

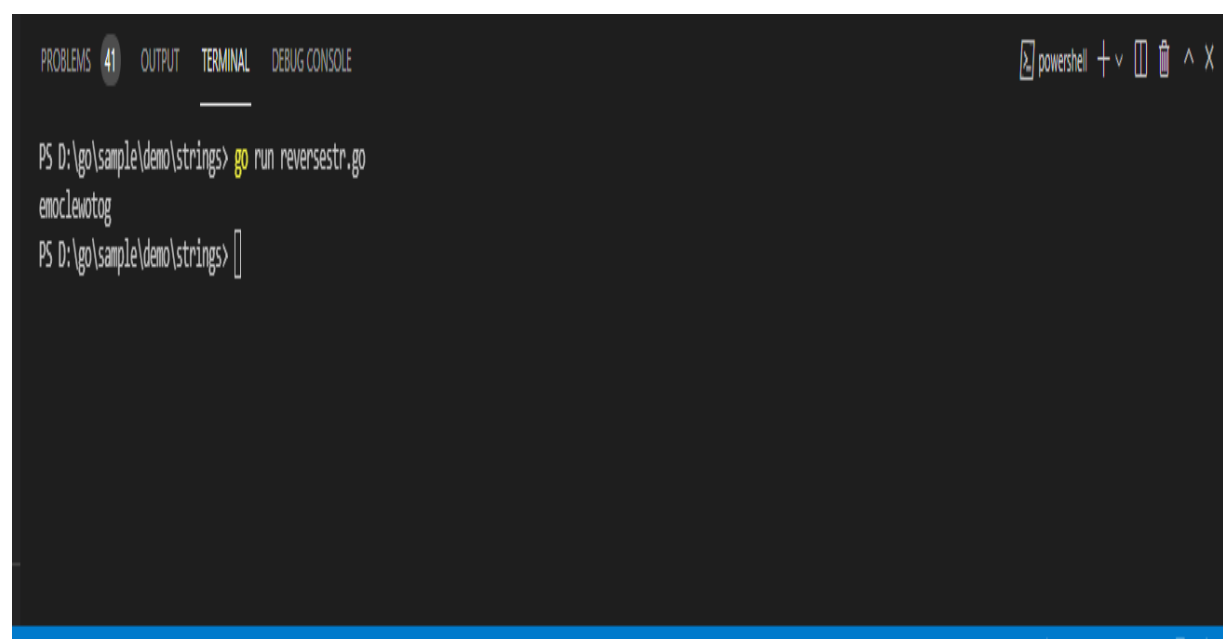
1. Go program to do wordwise reverse of a string.

```
package main

import (
    "fmt"
    "strings"
)

func main() {
    var result string
    var res string
    var str1 []string
    str := "welcome to go"
    str1 = strings.Split(str, " ")
    for _, v := range str1 {
        result = string(v) + result
    }
    for _, u := range result {
        res = string(u) + res
    }
    fmt.Println(res)
}
```

Output:



```
PROBLEMS 41 OUTPUT TERMINAL DEBUG CONSOLE
PS D:\go\sample\demo\strings> go run reversestr.go
emoclewotog
PS D:\go\sample\demo\strings>
```

