Super Reduced String

Shil has a string, S, consisting of N lowercase English letters. In one operation, he can delete any *pair of adjacent letters* with same value. For example, string "$\mathbf{aabcc}$" would become either "$\mathbf{aab}$" or "$\mathbf{bcc}$" after 1 operation.

Shil wants to reduce S as much as possible. To do this, he will repeat the above operation as many times as it can be performed. Help Shil out by finding and printing S's non-reducible form!

Note: If the final string is empty, print `Empty String`.

Input Format

A single string, S.

Constraints

- $1 \leq N \leq 100$

Output Format

If the final string is empty, print `Empty String`; otherwise, print the final non-reducible string.

Sample Input 0

```
aaabccddd
```

Sample Output 0

```
abd
```

Sample Input 1

```
baab
```

Sample Output 1

```
Empty String
```

Sample Input 2

```
aa
```

Sample Output 2

```
Empty String
```

Explanation

Sample Case 0:

Shil can perform the following sequence of operations to get the final string:

1. $\mathbf{aaabccddd} \rightarrow \mathbf{abccddd}$
2. $\mathbf{abccddd} \rightarrow \mathbf{abddd}$
3. $\mathbf{abddd} \rightarrow \mathbf{abd}$

Thus, we print **abd**.

Sample Case 1:
Shil can perform the following sequence of operations to get the final string:

1. $\mathbf{baab} \rightarrow \mathbf{bb}$
2. $\mathbf{bb} \rightarrow \mathbf{Empty\ String}$