МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ федеральное государственное образовательное учреждение высшего образования САНКТ – ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ

КАФЕДРА № 42

ОТЧЕТ	U			
ЗАЩИЩЕН С ОЦЕНКО	М			
ПРЕПОДАВАТЕЛЬ				
старший преподава	тель		С. Ю. Гуков	
должность, уч. степень,	звание	подпись, дата	инициалы, фамилия	
		.БОРАТОРНОЙ РАБО	TE M2	
	ОТЧЕТОЛА	льога горной раво	IE Nº2	
		Высота дерева		
по курсу: АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ				
РАБОТУ ВЫПОЛНИЛ				
СТУДЕНТ гр. №	4321		К.А. Лебедев	
		подпись, дата	инициалы, фамилия	

СОДЕРЖАНИЕ

1 Цель работы	3
2 Задание	4
3 Ход разработки	5
4 Исходный код программы	6
5 Результаты работы программы	9
6 Вывод	

1 Цель работы

Изучить работу деревьев. Вычислить высоту данного дерева. Научиться хранить и эффективно обрабатывать деревья, даже если в них сотни тысяч вершин

2 Задание

Написать программу, которая найдет высоту дерева. Необходимо реализовать программу, которая будет искать максимальное значение среди высот узлов.

3 Ход разработки

Были реализованы класс задачи и функция для вычисления высоты дерева, написана утилита для прочтения чисел из командной строки на языке TypeScript, интегрирован компилятор в JavaScript, который сразу собирает TypeScript проект без предварительной компиляции.

4 Исходный код программы

```
// index.ts
import Task from "./lib/Task";
new Task(); // инициализация задачи
import readline from "node:readline";
// readline интерфейс
export const readlineInterface = readline.createInterface({
    input: process.stdin,
    output: process.stdout,
});
// utils/getTreeHeight.ts
const getTreeHeight = (length: number, parents: number[]): number => {
    const height = new Array(length).fill(-1);
    // Функция для вычисления высоты узла
    function getHeight(node: number) {
        if (height[node] !== -1) {
            return height[node];
        // Узел - корень (родитель = -1), высота = 1
        if (parents[node] === -1) {
           height[node] = 1;
        } else {
            // Идем в глубину
            height[node] = getHeight(parents[node]) + 1;
        return height[node];
    // Максимальная высота дерева
```

```
return Array.from({ length: length }, (_, index) =>
        getHeight(index)
    ).reduce((acc, item) => Math.max(acc, item), 0);
};
export default getTreeHeight;
// lib/Task.ts
import { readlineInterface } from "../utils/readline";
import getTreeHeight from "../utils/getTreeHeight";
class Task {
   private parentsLength: number;
   private parents: number[];
    constructor() {
        this.parentsLength = 0; // длина массива родителей
        this.parents = []; // массив родителей
        this.read(); // инициализация таски
    }
   // получения ответа на вопрос (Высота дерева)
   getAnswer() {
        return getTreeHeight(this.parentsLength, this.parents);
    }
   read() {
        readlineInterface.question("", (length: string) => {
            this.parentsLength = parseInt(length); // чтение количества
            if (isNaN(this.parentsLength) || this.parentsLength <= 0) {
                console.log(
                    "Пожалуйста, введите корректное положительное число."
                readlineInterface.close();
                return;
            }
            readlineInterface.question("", (arr: string) => {
                this.parents = arr.split(" ").map(Number);
```

```
this.parents.length !== this.parentsLength
? console.log("Введен не корректный массив!")
: console.log(this.getAnswer());

readlineInterface.close();
});
});
}
export default Task;
```

5 Результаты работы программы

Результат работы программы включает в себя вывод очереди команд

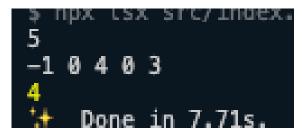


Рисунок 1 – тест программы №1

Рисунок 2 – тест программы №2

6 Вывод

Алгоритмы и структуры данных играют ключевую роль в эффективном решении задач в программировании. Они позволяют оптимизировать процессы и ускорить выполнение операций при правильном использовании.

Использование рекурсии для вычисления высоты дерева демонстрирует важный аспект работы с деревьями, так как многие задачи, связанные с деревьями, можно решать с помощью рекурсивных подходов.

Основная цель программы — вычислить высоту дерева, заданного через массив родителей, где каждый элемент массива указывает на родителя соответствующего узла. Деревья — это структуры данных, состоящие из узлов, связанных между собой, и они широко используются в различных алгоритмах, таких как обход, сортировка и хранение иерархических данных.