2. Go program to print average of numbers using array.

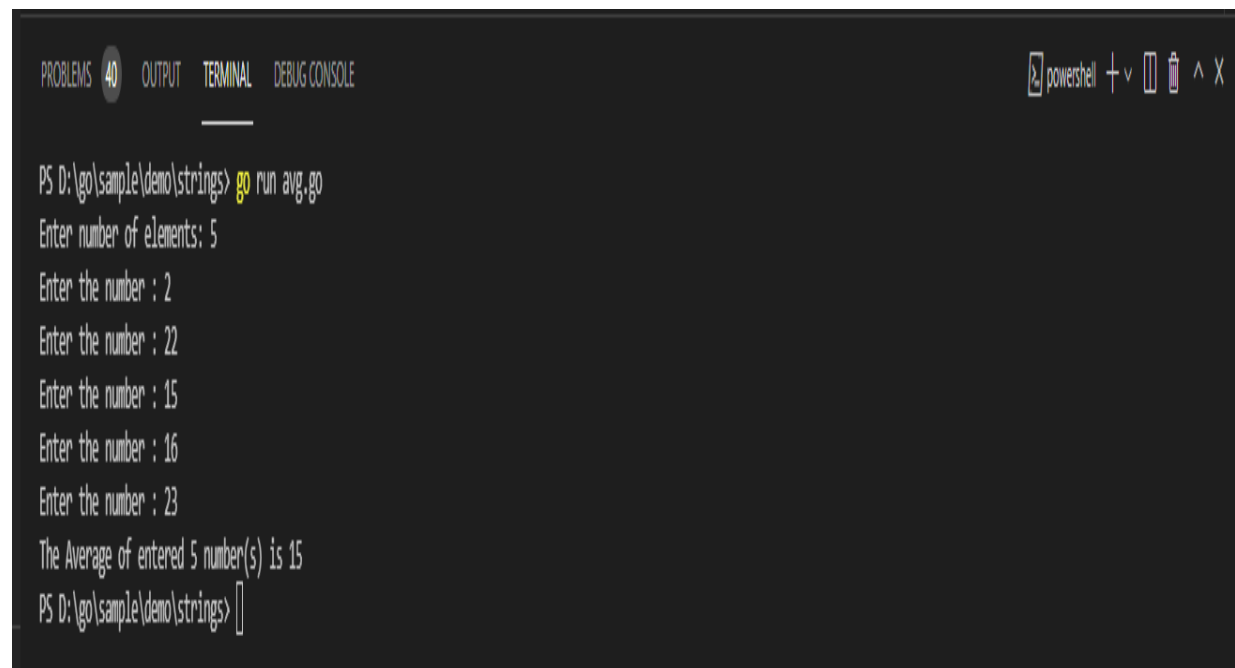
```
package main

import "fmt"

func main() {
    var num [100]int
    var temp, sum, avg int
    fmt.Print("Enter number of elements: ")
    fmt.Scanln(&temp)
    for i := 0; i < temp; i++ {
        fmt.Print("Enter the number : ")
        fmt.Scanln(&num[i])
        sum += num[i]
    }

    avg = sum / temp
    fmt.Printf("The Average of entered %d number(s) is %d", temp, avg)
}
```

Output:



The screenshot shows a terminal window with the following output:

```
PS D:\go\sample\demo\strings> go run avg.go
Enter number of elements: 5
Enter the number : 2
Enter the number : 22
Enter the number : 15
Enter the number : 16
Enter the number : 23
The Average of entered 5 number(s) is 15
PS D:\go\sample\demo\strings>
```

The terminal window has tabs for PROBLEMS (40), OUTPUT, TERMINAL, and DEBUG CONSOLE. The TERMINAL tab is active. The prompt is PS D:\go\sample\demo\strings>. The output shows the program running and calculating the average of 5 numbers: 2, 22, 15, 16, and 23, resulting in an average of 15.

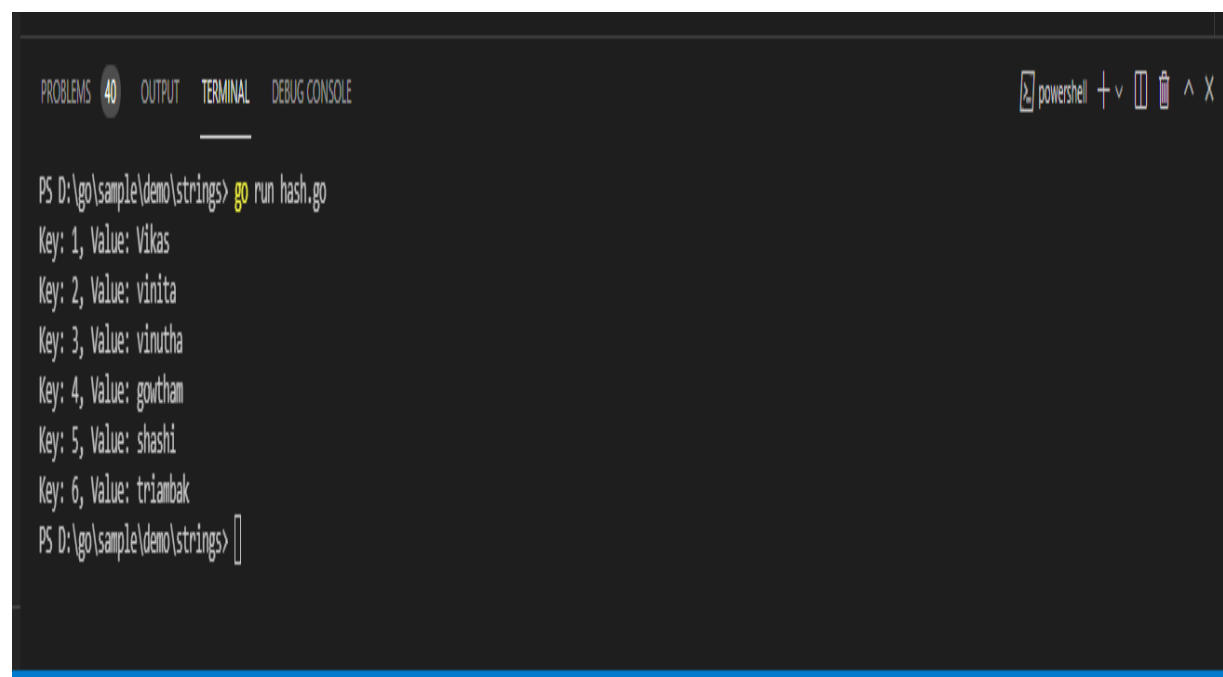
3. Hash table example using go lang.

```
package main

import (
    "fmt"
)

func main() {
    var country map[int]string
    country = make(map[int]string)
    country[1] = "Vikas"
    country[2] = "vinita"
    country[3] = "vinutha"
    country[4] = "gowtham"
    country[5] = "shashi"
    country[6] = "triambak"
    for i, j := range country {
        fmt.Printf("Key: %d, Value: %s\n", i, j)
    }
}
```

