МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних технологій

Звіт

Модульна контрольна робота №1

З дисципліни: «Кроссплатформенне програмування»

Варіант 1

Виконав:

здобувач групи АД-222

Буберенко С.О.

Перевірив:

Рудніченко М.М.

Оцінка\_\_\_\_\_\_\_\_\_\_\_\_

Одеса 2023

Теоретична частина

1. Поняття і призначення DOM, циклу подій, замикань.
2. Результати яких типів вийдуть в Javascript, якщо число скласти з рядком і навпаки, відняти від рядка і навпаки, поділити на рядок і навпаки?
3. Які Ви знаєте способи переадресувати сторінку в Javascript?

Відповіді:

1. DOM — об'єктна модель документа, яка дозволяє взаємодіяти з HTML-структурою сторінки через JavaScript.

Цикл подій — механізм, що дозволяє JavaScript виконувати асинхронні операції в однопотоковому середовищі.

Замикання — функція разом з її лексичним оточенням, що зберігає доступ до змінних навіть після виходу з контексту.

2.

- Додавання числа до рядка перетворює число на рядок і об'єднує їх.

- Віднімання числа від рядка викликає спробу перетворення рядка на число. Якщо це неможливо, результат — NaN.

- Ділення числа на рядок аналогічно викликає спробу перетворення рядка на число; якщо це неможливо — NaN.

3. Способи переадресації сторінки:

- window.location.href

- window.location.replace

- window.location.assign

- window.history.pushState або replaceState

Практична частина

1. Реалізуйте функцію isPrime(), яка повертає true або false, вказуючи, чи є передане їй число простим.
2. Реалізуйте функцію factorial(), яка повертає факторіал переданого їй числа.
3. Реалізуйте функцію fib(), що повертає n-ну кількість Фібоначчі
4. Реалізуйте функцію isSorted(), яка повертає true або false залежно від того, відсортований чи переданий їй числовий масив.
5. Створіть власну реалізацію функції filter(). Приклад виклику: filter ([1, 2, 3, 4], n => n <3) // [1, 2]
6. Створіть власну реалізацію функції reduce(). Приклад виклику: reduce ([1, 2, 3, 4], (a, b) => a + b, 0) // 10
7. Реалізуйте функцію reverse(), яка звертає порядок проходження символів переданої їй рядки. Не використовуйте вбудованою функцією reverse(). Приклад виклику: reverse ( 'abcdef') // 'fedcba'
8. Створіть власну реалізацію функції indexOf() для масивів. Приклади виклику: indexOf ([1, 2, 3], 1) // 0, indexOf ([1, 2, 3], 4) // -1
9. Реалізуйте функцію isPalindrome(), яка повертає true або false залежно від того, чи є передана їй рядок паліндромом (функція нечутлива до регістру і до наявності в рядку пробілів).
10. Реалізуйте функцію missing(), яка приймає невідсортоване масив унікальних чисел (числа в ньому не повторюються) від 1 до якогось числа n, і повертає число, якого немає у послідовності. Може бути або одне відсутнє число, або їх може не бути зовсім.
11. Реалізуйте функцію isBalanced() яка приймає рядок і повертає true або false, вказуючи на те, збалансовані чи фігурні дужки, що знаходяться в рядку. Приклади виклику: isBalanced ( '} {') // false, isBalanced ( '{} {}') // true
12. Реалізуйте функцію permute (), яка повертає масив рядків, що містить всі пермутації заданої рядки. Приклади виклику: permute( '') // [], permute ( 'abc') // [ 'abc', 'acb', 'bac', 'bca', 'cab', 'cba']
13. Створіть самостійну реалізацію функції debounce(). Реалізуйте клас LinkedList, не використовуючи вбудовані масиви JavaScript ([]). Ваш LinkedListдолжен підтримувати лише 2 методу: add() і has().
14. Реалізуйте клас HashMap, не використовуючи вбудовані об'єкти JavaScript({}) або функцію map(). Ваша реалізація HashMap повинна підтримувати лише 2 методу: get() та set().
15. Реалізуйте клас BinarySearchTree. Він повинен підтримувати 4 методу: add(), has(), remove(), і size().

