Яндекс. Тренировки по алгоритмам 2.0, занятие 8 (B)

2 апр 2024, 06:16:38
старт: 23 сен 2021, 14:00:00
...

Объявления жюри

A. Бинарное дерево (вставка, поиск, обход)

Ограничение времени	1 секунда
Ограничение памяти	64Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Напишите программу, которая будет реализовывать действия в бинарном дереве поиска «вставить» и «найти» (по значению). Программа должна обрабатывать запросы трёх видов:
ADD n — если указанного числа еще нет в дереве, вставлять его и выводить слово «DONE», если уже есть — оставлять дерево как было и выводить слово «ALREADY».

SEARCH — следует выводить слово «YES» (если значение найдено в дереве) или слово «NO» (если не найдено). Дерево при этом не меняется.

PRINTTREE — выводить все дерево, обязательно используя алгоритм, указанный в формате вывода результатов.

Формат ввода

В каждой строке входных данных записан один из запросов ADD n или SEARCH n или PRINTTREE. Гарантируется, что запросы PRINTTREE будут вызываться только в моменты, когда дерево не пустое. Общее количество запросов не превышает 1000, из них не более 20 запросов PRINTTREE.

Формат вывода

Для каждого запроса выводите ответ на него. Для запросов ADD и SEARCH — соответствующее слово в отдельной строке. На запрос PRINTTREE надо выводить дерево, обязательно согласно такому алгоритму:
1) Распечатать левое поддерево

2) Вывести количество точек, равное глубине узла

3) Вывести значение ключа

4) Распечатать правое поддерево

Пример

Ввод	Вывод
ADD 2	DONE
ADD 3	DONE
ADD 2	ALREADY
SEARCH 2	YES
ADD 5	DONE
PRINTTREE	2
SEARCH 7	.3
	..5
	NO

Язык Python 3.9 (PyPy 7.3.11)

Набрать здесь | Отправить файл

```
class BinarySearchTree():
    def __init__(self, val=None, left=None, right=None) -> None:
        if val is not None:
            self.root = [val, left, right]
        else:
            self.root = None

    def add(self, val, node=None, level=1):
        if self.root is None:
            self.root = [val, None, None]
            return 1

        if node is not None:
            curr_node = node
        else:
            curr_node = self.root
            level = 1

        if val == curr_node[0]:
            return 0
        elif val < curr_node[0]:
            if curr_node[1] is None:
                curr_node[1] = [val, None, None]
                return level+1
            else:
                return self.add(val, curr_node[1], level+1)
        elif val > curr_node[0]:
            if curr_node[2] is None:
                curr_node[2] = [val, None, None]
                return level+1
            else:
                return self.add(val, curr_node[2], level+1)

    def find(self, val, node=None):
            if self.root is None:
                return False

            if node is not None:
```

Отправить осталось 94 попытки

Предыдущая Следующая

Время посылки	ID	Задача	Компилятор	Вердикт	Тип посылки	Время	Память	Тест	Баллы	
25 ноя 2023, 22:17:25	99281826	A	Python 3.9 (PyPy 7.3.11)	OK	-	307ms	28.09Mb	-	-	отчёт
25 ноя 2023, 22:09:09	99280496	A	Python 3.9 (PyPy 7.3.11)	WA	-	188ms	28.09Mb	3	-	отчёт
25 ноя 2023, 21:54:01	99277846	A	Python 3.9 (PyPy 7.3.11)	WA	-	202ms	28.09Mb	1	-	отчёт
25 ноя 2023, 21:53:32	99277766	A	Python 3.9 (PyPy 7.3.11)	WA	-	191ms	28.10Mb	2	-	отчёт
25 ноя 2023, 20:41:29	99262778	A	Python 3.9 (PyPy 7.3.11)	WA	-	185ms	28.09Mb	3	-	отчёт
25 ноя 2023, 20:15:18	99256823	A	Python 3.9 (PyPy 7.3.11)	WA	-	191ms	28.09Mb	2	-	отчёт