Output:



```
PROBLEMS 40 OUTPUT TERMINAL DEBUG CONSOLE
PS D:\go\sample\demo\strings> go run hash.go
Key: 1, Value: Vikas
Key: 2, Value: vinita
Key: 3, Value: vinutha
Key: 4, Value: gowtham
Key: 5, Value: shashi
Key: 6, Value: triambak
PS D:\go\sample\demo\strings>
```

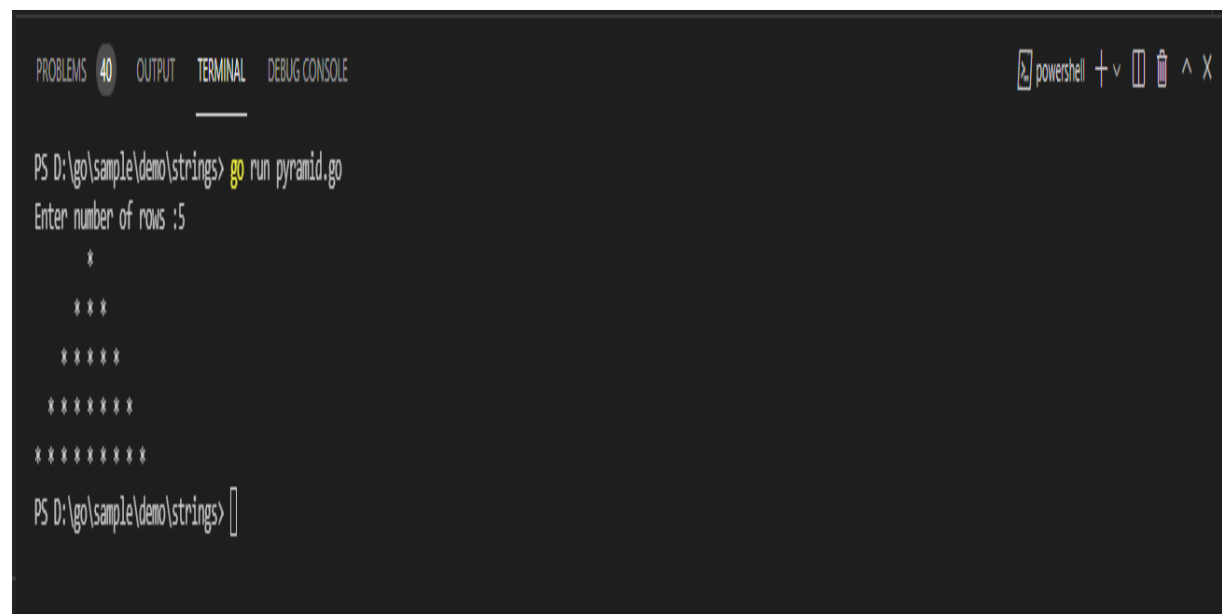
4. Go program to print full pyramid.

```
package main

import "fmt"

func main() {
    var rows int
    var k int = 0
    fmt.Print("Enter number of rows :")
    fmt.Scan(&rows)
    for i := 1; i <= rows; i++ {
        k = 0
        for space := 1; space <= rows-i; space++ {
            fmt.Print(" ")
        }
        for {
            fmt.Print("* ")
            k++
            if k == 2*i-1 {
                break
            }
        }
        fmt.Println("")
    }
}
```

