

1079. Letter Tile Possibilities

You have `n` `tiles`, where each tile has one letter `tiles[i]` printed on it.

Return *the number of possible non-empty sequences of letters* you can make using the letters printed on those `tiles`.

Example 1:

Input: `tiles = "AAB"`

Output: `8`

Explanation: The possible sequences are "A", "B", "AA", "AB", "BA", "AAB", "ABA", "BAA".

Example 2:

Input: `tiles = "AAABBC"`

Output: `188`

Example 3:

Input: `tiles = "v"`

Output: `1`

```
def numTilePossibilities(self, tiles: str) -> int:
    seen = set()                # Store indicies we have seen during
    our backtrack dfs
    answers = set()             # Store valid tile combinations
    self.backtrack(tiles, seen, answers, '')
    return len(answers)         # Return the NUMBER of possible
    answers, not actual

    # Recursive Function
    def backtrack(self, tiles, seen, answers, curr):
        if curr != '' and curr not in answers:
            answers.add(curr)

        for index in range(len(tiles)):
            if index not in seen:
                seen.add(index)
```

```
self.backtrack(tiles, seen, answers, curr + tiles[index])  
seen.remove(index)
```