МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Алгоритмы и структуры данных»

Тема: Вычисление высоты дерева

Студент гр. 0304 _____ Решоткин А.С.

Преподаватель _____ Берленко Т. А.

Санкт-Петербург

2021

Цель работы.

Изучить структуру данных «Корневое дерево» на примере последовательности parent0, . . . , parentn−1, где parenti - родитель i-й вершины

Задание.

На вход программе подается корневое дерево с вершинами {0, . . . , n−1}, заданное как последовательность $parent_0$, . . . , $parentn_{-1}$, где $parent_i$ — родитель i-й вершины. Требуется вычислить и вывести высоту этого дерева.

Формат входа.

Первая строка содержит натуральное число n. Вторая строка содержит n целых чисел $parent_0$, ... , $parentn_{-1}$. Для каждого $0 \le i \le n - 1$, $parent_i$ — родитель вершины i; если $parent_i = -1$, то i является корнем. Гарантируется, что корень ровно один и что данная последовательность задаёт дерево.

Формат выхода.

Высота дерева.

Примечание: высотой дерева будем считать количество вершин в самом длинном пути от корня к листу.

Выполнение работы.

1. Создание класса TreeNode для представления информации об узле графа. Класс содержит поле children (список дочерних вершин)

2. Создание класса Tree для представления информации о дереве как структуре. Поле inarray содержит входной массив с данными, root-корневой узел.

3. Метод *add_children* позволяет добавлять дочерние вершины.

4. Метод depth позволяет осуществляет поиск максимальной глубины графа.

5. Написан pytest (для примеров ниже).

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№	Входные данные	Выходные данные	Комментарий
1	5 4 -1 4 1 1	3	OK
2	7 1 -1 1 0 2 3 3	4	OK
3	2 1 -1	2	OK
4	0 0	0	OK
5	1 -1	1	OK

Выводы.

Была изучена структура данных «дерево». Была разработана программа, позволяющая найти длину дерева.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
class TreeNode:
    def __init__(self):
        self.children = []

    def add_child(self, child):
        self.children.append(child)

class Tree:
    def __init__(self, inarray):
        self.root = None
        nlist = [None]*len(inarray) #текущий элемент ( список узлов)
        for i in range(0, len(inarray)):
            nlist[i] = TreeNode()
        for i in range(0, len(inarray)):
            if inarray[i] == -1:
                self.root = nlist[i] #находим корень
            else:
                nlist[inarray[i]].add_child(nlist[i])


    def depth(self, root):
        if root is None:
            return 0
        max_depth = 0
        for i in range(0, len(root.children)):
            d = self.depth(root.children[i])
```

```
            if d > max_depth:
                max_depth = d
        return max_depth+1

if __name__== '__main__':
    n = int(input())
    inarray2 = list(map(int, input().split()))
    T = Tree(inarray2)
    print(T.depth(T.root))
```