Program 62. Number of Ways of Cutting a Pizza

Given a rectangular pizza represented as a rows x cols matrix containing the following characters: 'A' (an apple) and '.' (empty cell) and given the integer k. You have to cut the pizza into k pieces using k-1 cuts.

For each cut you choose the direction: vertical or horizontal, then you choose a cut position at the cell boundary and cut the pizza into two pieces. If you cut the pizza vertically, give the left part of the pizza to a person. If you cut the pizza horizontally, give the upper part of the pizza to a person. Give the last piece of pizza to the last person.

Return the number of ways of cutting the pizza such that each piece contains at least one apple. Since the answer can be a huge number, return this modulo 10^9 + 7.

Example 1:

Input: pizza = ["A..","AAA","..."], k = 3

Output: 3

Explanation: The figure above shows the three ways to cut the pizza. Note that pieces must contain at least one apple.

Program:

```
def ways(pizza, k):
    MOD = 10**9 + 7
    rows, cols = len(pizza), len(pizza[0])

    # Precompute the number of apples in each subrectangle using a prefix sum approach
    apples = [[0] * (cols + 1) for _ in range(rows + 1)]
    for r in range(rows - 1, -1, -1):
        for c in range(cols - 1, -1, -1):
            apples[r][c] = (1 if pizza[r][c] == 'A' else 0) + apples[r+1][c] + apples[r][c+1] - apples[r+1][c+1]

    # Memoization dictionary
    memo = {}

    # Define the recursive function with memoization
    def dp(r, c, k):
        if (r, c, k) in memo:
            return memo[(r, c, k)]

        # If no more cuts are needed, return 1 if there's at least one apple, else 0
        if k == 0:
            return 1 if apples[r][c] > 0 else 0

        # Count the ways to cut the pizza
        ways_to_cut = 0

        # Horizontal cuts
        for nr in range(r + 1, rows):
            if apples[r][c] - apples[nr][c] > 0:  # There are apples in the upper part
                ways_to_cut = (ways_to_cut + dp(nr, c, k - 1)) % MOD

        # Vertical cuts
        for nc in range(c + 1, cols):
```

```
            if apples[r][c] - apples[r][nc] > 0:  # There are apples in the left part
                ways_to_cut = (ways_to_cut + dp(r, nc, k - 1)) % MOD

        memo[(r, c, k)] = ways_to_cut
        return ways_to_cut

    # Initial call from the top-left corner of the pizza
    return dp(0, 0, k - 1)

# Example usage
pizza = ["A..", "AAA", "..."]
k = 3
print(ways(pizza, k))  # Output: 3
```

Output:

```
"C:\Program Files\Python312\python.exe" "C:\Work Space\DAA COADS.PYTHON\program 62.py"
3

Process finished with exit code 0
```

Time complexity:
O(rows×cols×k×(rows+cols))