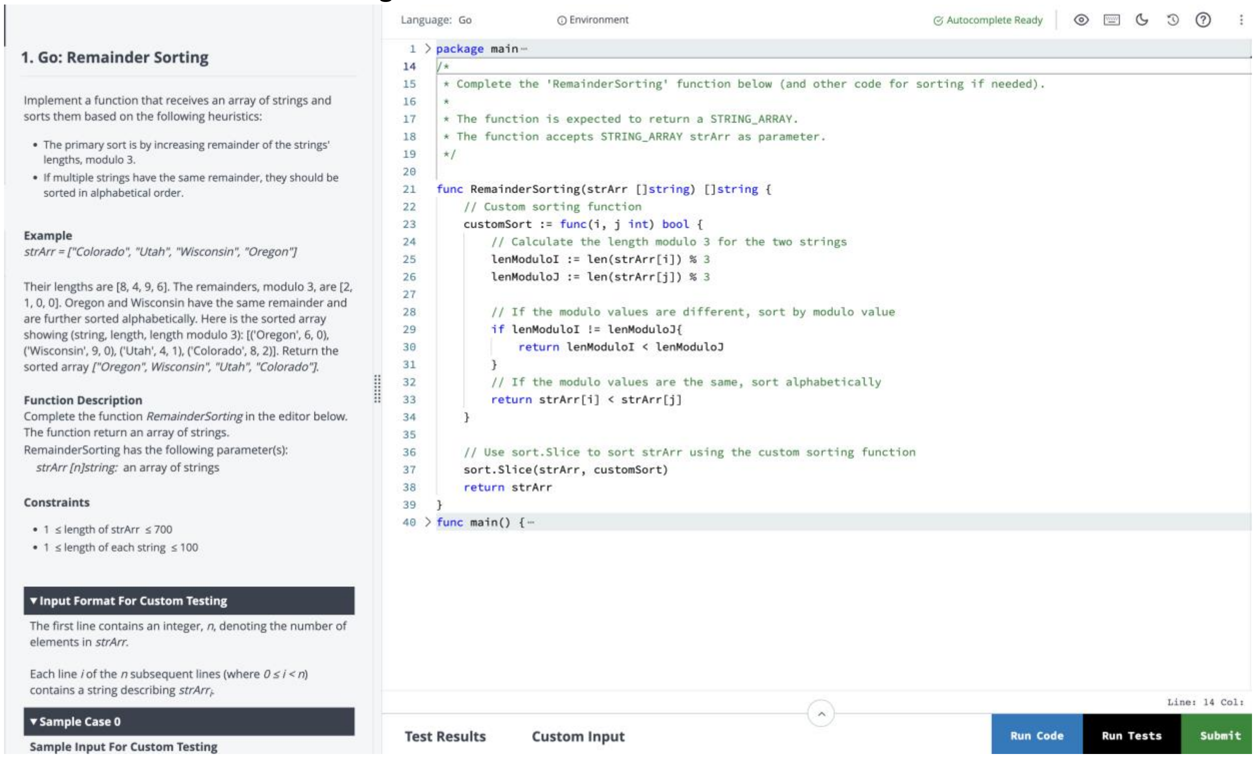
Golang Basic

1. Go: Remainder Sorting



1. Go: Remainder Sorting

Implement a function that receives an array of strings and sorts them based on the following heuristics:

- The primary sort is by increasing remainder of the strings' lengths, modulo 3.
- If multiple strings have the same remainder, they should be sorted in alphabetical order.

Example
strArr = ["Colorado", "Utah", "Wisconsin", "Oregon"]

Their lengths are [8, 4, 9, 6]. The remainders, modulo 3, are [2, 1, 0, 0]. Oregon and Wisconsin have the same remainder and are further sorted alphabetically. Here is the sorted array showing (string, length, length modulo 3): [('Oregon', 6, 0), ('Wisconsin', 9, 0), ('Utah', 4, 1), ('Colorado', 8, 2)]. Return the sorted array *["Oregon", Wisconsin", "Utah", "Colorado"]*.

Function Description
Complete the function *RemainderSorting* in the editor below. The function return an array of strings.
RemainderSorting has the following parameter(s):
strArr [n]string: an array of strings

Constraints

- 1 ≤ length of strArr ≤ 700
- 1 ≤ length of each string ≤ 100

▾ Input Format For Custom Testing

The first line contains an integer, *n*, denoting the number of elements in *strArr*.

