“Бінарні дерева пошуку”

Бінарні дерева пошуку (binary search trees) являють собою бінарні дерева, які мають наступну властивість: для кожного вузла X елементи, які знаходяться у лівому піддереві X, будуть мати значення менше за X, а елементи у правому піддереві — більше за X.

Завдання:

У даній роботі необхідно виконати два завдання.

1. Перетворити вхідне бінарне дерево у бінарне дерево пошуку

На вхід подається деяке бінарне дерево, із фіксованою структурою (тобто зв’язками між вузлами, їх батьком та нащадками). Необхідно переписати значення вузлів дерева таким чином, щоби:

a) їх нові значення брались тільки з того набору, який присутній у вхідному дереві;

b) зберігалась внутрішня структура дерева (зв’язки між вузол-батько та вузол-нащадки).

Наприклад, нехай задане наступне дерево:

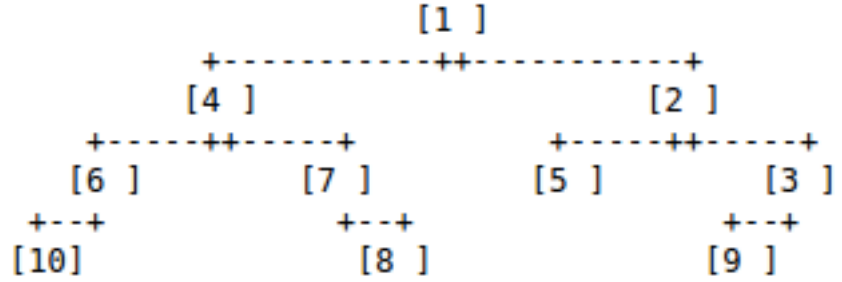

Рис. 1.

В ньому присутні 10 вузлів із значеннями від 1 до 10. Необхідно переписати значення вузлів так, щоби структура дерева зберігалась, але воно стало бінарним деревом пошуку:

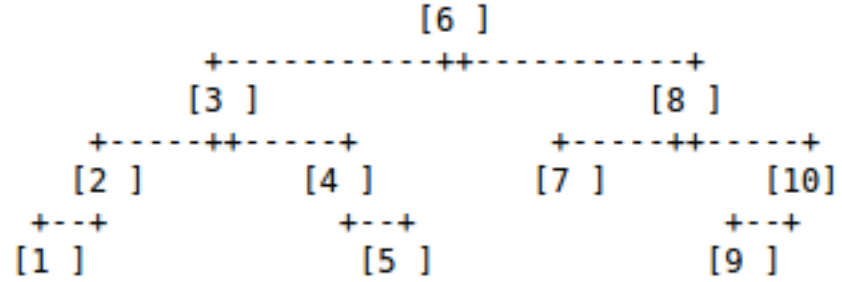

Рис. 2.

Як бачимо, для цього дерева (рис. 2) виконується умова бінарних дерев пошуку.

Маємо ще один приклад вхідного дерева:

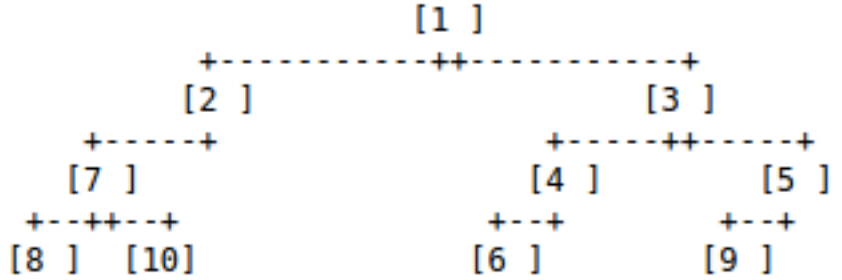

Рис. 3.