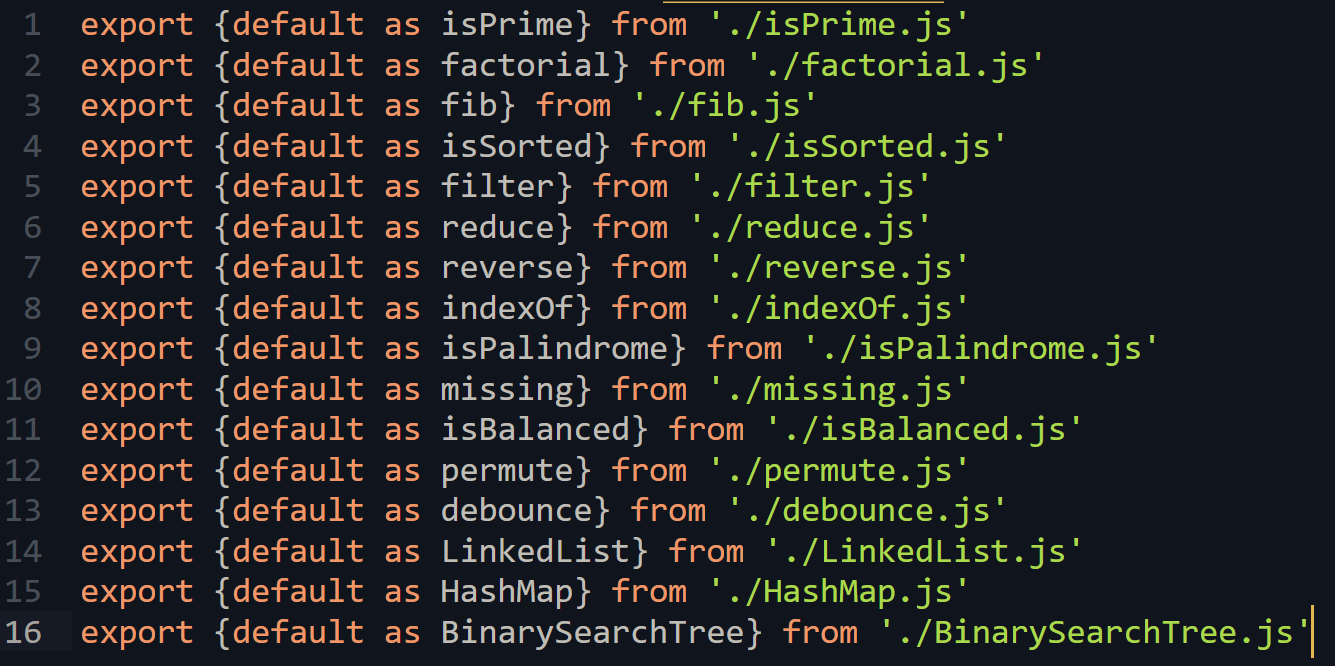


Рисунок 1. Імпорт експорт всіх функцій в одному файлі



Рисунок 2. Код функції isPrime, виклилк та результат.

