Assignment1

Contributor: 毕明阳，张泽凯

Task 1: Counting the words

Problem Description:

Write a program that allows the user to input a sentence composed of multiple words. The program should count the occurrences of each word and output the results in alphabetical order.

Input:

- $N(1 \leq N \leq 1000)$ word(s), seperated by commas. Each word is no more than 30 letters.

Output:

- Output each word and its occurrence count, sorted in alphabetical order.

Example 1:

Input:

```
apple,banana,apple,orange,banana,apple
```

Output:

```
apple: 3
banana: 2
orange: 1
```

Task 2: Distinct Teams Formation

Problem Description:

You are organizing a competition where you need to form teams. There are `N` people available, and you need to form a team of `K` distinct people. Your task is to calculate how many different ways you can form such teams.

Input:

- The first line contains two integers `N` and `K` (1 ≤ K ≤ N ≤ 20), where `N` is the total number of people and `K` is the number of people to be selected for the team.

Output:

- Output a single integer representing the number of ways to form the team, i.e., the value of `C(N, K)`.

Formula:

- $C(N, K) = \frac{N!}{K!(N-K)!}$

Example 1:

Input:

```
5 3
```

Output:

```
10
```

Example 2:

Input:

```
6 2
```

Output:

```
15
```

Task 3: 2048

Problem Description

You are likely familiar with or have heard of the game "2048", where the player makes movement decisions while adjacent tiles with the same value keeps merging into a single tile with double value until the value reaches 2048. In this task, you will implement a portion of the game's logic. Given an initial grid state and a sequence of moves, your job is to output the final grid state after applying all the moves.

Note: For simplicity, no new tiles will appear after each move in this problem. Your focus is only on merging tiles of the same value during each step.

Input

- The first line contains an integer N ($4 \leq N \leq 100$), which is the fixed size of the grid ($N \times N$).
- The next N lines each contain N ($0 \leq \text{value} \leq 2^{11}$) integers which are power of two representing the initial state of the grid. Each integer corresponds to a tile on the grid, with 0 representing an empty tile.

- The final line contains a sequence of moves ($1 \leq \text{the number of moves} \leq 100$), which consists of a sequence of movements (`left`, `right`, `up`, `down`) separated by spaces.

Output

- The final state of the grid after applying all the specified moves.

Example 1

Input:

```
4
2 2 0 0
0 0 0 0
0 0 0 0
0 0 0 0
right right
```

Output:

```
0 0 0 4
0 0 0 0
0 0 0 0
0 0 0 0
```

Explanation

1. First move (right): The two '2' tiles merge into a single '4' tile. The row becomes `0 0 0 4`.
2. Second move (right): No additional merges occur as there are no tiles to move into the rightmost position. The final state remains `0 0 0 4`.

Example 2

Input:

```
4
2 4 4 2
2 0 0 2
0 0 0 0
0 0 0 0
left down right
```

Output:

```
0 0 0 0
0 0 0 0
0 0 0 2
0 4 8 2
```

Explanation

1. First move (left): The first row will have two '4' tiles merged, and the second row will have two '2' tiles merged. After the left move, the state would be:

```
2 8 2 0
4 0 0 0
0 0 0 0
0 0 0 0
```

2. Second move (down): Tiles will move down if possible and merge where applicable. After the down move, the state would be:

```
0 0 0 0
0 0 0 0
2 0 0 0
4 8 2 0
```

3. Final move (right): After the right move, the state would be:

```
0 0 0 0
0 0 0 0
0 0 0 2
0 4 8 2
```

Task 4: Valorant Token

Problem Description:

`Cypher` is the security guard of the `Ascent` map, responsible for guarding the main gate and preventing unfamiliar Teenagers from breaking in and causing trouble. For instance, `Jett` may dash in, `Raze` may jump in, and `Neon` may slide in. These impolite individuals will all be stopped by `Cypher`.

To ensure that only those who meet the requirements can enter, `Cypher` checks their tokens and determines who is allowed entry based on the following rules:

1. The token must have at least 8 characters.
2. It must contain at least one lowercase letter.
3. It must contain at least one uppercase letter.
4. It must contain at least one digit.
5. It must contain at least one special character, which should be one of the following: `!@#$%^&*()-+`.
6. It must not contain two consecutive identical characters (for example, "bee" does not meet this condition, but "ebe" does).

Based on the inspection results, `Cypher` will make the following judgments:

- If the token meets all the conditions, only `Jett` is allowed to enter, output `Jett`.

- If the token satisfies any five of the conditions, only `Raze` is allowed to enter, output `Raze`.

- If the token satisfies any four of the conditions, only `Neon` is allowed to enter, output `Neon`.

- In all other cases, no one is allowed to enter.

Through this token inspection process, `Cypher` can determine who is permitted to enter. Please design a program to help `Cypher` quickly determine who can be granted access based on the token. If no one is allowed, the program should output `No`.

Input:

- String s, representing the token.

Output:

- Output the person who is allowed to enter, the output can only be one of the set $\{Jett, Raze, Neon, No\}$.

Example 1:

Input:

```
ccanHorcusbeatsFan
```

Output:

```
No
```

Example 2:

Input:

```
UrP0werles!
```

Output:

```
Jett
```

Example 3:

Input:

```
UrP0werless!
```

Output:

```
Raze
```

Example 4:

Input:

```
UrPowerless!
```

Output:

```
Neon
```