Each line *i* of the *n* subsequent lines (where *0 ≤ i < n*) contains a string describing $strArr_i$.

▾ Sample Case 0

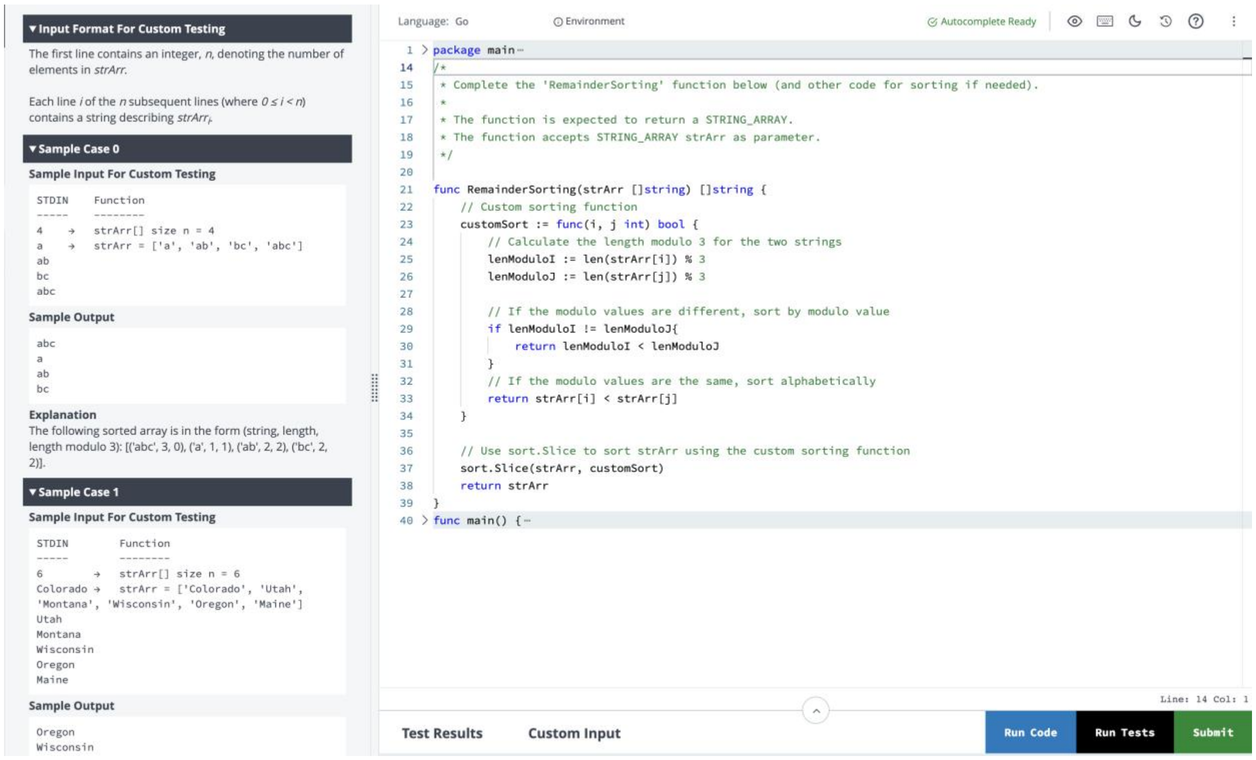
Sample Input For Custom Testing

Language: Go   Environment   Autocomplete Ready

```go
package main…
/*
 * Complete the 'RemainderSorting' function below (and other code for sorting if needed).
 *
 * The function is expected to return a STRING_ARRAY.
 * The function accepts STRING_ARRAY strArr as parameter.
 */

func RemainderSorting(strArr []string) []string {
    // Custom sorting function
    customSort := func(i, j int) bool {
        // Calculate the length modulo 3 for the two strings
        lenModuloI := len(strArr[i]) % 3
        lenModuloJ := len(strArr[j]) % 3

        // If the modulo values are different, sort by modulo value
        if lenModuloI != lenModuloJ{
            return lenModuloI < lenModuloJ
        }
        // If the modulo values are the same, sort alphabetically
        return strArr[i] < strArr[j]
    }

    // Use sort.Slice to sort strArr using the custom sorting function
    sort.Slice(strArr, customSort)
    return strArr
}
func main() {…
```

Line: 14 Col:

Test Results   Custom Input   Run Code   Run Tests   Submit



▾ Input Format For Custom Testing

The first line contains an integer, *n*, denoting the number of elements in *strArr*.