та відповідне йому бінарне дерево пошуку:

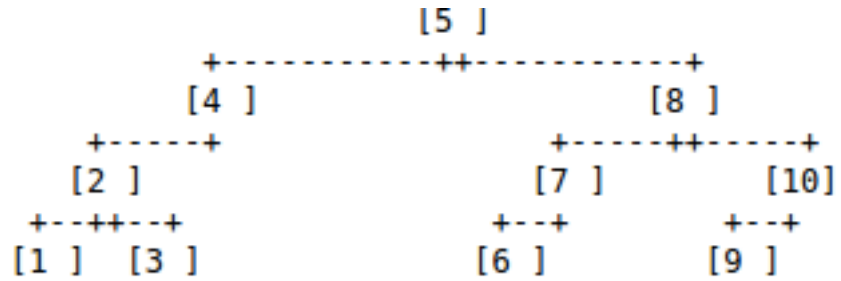

Рис. 4.

Розв’язати цю задачу можна за допомогою наступного алгоритму:

1. Обійти задане дерево у внутрішньому порядку (in-order) та зберігати всі значення в масиві.
2. Відсортувати масив у зростаючому порядку.
3. Знову обійти дерево у внутрішньому порядку та послідовно вписати значення відсортованого масиву у вузли дерева за порядком обходу.

2. Пошук сум послідовних вузлів в дереві

Після того, як вхідне дерево перетворене на бінарне дерево пошуку, необхідно розв’язати наступну задачу. Додатково задається деяке число S. В отриманому бінарному дереві пошуку необхідно знайти всі такі монотонні шляхи (які не обов'язково йдуть від кореня, але всі прямують згори вниз), що сума значень вузлів, які належать знайденим шляхам, дорівнює числу S.

Наприклад, для першого прикладу бінарного дерева (Рис. 2) і для S = 9 маємо три монотонні шляхи в дереві, сума вузлів яких утворює 9: 6+3, 4+5 та окремо вузол 9.

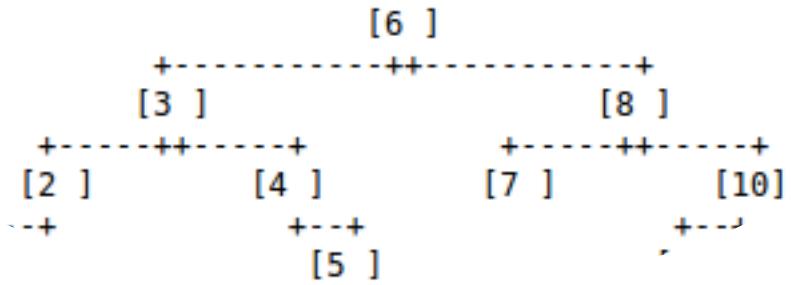

Рис 5.

Для дерева на рис. 4 та S = 9 маємо також три монотонних шляхи: 4+2+3, 5+4 та окремо вузол 9.

Формат вхідних/вихідних даних

Розроблена програма повинна зчитувати вхідні дані з файлу заданого формату та записувати дані у файл заданого формату. У вхідному файлі зберігається задане бінарне дерево

Вхідний файл представляє собою текстовий файл, в якому в один рядок записані всі вузли дерева, якщо обходити його у прямому порядку (pre-order tree walk). У цей обхід також додаються нульові значення для того, щоб позначити відсутність тих або інших листків дерева.

Наприклад, дерево на рис. 1 має наступний вхідний файл:

1 4 6 10 0 0 0 7 0 8 0 0 2 5 0 0 3 9 0 0 0

Перший елемент — завжди корінь. Наступний елемент — його лівий нащадок 4. Потім йде рекурсивний запис елементів вже для лівого нащадку кореня (вузол 4) — його лівий нащадок 6 і т.д. Вузол 10 є листком, тож щоб позначити це у вхідному файлі зазначається,

що обидва його потенційних нащадки відсутні — після 10 йде два нулі (0 0). Далі у файлі йде ще один 0, який вказує на відсутність правого нащадка для вузла 6.
У наведеному вище дереві вказані нульові елементи ({0}), які позначають відсутність нащадку.
Ім'я вхідного файлу та число суми S передається в якості аргументів командного рядку.
Вихідний файл є текстовим. Кожен його рядок містить одну монотонну послідовність вузлів.
Числа в рядку записані через пробіл. Так для наведеного вище дерева (рис. 1) та S = 9, як вже зазначалось, маємо три послідовності: 6+3, 4+5 та окремо вузол 9. Тож вихідний файл буде мати вигляд:

```
4 5
9
6 3
```

Приклад вихідного файлу для дерева з рис. 1.