Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

**Лабораторна робота №1**
з дисципліни «Основи розроблення програмного забезпечення на платформі Node.js»
Тема: «Знайомство з середовищем Node.js»

**Виконали:**
студенти групи ІТ-04
Коновальчук Андрій Володимирович,
Полтава Віолетта Віталіївна,
Яцентій Богдан Богданович

**Перевірив:**
викладач кафедри ІСТ
Омельченко Віталій Вікторович

Київ 2023

**Завдання**

**Хід роботи:**

1. Встановити Node.js з офіційного сайту - https://nodejs.org/uk/
2. Встановіть IDE/редактор коду з підтримкою Node.js/JavaScript:
    a. WebStorm - https://www.jetbrains.com/webstorm/
    b. Visual Studio Code - https://code.visualstudio.com/
    c. Atom, Brackets або будь-який інший.
3. Ознайомитись з базовими типами та конструкціями мови JavaScript (за допомогою посилань або інших джерел).
4. Обрати на https://www.codewars.com/dashboard або https://leetcode.com/ 3 або більше задачі рівня Easy/Medium та вирішити їх використовуючи середовище Node.js.
5. Дайте відповіді на контрольні питання.

**Звіт має включати:**

1. Репозиторій з описом проблеми та кодом вирішення

**Контрольні питання:**

1. Чи є різниця між виконанням JavaScript в браузері та в середовищі Node.js?
2. Назвіть основні типи в JavaScript.
3. Як працює замикання(closure) в JavaScript?
4. Назвіть основні стандартні бібліотеки Node.js.
5. Які є способи імпортувати модулі в Node.js?
6. Як пов'язані Google Chrome / Chromium та Node.js?*
7. Як можна дозволити імпортувати змінні з поточного модуля?

**Посилання:**

1. https://w3schoolsua.github.io/nodejs/index.html#gsc.tab=0
2. https://developer.mozilla.org/en-US/docs/Learn/JavaScript
3. https://nodejs.dev/en/learn/
4. https://www.youtube.com/c/MaksymRudnyi

**Хід роботи:**

1. Коновальчук Андрій:
   https://www.codewars.com/kata/521c2db8ddc89b9b7a0000c1
   Дано масив n x n, поверніть елементи масиву, розташовані від крайніх елементів до середнього елемента, рухаючись за годинниковою стрілкою.
   Ідея полягає не в сортуванні елементів від найменшого значення до найвищого; ідея полягає в тому, щоб обійти 2d-масив за годинниковою стрілкою за шаблоном мушлі равлика.
   0x0 (порожня матриця) представляється як порожній масив всередині масиву [[]].

```
const snailSort = (array) => {
  // inner square, next circle of spiral
  const snailCore = array
    .slice(1, array.length - 1)
    .map((row) => row.slice(1, row.length - 1));
  const snailShell = [
    // first row
    array[0],
    // right column
    array.slice(1, array.length - 1).map((row) => row[row.length
- 1]),
    // last row
    array.length > 1 ? array[array.length - 1].reverse() : [],
    // left column
    array
      .slice(1, array.length - 1)
      .reverse()
      .map((row) => row[0]),
    // recursion with inner square of array (next circle of
spiral)
    snailCore.length > 0 ? snailSort(snailCore) : [],
  ].flat();
  return snailShell;
};
```

   Функція працює наступним чином: ми дістаємо “внутрішній квадрат” з матриці, потім збираємо по порядку перший рядок, правий стовпчик, нижній рядок, лівий стовпчик, а потім рекурсивно визиваємо цю функцію з “внутрішнім квадратом”.

2. Полтава Віолетта:
   https://www.codewars.com/kata/54da5a58ea159efa38000836/javascript

Задача полягає в знаходженні непарної к-сті певного числа у масиві. Завжди буде лише одне ціле число, яке з'являється непарним числом разів. Наприклад, [0,1,0,1,0] має повернути 0, оскільки він зустрічається 3 рази.

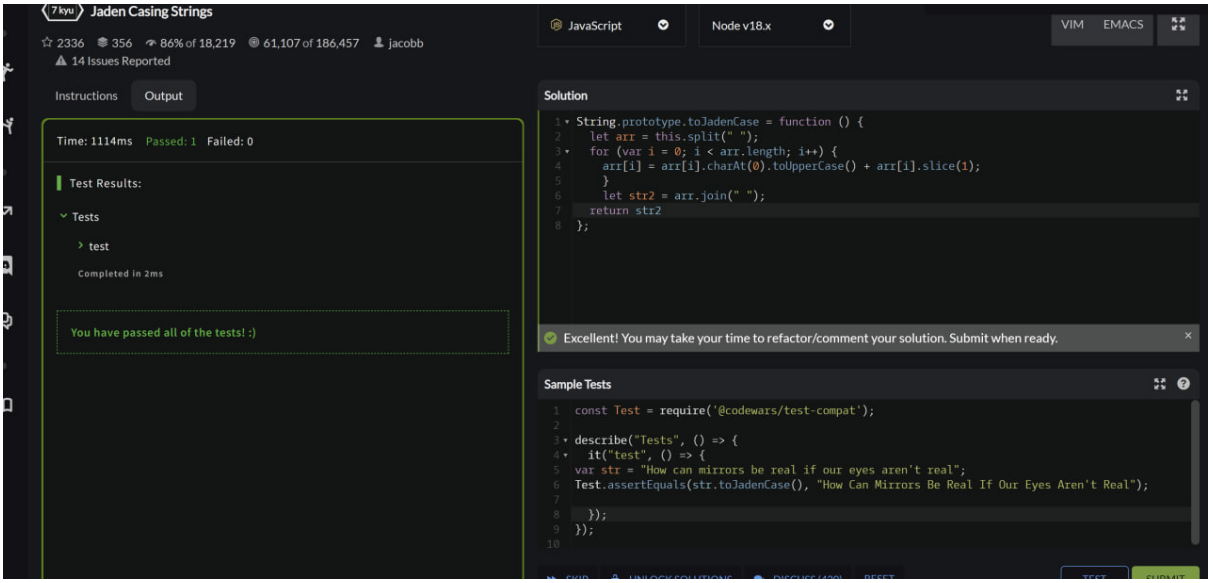
```
function findOdd(A) {
    let count = 0;
    let arr = A.sort((a, b) => a - b);
    for (let i = 0; i < arr.length; i++) {
        for (let j = 0; j < arr.length; j++) {
            if (arr[i] == arr[j]) {
                count++;
            }
        }
        if (count % 2 !== 0) {
            return arr[i];
        }
    }
}
let arr1 = [20, 1, 1, 2, 2, 3, 3, 5, 5, 4, 20, 4, 5];
let res1 = findOdd(arr1);
console.log("The odd is: " + res1); // 5
```

