < 1) {

console.log('Тепліше');

} else if (distance > 2) {

console.log('Холодніше');

} else {

console.log('Тепліше, але не дуже');

}

getUserInput();

}

});

}

console.log('Спробуйте влучити точкою в область, утворену об\'єднанням двох вертикальних стрічок та перетином з горизонтальною стрічкою.');

getUserInput();

const readline = require('readline');

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout

});

function sumOfDivisors(n) {

let sum = 0;

for (let i = 1; i <= Math.floor(n / 2); i++) {

if (n % i === 0) {

sum += i;

}

}

return sum;

}

function isEven(n) {

return n % 2 === 0;

}

function processSequence(sequence) {

let minValue = Infinity;

let count = 0;

let validNumbers = [];

for (let num of sequence) {

const sumDivisors = sumOfDivisors(num);

if (isEven(sumDivisors)) {

validNumbers.push(num);

if (num < minValue) {

minValue = num;

}

}

}

for (let num of validNumbers) {

if (num === minValue) {

count++;

}

}

return count;

}

function getUserInput() {

rl.question('Введіть послідовність цілих чисел, розділених пробілами: ', (input) => {

const sequence = input.split(' ').map(Number);

const result = processSequence(sequence);

console.log(`Кількість найменших значень серед чисел з парною сумою власних дільників: ${result}`);

rl.close();

});

}

console.log('Програма обчислює кількість найменших значень серед чисел з парною сумою власних дільників.');

getUserInput();

const readline = require('readline');

const fs = require('fs');

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout

});

function countDigits(n) {

return Math.abs(n).toString().length;

}

function selectionSort(arr) {

for (let i = 0; i < arr.length - 1; i++) {

let minIndex = i;

for (let j = i + 1; j < arr.length; j++) {

if (countDigits(arr[j]) < countDigits(arr[minIndex])) {

minIndex = j;

}

}

if (minIndex !== i) {

[arr[i], arr[minIndex]] = [arr[minIndex], arr[i]];

}

}

return arr;

}

function getUserInput() {

rl.question('Введіть кількість елементів N: ', (N) => {

N = parseInt(N);

if (isNaN(N) || N <= 0) {

console.log('Некоректне значення N.');

rl.close();

return;

}

let sequence = [];

let count = 0;

function askNextNumber() {

if (count < N) {

rl.question(`a[${count + 1}] = ? `, (num) => {

num = parseInt(num);

if (!isNaN(num)) {

sequence.push(num);

count++;

askNextNumber();

} else {

console.log('Некоректне значення. Спробуйте ще раз.');

askNextNumber();

}

});

} else {

const sortedSequence = selectionSort(sequence);

console.log('Упорядкована послідовність:');

console.log(sortedSequence.join(' '));

rl.close();

}

}

askNextNumber();

});

}

// Перевірка, чи програма запущена з файлу

if (process.argv[2] === '--file') {

const inputFile = process.argv[3];

const outputFile = process.argv[4];

fs.readFile(inputFile, 'utf8', (err, data) => {

if (err) {

console.error(`Помилка читання файлу: ${err.message}`);

return;

}

const lines = data.trim().split('\n');

const N = parseInt(lines[0]);

const sequence = lines.slice(1).map(Number);

const sortedSequence = selectionSort(sequence);

const outputData = sortedSequence.join(' ');

fs.writeFile(outputFile, outputData, (err) => {

if (err) {

console.error(`Помилка запису файлу: ${err.message}`);

return;

}

console.log(`Упорядкована послідовність записана у файл ${outputFile}`);

});

});

} else {

console.log('Програма сортує послідовність цілих чисел за кількістю цифр у числі, використовуючи метод Selection Sort.');

getUserInput();

}

4. Опис інтерфейсу програми (керівництво користувача)

**Завдання 1: "Влучити точкою в область"**

**Опис програми**

Ця програма пропонує користувачеві "влучити" точкою в задану область на площині, яка утворена об'єднанням двох вертикальних стрічок та перетином з горизонтальною стрічкою. Програма надає користувачеві декілька спроб та супроводжує кожну спробу коментарями: "тепліше", "холодніше", "нарешті ви влучили з N-ї спроби".

**Інтерфейс програми**

1. **Запуск програми**:
   * Відкрийте термінал або командний рядок.
   * Перейдіть до директорії, де знаходиться файл програми.
   * Запустіть програму командою:

bash

Copy

node hitTheTarget.js

1. **Введення координат**:
   * Програма запропонує ввести координати точки у форматі x, y.
   * Наприклад, якщо ви хочете ввести точку з координатами (2, 3), введіть:

Copy

2, 3

1. **Результат**:
   * Після кожної спроби програма надасть відповідний коментар: "тепліше", "холодніше" або "нарешті ви влучили з N-ї спроби".
   * Після успішного попадання програма завершить роботу.

**Завдання 2: "Обчислення кількості найменших значень"**

**Опис програми**

Ця програма обчислює кількість найменших значень серед чисел, сума власних дільників яких є парною.

**Інтерфейс програми**

1. **Запуск програми**:
   * Відкрийте термінал або командний рядок.
   * Перейдіть до директорії, де знаходиться файл програми.
   * Запустіть програму командою:

bash

Copy

node countMinValues.js

1. **Введення даних**:
   * Програма запропонує ввести кількість елементів N.
   * Наприклад, якщо ви хочете ввести 5 елементів, введіть:

Copy

5

* + Далі програма запропонує ввести кожне число послідовності.
  + Наприклад, якщо ви хочете ввести числа 15, -2, 8, 10, 7, введіть:

Copy

a[1] = ? 15

a[2] = ? -2

a[3] = ? 8

a[4] = ? 10

a[5] = ? 7

1. **Результат**:
   * Програма виведе кількість найменших значень серед чисел з парною сумою дільників.