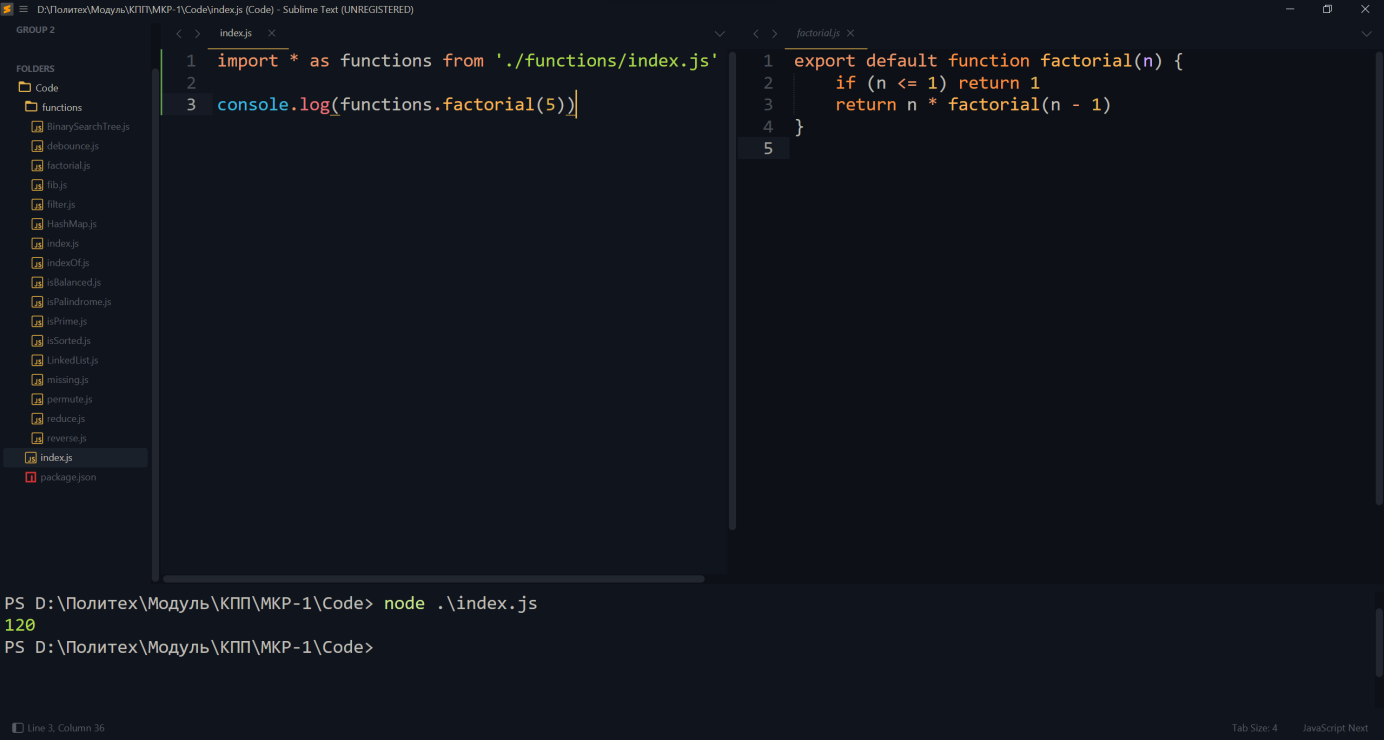


Рисунок 3. Код функції factorial, виклик та результат.