Output:



The screenshot shows a Go IDE interface with a dark theme. At the top, there are tabs for 'PROBLEMS', '40', 'OUTPUT', 'TERMINAL', and 'DEBUG CONSOLE'. The 'TERMINAL' tab is active, displaying the command prompt output. The command 'go run pyramid.go' has been executed, and the program has prompted 'Enter number of rows :5'. The output shows a full pyramid of asterisks with 5 rows. The first row has 1 asterisk, the second has 3, the third has 5, the fourth has 7, and the fifth has 9. The prompt 'PS D:\go\sample\demo\strings>' is visible at the bottom.

```
PROBLEMS 40 OUTPUT TERMINAL DEBUG CONSOLE
PS D:\go\sample\demo\strings> go run pyramid.go
Enter number of rows :5
  *
 * *
* * *
* * * *
* * * * *
* * * * *
* * * * *
PS D:\go\sample\demo\strings>
```

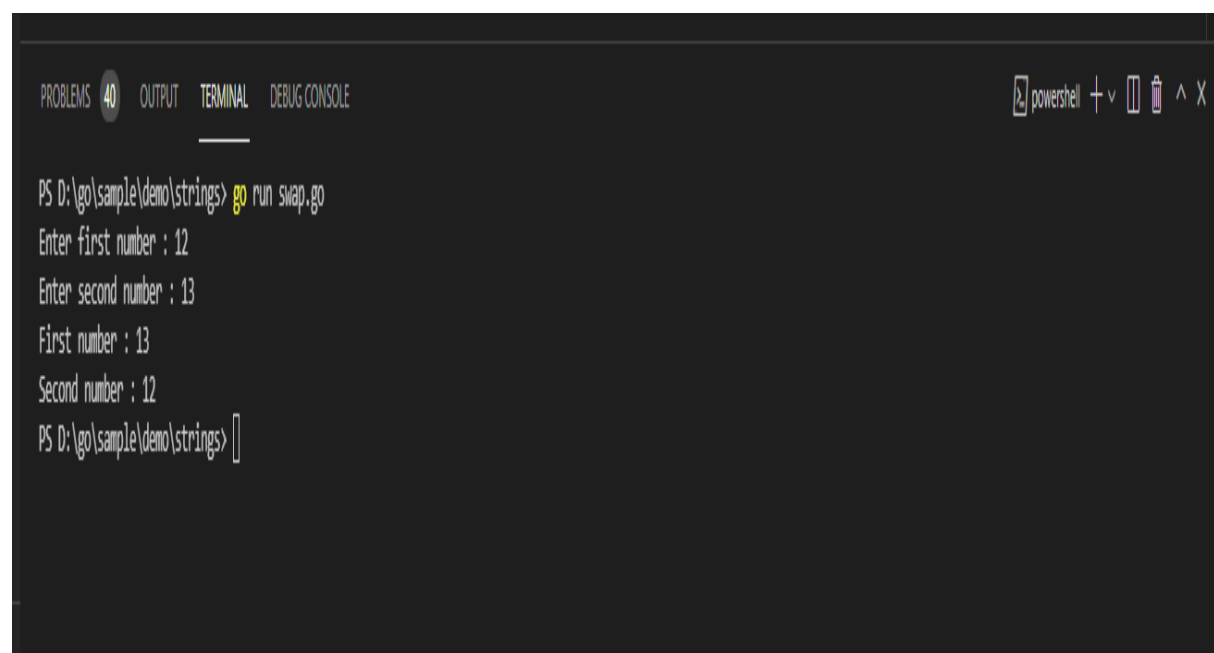
5. Go program to swap 2 numbers without using temporary variable.

```
package main

import "fmt"

func main() {
    fmt.Print("Enter first number : ")
    var first int
    fmt.Scanln(&first)
    fmt.Print("Enter second number : ")
    var second int
    fmt.Scanln(&second)
    first = first - second
    second = first + second
    first = second - first
    fmt.Println("First number :", first)
    fmt.Println("Second number :", second)
}
```

Output:



The screenshot shows a PowerShell terminal window with the following content:

```
PS D:\go\sample\demo\strings> go run swap.go
Enter first number : 12
Enter second number : 13
First number : 13
Second number : 12
PS D:\go\sample\demo\strings>
```

The terminal window has a title bar with "powershell" and standard window controls. The top of the window shows tabs for "PROBLEMS", "OUTPUT", "TERMINAL", and "DEBUG CONSOLE", with "TERMINAL" being the active tab. The output of the Go program is displayed in the terminal, showing the input numbers 12 and 13, and the resulting swapped values 13 and 12.