Each line *i* of the *n* subsequent lines (where *0 ≤ i < n*) contains a string describing $strArr_i$.

▾ Sample Case 0

Sample Input For Custom Testing

```
STDIN     Function
-----     --------
4    →    strArr[] size n = 4
a    →    strArr = ['a', 'ab', 'bc', 'abc']
ab
bc
abc
```

Sample Output

```
abc
a
ab
bc
```

Explanation
The following sorted array is in the form (string, length, length modulo 3): [('abc', 3, 0), ('a', 1, 1), ('ab', 2, 2), ('bc', 2, 2)].

▾ Sample Case 1

Sample Input For Custom Testing

```
STDIN        Function
-----        --------
6        →   strArr[] size n = 6
Colorado →   strArr = ['Colorado', 'Utah',
'Montana', 'Wisconsin', 'Oregon', 'Maine']
Utah
Montana
Wisconsin
Oregon
Maine
```

Sample Output

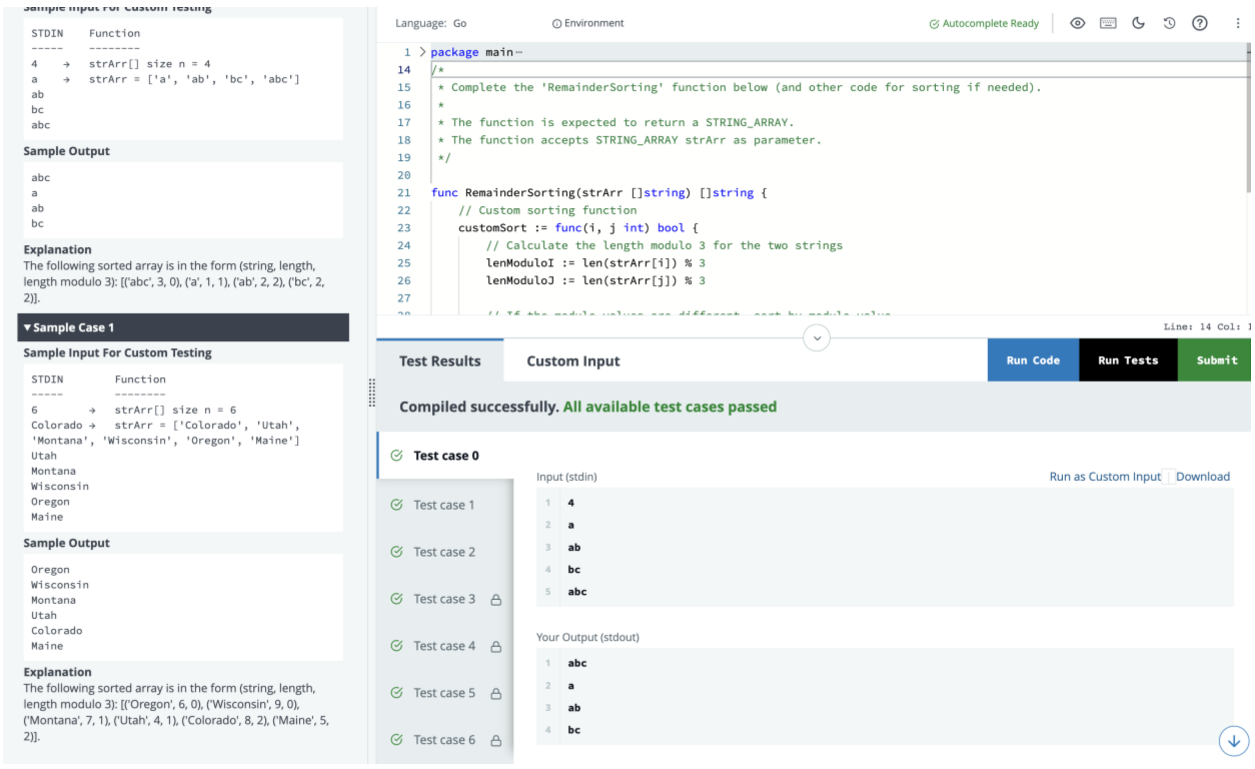
```
Oregon
Wisconsin
```

Language: Go   Environment   Autocomplete Ready

```go
package main…
/*
 * Complete the 'RemainderSorting' function below (and other code for sorting if needed).
 *
 * The function is expected to return a STRING_ARRAY.
 * The function accepts STRING_ARRAY strArr as parameter.
 */

func RemainderSorting(strArr []string) []string {
    // Custom sorting function
    customSort := func(i, j int) bool {
        // Calculate the length modulo 3 for the two strings
        lenModuloI := len(strArr[i]) % 3
        lenModuloJ := len(strArr[j]) % 3

        // If the modulo values are different, sort by modulo value
        if lenModuloI != lenModuloJ{
            return lenModuloI < lenModuloJ
        }
        // If the modulo values are the same, sort alphabetically
        return strArr[i] < strArr[j]
    }

    // Use sort.Slice to sort strArr using the custom sorting function
