

# Problem 816: Ambiguous Coordinates

## Problem Information

**Difficulty:** Medium

**Acceptance Rate:** 56.25%

**Paid Only:** No

**Tags:** String, Backtracking, Enumeration

## Problem Description

We had some 2-dimensional coordinates, like `"(1, 3)"` or `"(2, 0.5)"`. Then, we removed all commas, decimal points, and spaces and ended up with the string `s`.

\* For example, `"(1, 3)"` becomes `s = "(13)"` and `"(2, 0.5)"` becomes `s = "(205)"`.

Return `_` a list of strings representing all possibilities for what our original coordinates could have been `_`.

Our original representation never had extraneous zeroes, so we never started with numbers like `"00"`, `"0.0"`, `"0.00"`, `"1.0"`, `"001"`, `"00.01"`, or any other number that can be represented with fewer digits. Also, a decimal point within a number never occurs without at least one digit occurring before it, so we never started with numbers like `".1"`.

The final answer list can be returned in any order. All coordinates in the final answer have exactly one space between them (occurring after the comma.)

**Example 1:**

**Input:** `s = "(123)"` **Output:** `["(1, 2.3)", "(1, 23)", "(1.2, 3)", "(12, 3)"]`

**Example 2:**

**Input:** `s = "(0123)"` **Output:** `["(0, 1.23)", "(0, 12.3)", "(0, 123)", "(0.1, 2.3)", "(0.1, 23)", "(0.12, 3)"]` **Explanation:** `0.0`, `00`, `0001` or `00.01` are not allowed.

**Example 3:**

**\*\*Input:\*\*** s = "(00011)" **\*\*Output:\*\*** ["(0, 0.011)", "(0.001, 1)"]

**\*\*Constraints:\*\***

\* `4 <= s.length <= 12` \* `s[0] == '('` and `s[s.length - 1] == ')'`. \* The rest of `s` are digits.

## Code Snippets

### C++:

```
class Solution {
public:
    vector<string> ambiguousCoordinates(string s) {

    }
};
```

### Java:

```
class Solution {
    public List<String> ambiguousCoordinates(String s) {

    }
}
```

### Python3:

```
class Solution:
    def ambiguousCoordinates(self, s: str) -> List[str]:
```