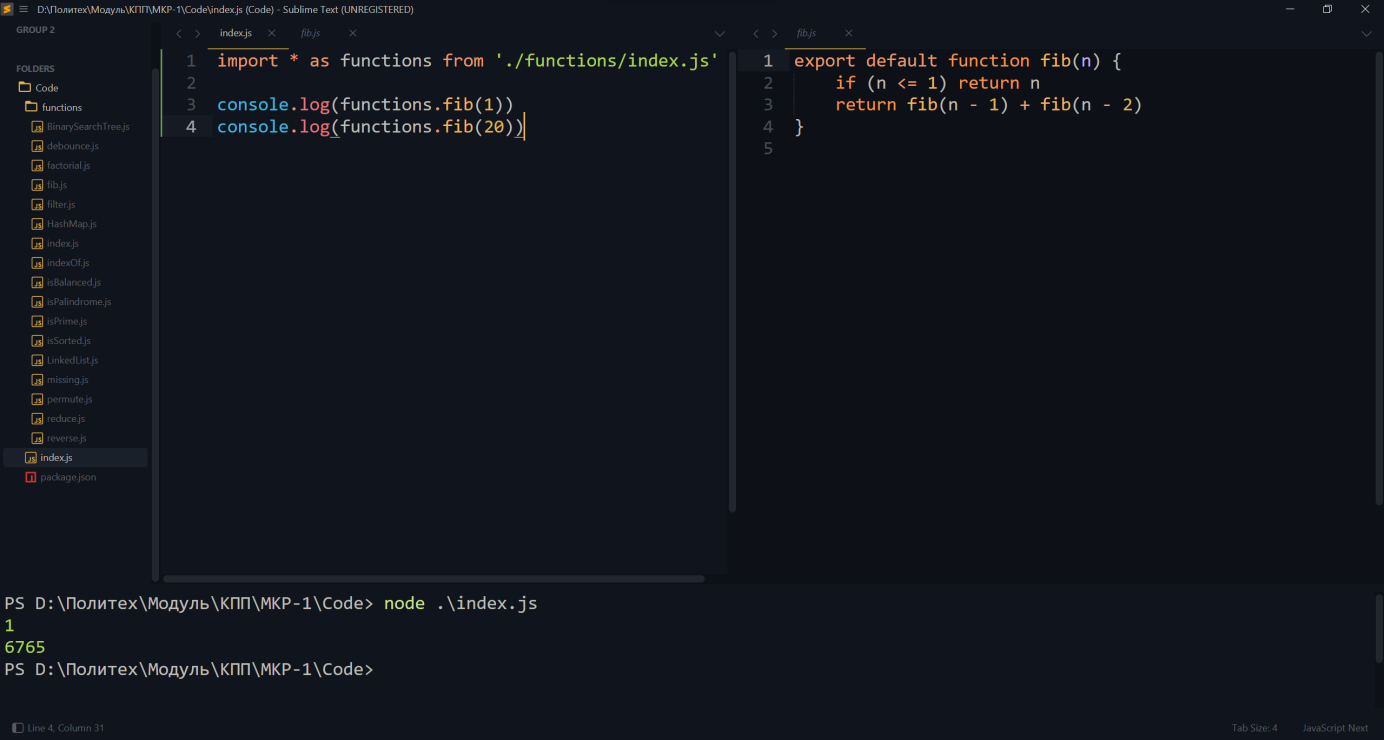


Рисунок 4. Код функції fib, виклик та результат.

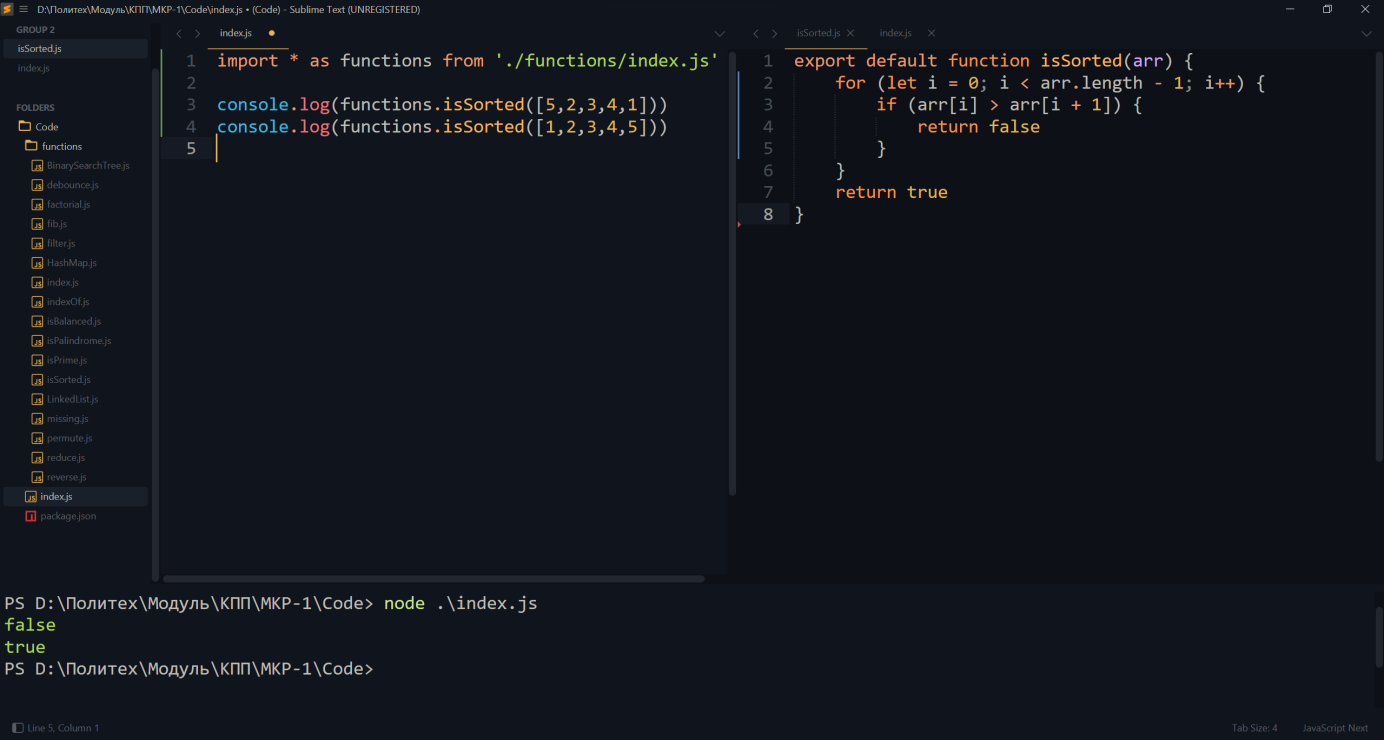


Рисунок 5. Код функції isSorted, виклик та результат.



Рисунок 6. Код функції filter, виклик та результат.



Рисунок 7. Код функції reduce, виклик та результат.



