87. Scramble String

Hard 536 724 Add to List Share

Given a string *s1*, we may represent it as a binary tree by partitioning it to two non-empty substrings recursively.

Below is one possible representation of *s1* = `"great"`:

```
    great
   /    \
  gr    eat
 / \    /  \
g   r  e   at
           / \
          a   t
```

To scramble the string, we may choose any non-leaf node and swap its two children.

For example, if we choose the node `"gr"` and swap its two children, it produces a scrambled string `"rgeat"`.

```
    rgeat
   /    \
  rg    eat
 / \    /  \
r   g  e   at
           / \
          a   t
```

We say that `"rgeat"` is a scrambled string of `"great"`.

Similarly, if we continue to swap the children of nodes `"eat"` and `"at"`, it produces a scrambled string `"rgtae"`.

```
    rgtae
```

```
          a   t
```

We say that `"rgeat"` is a scrambled string of `"great"`.

Similarly, if we continue to swap the children of nodes `"eat"` and `"at"`, it produces a scrambled string `"rgtae"`.

```
    rgtae
   /    \
  rg    tae
 / \    /  \
r   g  ta  e
      / \
     t   a
```

We say that `"rgtae"` is a scrambled string of `"great"`.

Given two strings *s1* and *s2* of the same length, determine if *s2* is a scrambled string of *s1*.

Example 1:

```
Input: s1 = "great", s2 = "rgeat"
Output: true
```

Example 2:

```
Input: s1 = "abcde", s2 = "caebd"
Output: false
```

Accepted 111,352 | Submissions 330,052

Seen this question in a real interview before? Yes No