    sort.Slice(strArr, customSort)
    return strArr
}
func main() {…
```

Line: 14 Col: 1

Test Results   Custom Input   Run Code   Run Tests   Submit



Sample Input For Custom Testing

```
STDIN    Function
-----    --------
4    →   strArr[] size n = 4
a    →   strArr = ['a', 'ab', 'bc', 'abc']
ab
bc
abc
```

Sample Output

```
abc
a
ab
bc
```

Explanation

The following sorted array is in the form (string, length, length modulo 3): [('abc', 3, 0), ('a', 1, 1), ('ab', 2, 2), ('bc', 2, 2)].

▾ Sample Case 1

Sample Input For Custom Testing

```
STDIN        Function
-----        --------
6        →   strArr[] size n = 6
Colorado →   strArr = ['Colorado', 'Utah',
'Montana', 'Wisconsin', 'Oregon', 'Maine']
Utah
Montana
Wisconsin
Oregon
Maine
```

Sample Output

```
Oregon
Wisconsin
Montana
Utah
Colorado
Maine
```

Explanation

The following sorted array is in the form (string, length, length modulo 3): [('Oregon', 6, 0), ('Wisconsin', 9, 0), ('Montana', 7, 1), ('Utah', 4, 1), ('Colorado', 8, 2), ('Maine', 5, 2)].

```go
package main⋯
/*
 * Complete the 'RemainderSorting' function below (and other code for sorting if needed).
 *
 * The function is expected to return a STRING_ARRAY.
 * The function accepts STRING_ARRAY strArr as parameter.
 */

func RemainderSorting(strArr []string) []string {
    // Custom sorting function
    customSort := func(i, j int) bool {
        // Calculate the length modulo 3 for the two strings
        lenModuloI := len(strArr[i]) % 3
        lenModuloJ := len(strArr[j]) % 3

```

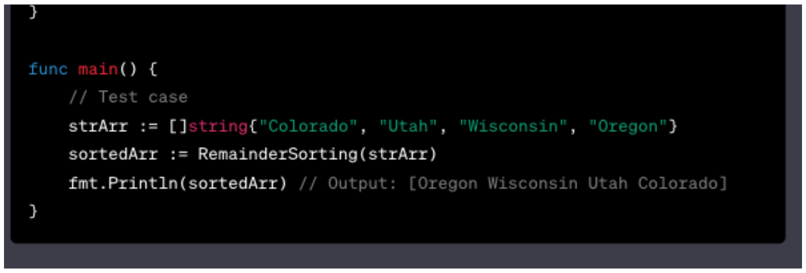
Compiled successfully. All available test cases passed