Рисунок 8. Код функції reverse, виклик та результат.



Рисунок 9. Код функції indexOf, виклик та результат.

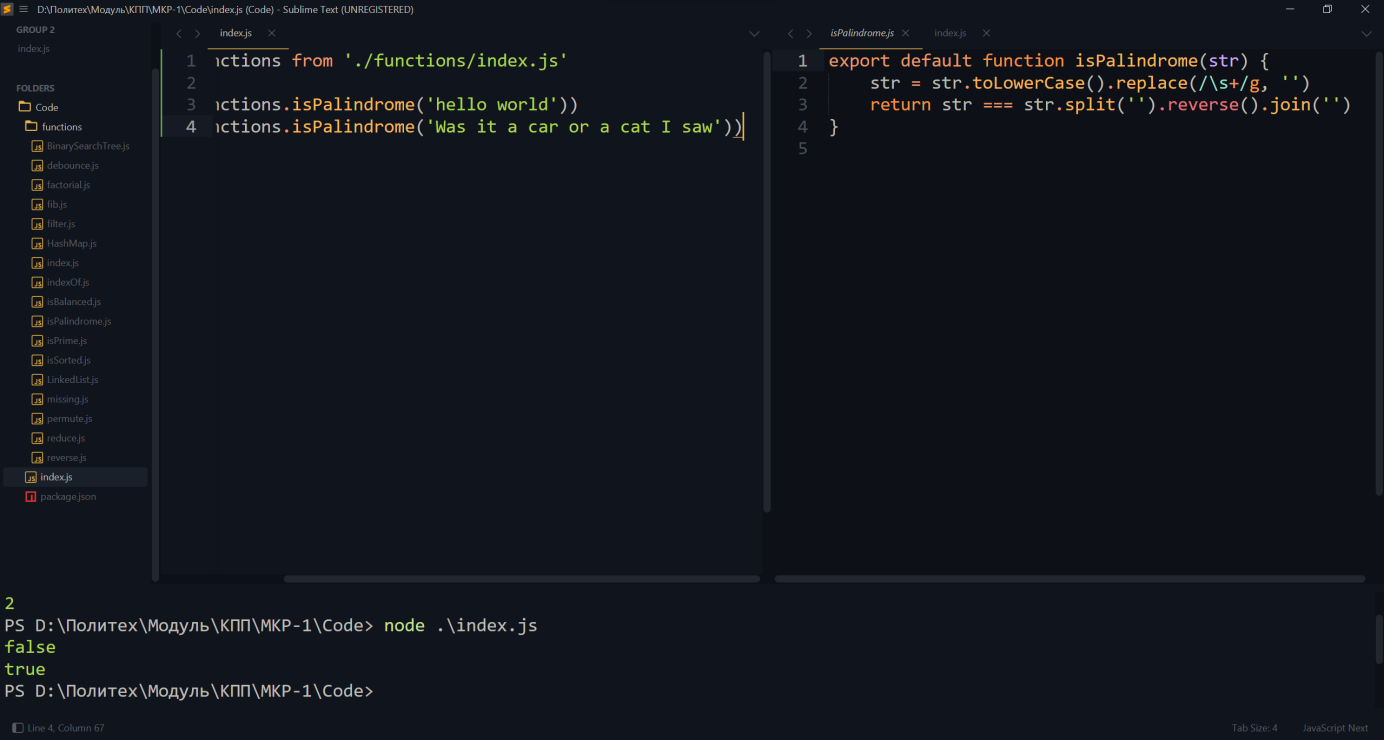


Рисунок 10. Код функції isPalindrome, виклик та результат.



Рисунок 11. Код функції missing, виклик та результат.

Рисунок 12. Код функції isBalanced, виклик та результат.



Рисунок 13. Код функції permute, виклик та результат.