**Завдання 3: "Сортування послідовності за кількістю цифр"**

**Опис програми**

Ця програма сортує послідовність цілих чисел за кількістю цифр у числі, використовуючи метод Selection Sort. Програма також підтримує файловий ввід та вивід даних.

**Інтерфейс програми**

1. **Запуск програми**:
   * Відкрийте термінал або командний рядок.
   * Перейдіть до директорії, де знаходиться файл програми.
   * Запустіть програму командою:

bash

Copy

node sortByDigits.js

1. **Введення даних**:
   * Програма запропонує ввести кількість елементів N.
   * Наприклад, якщо ви хочете ввести 5 елементів, введіть:

Copy

5

* + Далі програма запропонує ввести кожне число послідовності.
  + Наприклад, якщо ви хочете ввести числа 15, -2, 8, 10, 7, введіть:

Copy

a[1] = ? 15

a[2] = ? -2

a[3] = ? 8

a[4] = ? 10

a[5] = ? 7

1. **Результат**:
   * Програма виведе відсортовану послідовність за кількістю цифр у числі.

**Файловий ввід/вивід**

1. **Запуск програми з файлами**:
   * Відкрийте термінал або командний рядок.
   * Перейдіть до директорії, де знаходиться файл програми.
   * Запустіть програму з параметрами для файлового вводу та виводу:

bash

Copy

node sortByDigits.js --file input.txt output.txt

* + Де input.txt — файл з вхідними даними, а output.txt — файл для вихідних даних.

1. **Формат файлу вводу**:
   * Перший рядок файлу повинен містити кількість елементів N.
   * Наступні рядки повинні містити елементи послідовності, по одному числу в рядку.
2. **Формат файлу виводу**:
   * Файл буде містити відсортовану послідовність, розділену пробілами.

5. Опис тестових прикладів

**Завдання 1: "Влучити точкою в область"**

**Тестовий приклад 1**

* **Вхідні дані**:
  + Координати точки: (2, 3)
* **Очікуваний результат**:
  + Коментар: "Нарешті ви влучили з 1-ї спроби!"

**Тестовий приклад 2**

* **Вхідні дані**:
  + Координати точки: (5, 5)
  + Координати точки: (3, 3)
* **Очікуваний результат**:
  + Коментар: "Холодніше"
  + Коментар: "Нарешті ви влучили з 2-ї спроби!"

**Тестовий приклад 3**

* **Вхідні дані**:
  + Координати точки: (1, 1)
  + Координати точки: (2, 2)
  + Координати точки: (2, 3)
* **Очікуваний результат**:
  + Коментар: "Холодніше"
  + Коментар: "Тепліше"
  + Коментар: "Нарешті ви влучили з 3-ї спроби!"

**Завдання 2: "Обчислення кількості найменших значень"**

**Тестовий приклад 1**

* **Вхідні дані**:
  + Кількість елементів: 5
  + Послідовність: 15, -2, 8, 10, 7
* **Очікуваний результат**:
  + Кількість найменших значень: 1

**Тестовий приклад 2**

* **Вхідні дані**:
  + Кількість елементів: 6
  + Послідовність: 22, 10, 4, 6, 8, 12
* **Очікуваний результат**:
  + Кількість найменших значень: 2

**Тестовий приклад 3**

* **Вхідні дані**:
  + Кількість елементів: 4
  + Послідовність: 1, 2, 3, 4
* **Очікуваний результат**:
  + Кількість найменших значень: 1

**Завдання 3: "Сортування послідовності за кількістю цифр"**

**Тестовий приклад 1**

* **Вхідні дані**:
  + Кількість елементів: 5
  + Послідовність: 15, -2, 8, 10, 7
* **Очікуваний результат**:
  + Відсортована послідовність: -2 7 8 10 15

**Тестовий приклад 2**

* **Вхідні дані**:
  + Кількість елементів: 6
  + Послідовність: 22, 10, 4, 6, 8, 12
* **Очікуваний результат**:
  + Відсортована послідовність: 4 6 8 10 12 22

**Тестовий приклад 3**

* **Вхідні дані**:
  + Кількість елементів: 4
  + Послідовність: 1, 2, 3, 4
* **Очікуваний результат**:
  + Відсортована послідовність: 1 2 3 4

**Файловий ввід/вивід для завдання 3**

**Тестовий приклад 1**

* **Вхідні дані**:
  + Файл input.txt:

Copy

5

15

-2

8

10

7

* **Очікуваний результат**:
  + Файл output.txt:

Copy

-2 7 8 10 15

**Тестовий приклад 2**

* **Вхідні дані**:
  + Файл input.txt:

Copy

6

22

10

4

6

8

12

* **Очікуваний результат**:
  + Файл output.txt:

Copy

4 6 8 10 12 22

**Тестовий приклад 3**

* **Вхідні дані**:
  + Файл input.txt:

Copy

4

1

2

3

4

* **Очікуваний результат**:
  + Файл output.txt:

Copy

1 2 3 4

6. Аналіз помилок (опис усунення зауважень)

-