Input (stdin)

```
4
a
ab
bc
abc
```

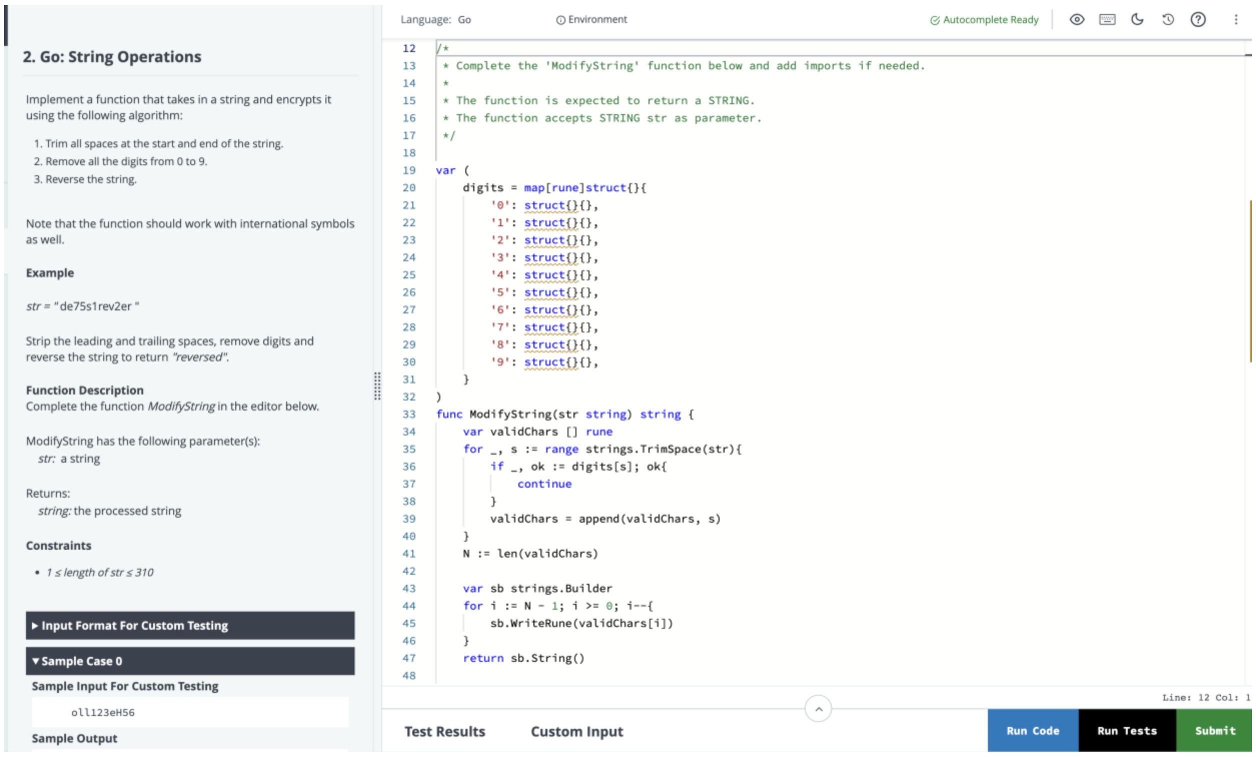
Your Output (stdout)

```
abc
a
ab
bc
```



```go
}

func main() {
    // Test case
    strArr := []string{"Colorado", "Utah", "Wisconsin", "Oregon"}
    sortedArr := RemainderSorting(strArr)
    fmt.Println(sortedArr) // Output: [Oregon Wisconsin Utah Colorado]
}
```

2. Go String Operations



2. Go: String Operations

Implement a function that takes in a string and encrypts it using the following algorithm:

1. Trim all spaces at the start and end of the string.
2. Remove all the digits from 0 to 9.
3. Reverse the string.

Note that the function should work with international symbols as well.

Example

str = "de75s1rev2er "

Strip the leading and trailing spaces, remove digits and reverse the string to return *"reversed"*.

Function Description
Complete the function *ModifyString* in the editor below.