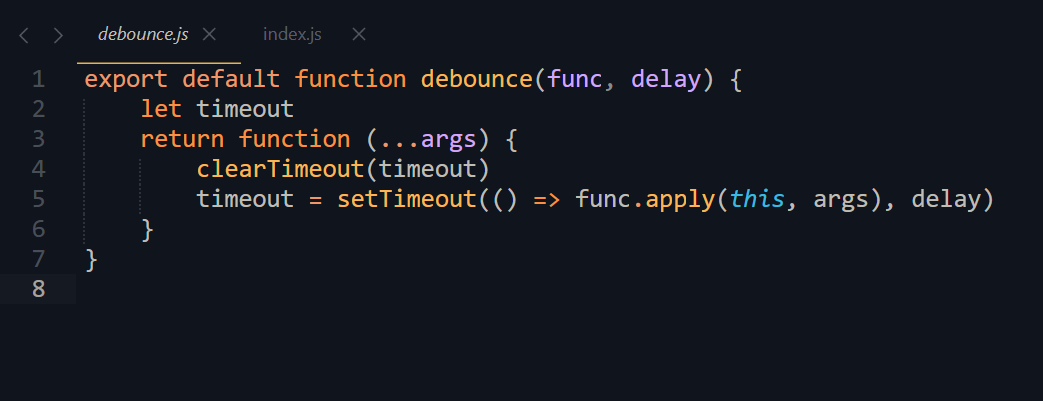


Рисунок 14. Код функції debounce

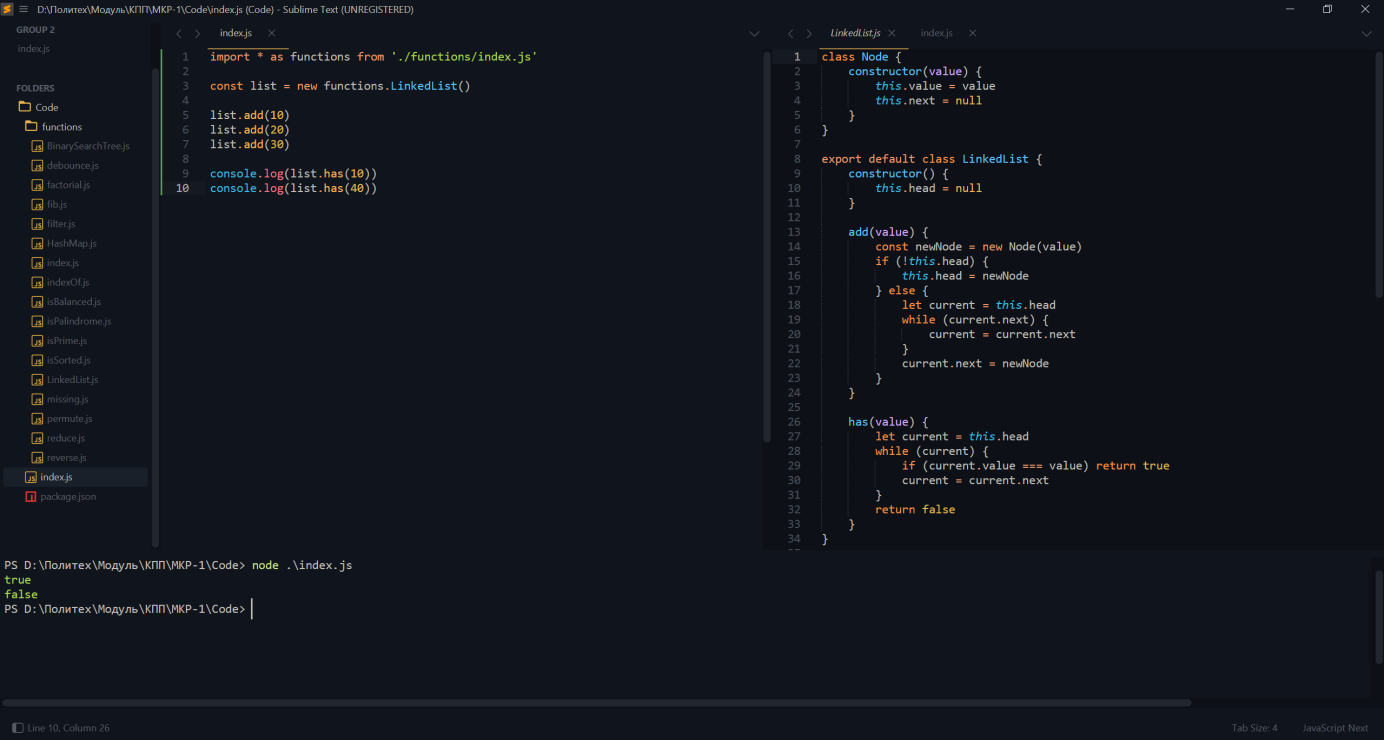


Рисунок 15. Код LinkedList, виклик та результат.



Рисунок 16. Код HashMap, виклик та результат.



Рисунок 17. Код BinarySearchTree, виклик та результат.