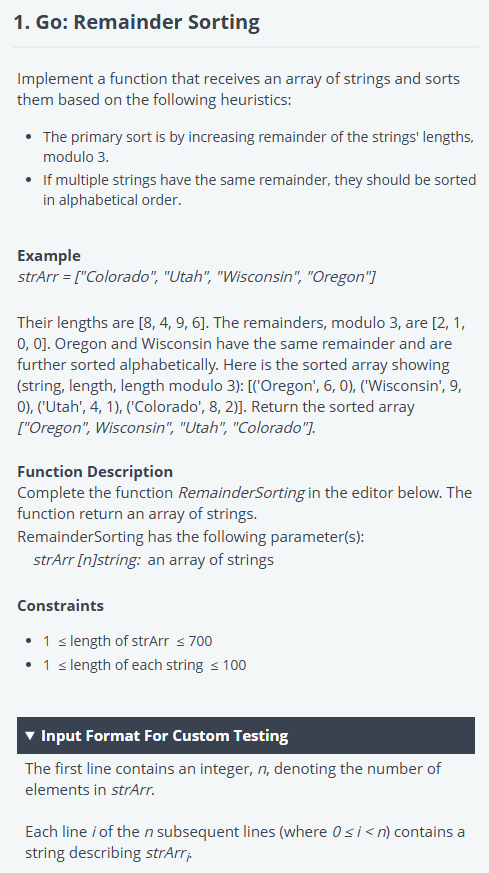
# Go: Remainder Sorting



package main

import (

    "bufio"

    "fmt"

    "io"

    "os"

    "sort"

    "strconv"

    "strings"

)

/\*

 \* 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 {

    // Define a custom sorting logic

    sort.Slice(strArr, func(i, j int) bool {

        lenI := len(strArr[i]) % 3

        lenJ := len(strArr[j]) % 3

        if lenI == lenJ {

            return strArr[i] < strArr[j] // Alphabetical order if remainders are equal

        }

        return lenI < lenJ // Sort by remainder

    })

    return strArr

}

func main() {

    reader := bufio.NewReaderSize(os.Stdin, 16\*1024\*1024)

    stdout, err := os.Create(os.Getenv("OUTPUT\_PATH"))

    checkError(err)

    defer stdout.Close()

    writer := bufio.NewWriterSize(stdout, 16\*1024\*1024)

    strArrCount, err := strconv.ParseInt(strings.TrimSpace(readLine(reader)), 10, 64)

    checkError(err)

    var strArr []string

    for i := 0; i < int(strArrCount); i++ {

        strArrItem := readLine(reader)

        strArr = append(strArr, strArrItem)

    }

    result := RemainderSorting(strArr)

    for i, resultItem := range result {

        fmt.Fprintf(writer, "%s", resultItem)

        if i != len(result)-1 {

            fmt.Fprintf(writer, "\n")

        }

    }

    fmt.Fprintf(writer, "\n")

    writer.Flush()

}

func readLine(reader \*bufio.Reader) string {

    str, \_, err := reader.ReadLine()

    if err == io.EOF {

        return ""

    }

    return strings.TrimRight(string(str), "\r\n")

}

func checkError(err error) {

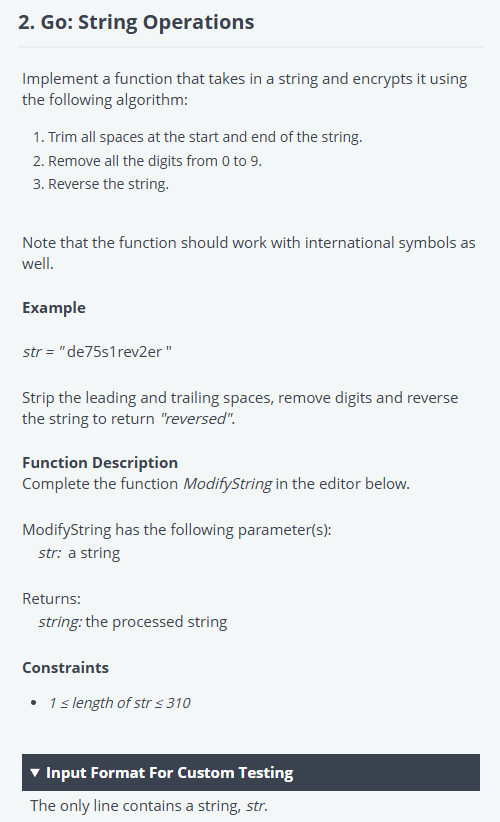
    if err != nil {

        panic(err)

    }

}

# Go: String Operations



package main

import (

    "bufio"

    "fmt"

    "io"

    "os"

    "strings"

    "unicode"

)

/\*

 \* 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.

 \*/

func ModifyString(str string) string {

    // Step 1: Trim spaces

    str = strings.TrimSpace(str)

    // Step 2: Remove digits from the string

    var result strings.Builder

    for \_, char := range str {

        if !unicode.IsDigit(char) {

            result.WriteRune(char)

        }

    }

    // Step 3: Reverse the string

    runes := []rune(result.String())

    for i, j := 0, len(runes)-1; i < j; i, j = i+1, j-1 {

        runes[i], runes[j] = runes[j], runes[i]

    }

    return string(runes)

}

func main() {

    reader := bufio.NewReaderSize(os.Stdin, 16\*1024\*1024)

    stdout, err := os.Create(os.Getenv("OUTPUT\_PATH"))

    checkError(err)

    defer stdout.Close()

    writer := bufio.NewWriterSize(stdout, 16\*1024\*1024)

    str := readLine(reader)

    result := ModifyString(str)

    fmt.Fprintf(writer, "%s\n", result)

    writer.Flush()

}

func readLine(reader \*bufio.Reader) string {

    str, \_, err := reader.ReadLine()

    if err == io.EOF {

        return ""

    }

    return strings.TrimRight(string(str), "\r\n")

}

func checkError(err error) {

    if err != nil {

        panic(err)

    }

}