ModifyString has the following parameter(s):
str: a string

Returns:
string: the processed string

Constraints

- *1 ≤ length of str ≤ 310*

▸ Input Format For Custom Testing

▾ Sample Case 0

Sample Input For Custom Testing

```
oll123eH56
```

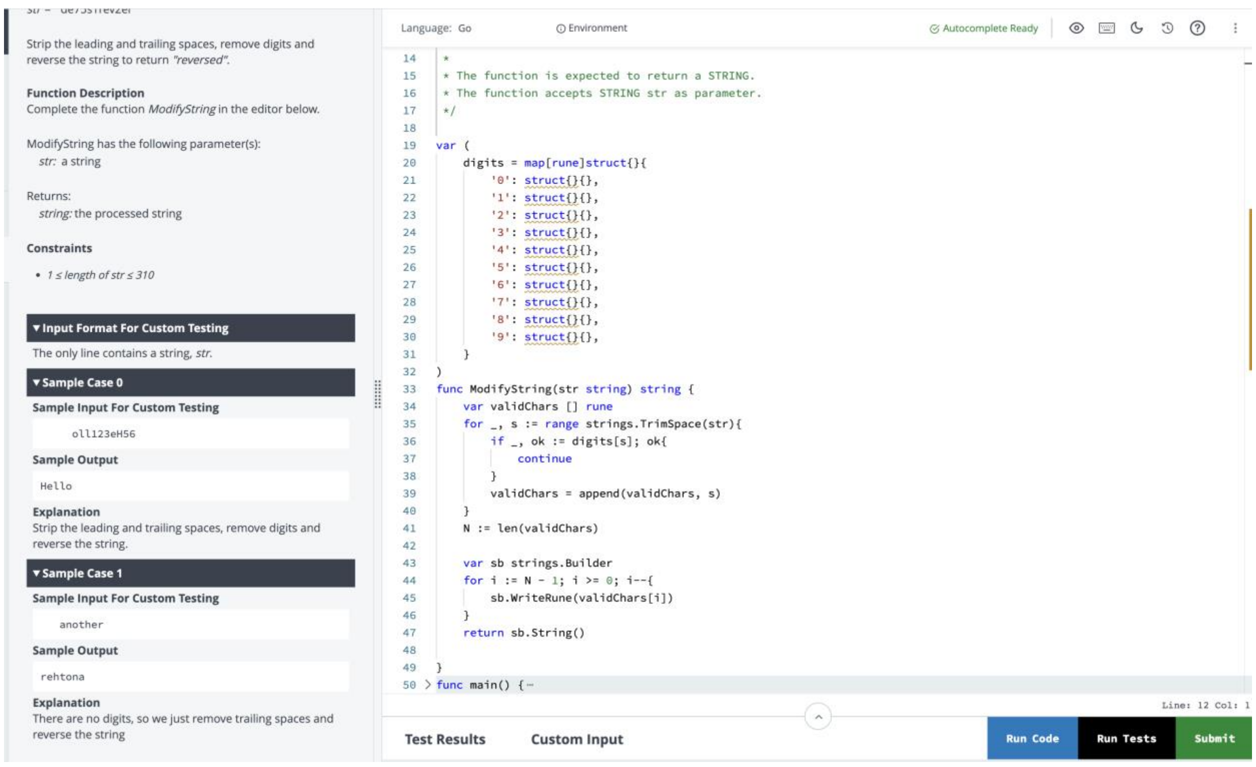
Sample Output

Language: Go   Environment   Autocomplete Ready

```go
/*
 * Complete the 'ModifyString' function below and add imports if needed.
 *
 * The function is expected to return a STRING.
 * The function accepts STRING str as parameter.
 */

var (
    digits = map[rune]struct{}{
        '0': struct{}{},
        '1': struct{}{},
        '2': struct{}{},
        '3': struct{}{},
        '4': struct{}{},
        '5': struct{}{},
        '6': struct{}{},
        '7': struct{}{},
        '8': struct{}{},
        '9': struct{}{},
    }
)
func ModifyString(str string) string {
    var validChars [] rune
    for _, s := range strings.TrimSpace(str){
        if _, ok := digits[s]; ok{
            continue
        }
        validChars = append(validChars, s)
    }
    N := len(validChars)

    var sb strings.Builder
    for i := N - 1; i >= 0; i--{
        sb.WriteRune(validChars[i])
    }
    return sb.String()

```

Line: 12 Col: 1

Test Results   Custom Input   Run Code   Run Tests   Submit



Strip the leading and trailing spaces, remove digits and reverse the string to return *"reversed"*.

Function Description
Complete the function *ModifyString* in the editor below.

ModifyString has the following parameter(s):
str: a string

Returns:
string: the processed string

Constraints

- *1 ≤ length of str ≤ 310*

▾ Input Format For Custom Testing

The only line contains a string, *str*.

▾ Sample Case 0

Sample Input For Custom Testing

```
oll123eH56
```

Sample Output

```
Hello
```

Explanation
Strip the leading and trailing spaces, remove digits and reverse the string.