Як працює: нам треба повернути число, яке є непарним у к-сті. Використаємо два цикли, вони будуть порівнювати елементи і у разі знаходження пари повернуть "true". Якщо парних елементів буде три і вони повернуть "true". Оскільки маємо три "true", які повертаються для числа 5, яке є непарним в цьому масиві.

3. Бодя

https://www.codewars.com/kata/5390bac347d09b7da40006f6/javascript

Задача полягає в написанні функції, яке повертає стрінг, кожне слово якого буде написано з великої букви




Спочатку ми розбиваємо стрінг на масив, потім розподіляємо його завдяки “ “. Далі проходимся по кожному елементу масива, роблячи перший символ кожного слова аппер-кейсом додаючи потім слова з обрізаним першим сисволом. Далі приєднуємо всі слова в 1 і повертаємо.

**Контрольні питання:**

1. Node.js не має DOM, немає деяких об’єктів (“window”, “location”, “document”) та UI, має відмінності у runtime. Проте має доступ до файлової системи та потоків.
   Обидва мають один потік для запуску JavaScript. Обидва використовують Event Loop. Обидва неблокуються. Обидва мають можливості синхронізації та асинхронізації. Обидва використовують винятки, потік і область видимості однаково.
   JS – мова для написання веб-скриптів, і тому достатньо завантажити файл JS у документ HTML, а потім дозволити веб-переглядачу отримати вихідний код і, зрештою, запустити JavaScript. Node.js можна запускати безпосередньо через командний рядок або термінал, запускаючи файл командою “node [file_path.js]”.

2. JavaScript має 8 типів даних: string, number, bigint, boolean, undefined, null, symbol, object.
3. В JS використовується лексична область видимості. Це означає, що функції виконуються із застосуванням області видимості змінних, яка діяла, коли вони були визначені, а не коли визвані. Для реалізації лексичної області видимості внутрішній стан об'єкту функції JavaScript повинен включати не тільки код функції, а також посилання на область видимості, в якому знаходиться певна функція. Таке поєднання об'єкта функції і області видимості (набір прив'язок змінних), в якій розпізнається змінна функції, в комп'ютерній літературі називається замиканням (closure). Формально всі функції в JavaScript є замиканнями.
4. fs – для обробки файлової системи
   net – для створення серверів і клієнтів
   http (https) – для того щоб Node.js поводився як HTTP-сервер
   path – для обробки шляхів до файлів
   url – для парсингу рядків URL
   events – для обробки подій
   process – надає інформацію про поточний процес Node.js і контролює його
   os – надає інформацію про операційну систему
5. Головною перевагою модульності є можливість повторного використання. Як імпортувати модуль Node JS: використовуйте ту саму техніку, щоб імпортувати наші модулі або існуючі модулі Node JS. Платформа Node JS надала виклик функції "require()" для імпорту одного модуля в інший. Синтаксис: var someName = require("module-name");
   Тут «назва-модуля» – це необхідне ім'я модуля Node JS. "some-name" є посиланням на цей модуль. Виклик require() імпортує вказаний модуль і кешується в програмі, щоб нам не потрібно було імпортувати його знову і знову.
6. Node.js – це середовище виконання JS, засноване на двигуні JavaScript від Chrome під назвою V8. Простіше кажучи, двигун V8 JS було витягнуто з Хрому, і за його допомогою створено нову технологію для автономної роботи – Node.js.
7. Ключове слово "require" стосується функції, яка використовується для імпорту всіх змінних і функцій, експортованих за допомогою

об’єкта module.exports з іншого модуля. Якщо коротко, якщо файл хоче щось імпортувати, він має оголосити це за допомогою такого синтаксису: require('idOrPathOfModule');
Експортуючи щось із модуля, ми можемо використовувати будь-який дійсний ідентифікатор. Не обов’язково вказувати точне ім’я змінної/функції як ключ властивості, доданої до об’єкта module.exports. Просто треба переконатись, що використовується той самий ключ для доступу до того, що використовується під час експорту.

**Посилання на репозиторій:** https://github.com/m4cy43/NodeJs-Labs