**Title**

-------------------------------------------------------------------------------------------------------------

**Longest Substring Without Repeating Characters**

-------------------------------------------------------------------------------------------------------------

**Description**

-------------------------------------------------------------------------------------------------------------

Given a string s, find the length of the longest substring without repeating characters.

**Example 1:**

Input: s = "abcabcbb"

Output: 3

Explanation: The answer is "abc", with the length of 3.

**Example 2:**

Input: s = "bbbbb"

Output: 1

Explanation: The answer is "b", with the length of 1.

**Example 3:**

Input: s = "pwwkew"

Output: 3

Explanation: The answer is "wke", with the length of 3.

Notice that the answer must be a substring, "pwke" is a subsequence and not a substring.

**Constraints:**

0 <= s.length <= 5 \* 104

s consists of English letters, digits, symbols and spaces.

-------------------------------------------------------------------------------------------------------------

**Code**

-------------------------------------------------------------------------------------------------------------

package main

import (

    "bufio"

    "fmt"

    "os"

)

func main(){

    reader := bufio.NewReader(os.Stdin)

    fmt.Print("Enter String: ")

    str, \_, err := reader.ReadLine()

    if err!=nil{

        fmt.Println("Wrong Input")

    }

    //str := "dvdf"

    n,res := LongestSubarray(string(str))

    fmt.Println("String: ",res)

    fmt.Println("Length: ",n)

}

func LongestSubarray(s string)(int,string){

    rns := []rune(s)

    maxlen := 0

    st := 0

    end := 0

    var res string

    if len(rns)<2{

        return len(rns),string(rns)

    } else{

        for i:= 0; i<len(rns);i++{

            flg := false

            indx := 0

            for j:=st; j<end;j++{

                if rns[j] == rns[i]{

                    flg = true

                    indx = j

                    break

                }

            }

            if flg{

                if end-st > maxlen{

                    maxlen = end-st

                    res = string(rns[st:end])

                }

                st = indx+1

                end = i+1

            } else{

                end++

                if i == len(rns)-1{

                    if end-st > maxlen{

                        maxlen = end-st

                        res = string(rns[st:end])

                    }

                    return maxlen, res

                }

            }

        }

        return maxlen,res

    }

}

-------------------------------------------------------------------------------------------------------------