▾ Sample Case 1

Sample Input For Custom Testing

```
another
```

Sample Output

```
rehtona
```

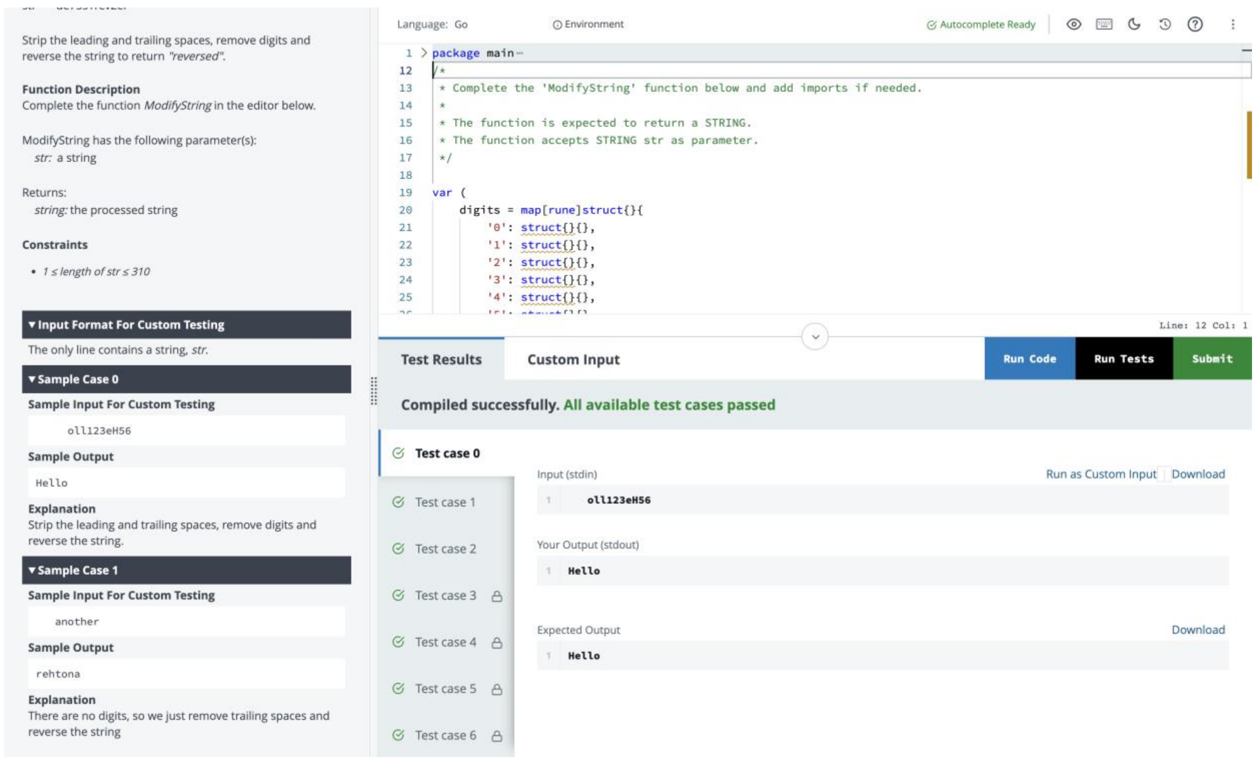
Explanation
There are no digits, so we just remove trailing spaces and reverse the string

Language: Go   Environment   Autocomplete Ready

```go
 *
 * The function is expected to return a STRING.
 * The function accepts STRING str as parameter.
 */

var (
    digits = map[rune]struct{}{
        '0': struct{}{},
        '1': struct{}{},
        '2': struct{}{},
        '3': struct{}{},
        '4': struct{}{},
        '5': struct{}{},
        '6': struct{}{},
        '7': struct{}{},
        '8': struct{}{},
        '9': struct{}{},
    }
)
func ModifyString(str string) string {
    var validChars [] rune
    for _, s := range strings.TrimSpace(str){
        if _, ok := digits[s]; ok{
            continue
        }
        validChars = append(validChars, s)
    }
    N := len(validChars)

    var sb strings.Builder
    for i := N - 1; i >= 0; i--{
        sb.WriteRune(validChars[i])
    }
    return sb.String()

}
> func main() {↔
```

Line: 12 Col: 1

Test Results   Custom Input   Run Code   Run Tests   Submit



Strip the leading and trailing spaces, remove digits and reverse the string to return *"reversed"*.

