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

Студентка гр. 1383 _____ Седова Э.А.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург

2022

Цель работы.

Ознакомиться с такой структурой данных как дерево.

Задание.

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

Выполнение работы.

Реализована функция *get_h(tree_data, len)*, которая принимает на вход список родителей i-й вершины и количество вершин дерева.

Далее, в цикле for, вызывается функция *height(tree_data, dict, leaf)*. Она принимает на вход список родителей i-й вершины, словарь и значение переменной *i*, задающей количество повторов цикла. Эта рекурсионная функция совершает обход всего списка *tree_data*. Для каждого элемента вычисляется высота до корня и записывается в словарь, где ключ – это номер вершины, а значение – это высота от данной вершины до корня. Если в процессе нахождения высоты вершины проходится ранее обработанная вершина, с известной высотой, то её высота прибавляется высоте текущей вершины. Это нужно для того, чтобы не вызывать функцию *height()* лишний раз, тем самым ускорив работу программы. Функция *height()* возвращает значение *h* – высоту от конкретной вершины до корня.

По мере поступления этих значений в функцию *get_h()* они попарно сравниваются с помощью функции *max()*. Программа *get_h()* возвращает значение *t* – высоту дерева.

Программа сохраняет входные данные в переменные *len* и *tree_data*. В переменную *tree_data* записывается строка чисел, разделённых пробелом, но с помощью функций *split()* и *map()* строка сначала преобразуется в список, а затем каждый его элемент преобразуется в целое число. Таким образом в

переменной *tree_data* хранится список целых чисел. Далее список *tree_data* и количество вершин *len* передаются в функцию *get_h* и полученный результат выводится в консоль.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные
1.	4, -1, 4, 1, 1	3
2.	-1, 0, 4, 0, 3	4
3.	11, 4, 6, 9, 9, 12, -1, 3, 2, 6, 7, 2, 7, 12	6
4.	5, -1, 6, 2, 3, 4, 1, 4	7
5.	-1	1

Выводы.

Ознакомились со структурой данных “дерево”. Научились вычислять высоту дерева.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
def height(tree_data, dict, leaf):
    h = 1
    if leaf in dict.keys():
        h = dict[leaf]
    else:
        if tree_data[leaf] == -1:
            dict[leaf] = 1
        if tree_data[leaf] != -1:
            h += height(tree_data, dict, tree_data[leaf])
            dict[leaf] = h
    return h

def get_h(tree_data, len):
    t = 0
    dict = {}
    for i in range(len):
        temp = height(tree_data, dict, i)
        t=max(t, temp)
    return t

if __name__ == '__main__':
    len = int(input())
    tree_data= list(map(int, input().split()))
    print(get_h(tree_data, len))
```