Function Description
Complete the function *ModifyString* in the editor below.

ModifyString has the following parameter(s):
str: a string

Returns:
string: the processed string

Constraints

- *1 ≤ length of str ≤ 310*

▾ Input Format For Custom Testing

The only line contains a string, *str*.

▾ Sample Case 0

Sample Input For Custom Testing

```
oll123eH56
```

Sample Output

```
Hello
```

Explanation
Strip the leading and trailing spaces, remove digits and reverse the string.

▾ Sample Case 1

Sample Input For Custom Testing

```
   another
```

Sample Output

```
rehtona
```

Explanation
There are no digits, so we just remove trailing spaces and reverse the string

```go
package main⋯
/*
 * Complete the 'ModifyString' function below and add imports if needed.
 *
 * The function is expected to return a STRING.
 * The function accepts STRING str as parameter.
 */

var (
    digits = map[rune]struct{}{
        '0': struct{}{},
        '1': struct{}{},
        '2': struct{}{},
        '3': struct{}{},
        '4': struct{}{},
```

Compiled successfully. All available test cases passed

Input (stdin)

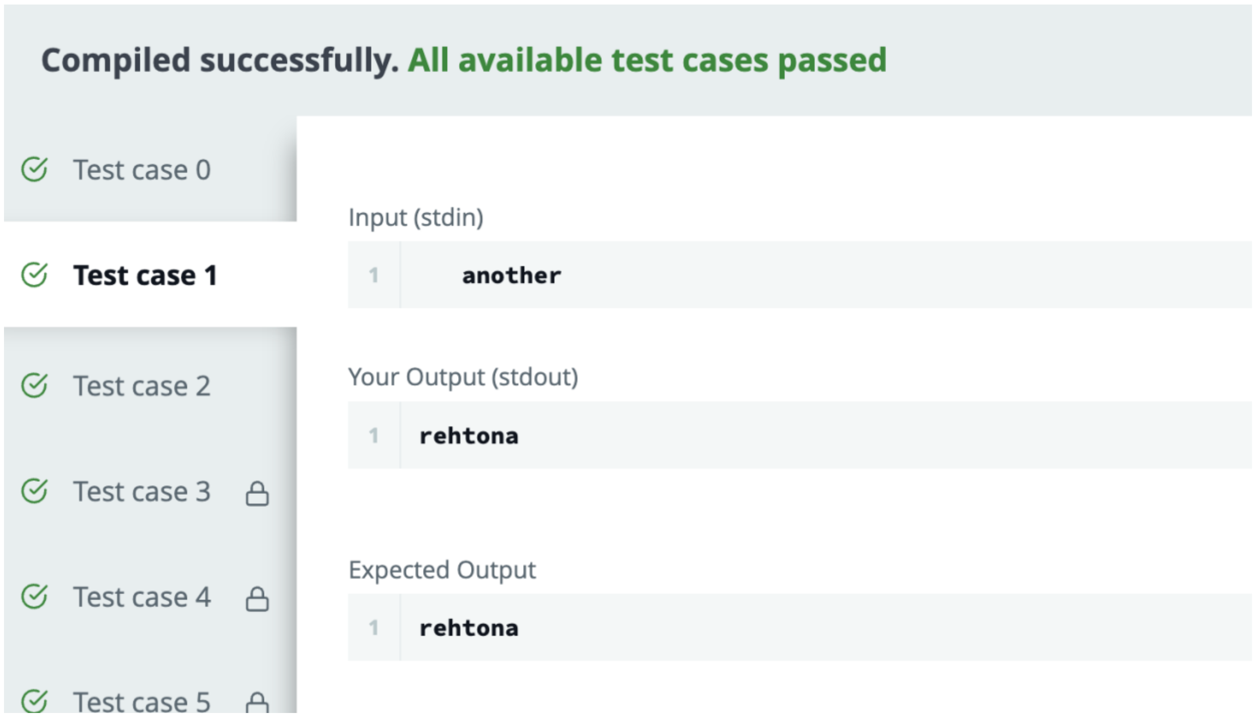
```
oll123eH56
```

Your Output (stdout)

```
Hello
```

Expected Output

```
Hello
```



Compiled successfully. All available test cases passed

Input (stdin)

```
   another
```

Your Output (stdout)

```
rehtona
```

Expected Output

```
rehtona
```