6. Go program to find area of rectangle and square.

```
package main

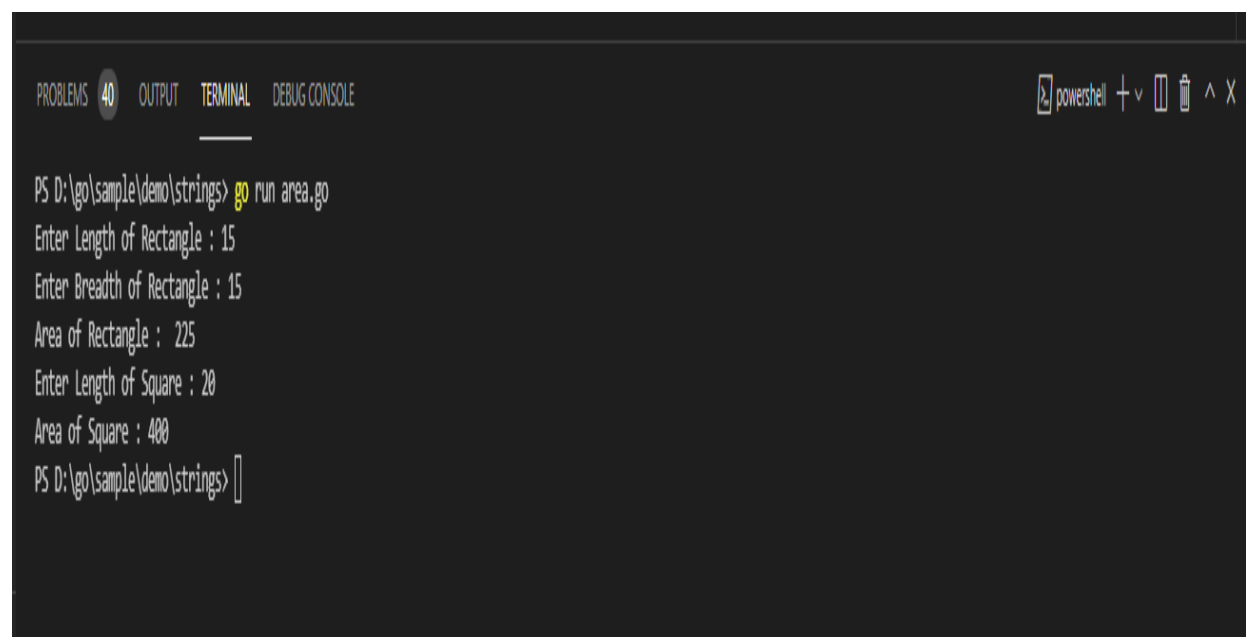
import "fmt"

var area int // Variable declare outside the main function

func main() {
    var l, b int //Declaration of multiple Variables
    fmt.Print("Enter Length of Rectangle : ")
    fmt.Scan(&l)
    fmt.Print("Enter Breadth of Rectangle : ")
    fmt.Scan(&b)
    area = l * b
    fmt.Println("Area of Rectangle : ", area) //move to new line

    fmt.Print("Enter Length of Square : ")
    fmt.Scan(&l)
    area = l * l
    fmt.Print("Area of Square : ", area)
}
```

Output:

A screenshot of a terminal window showing the execution of a Go program. The terminal has a dark background with light-colored text. At the top, there are tabs for 'PROBLEMS', 'OUTPUT', 'TERMINAL', and 'DEBUG CONSOLE'. The 'TERMINAL' tab is active. The command prompt shows the user running 'go run area.go' in the directory 'D:\go\sample\demo\strings'. The program prompts for the length and breadth of a rectangle, then calculates and displays the area. It then prompts for the length of a square, calculates and displays the area. The terminal window also shows standard window controls (minimize, maximize, close) and a PowerShell icon in the title bar.

```
PS D:\go\sample\demo\strings> go run area.go
Enter Length of Rectangle : 15
Enter Breadth of Rectangle : 15
Area of Rectangle : 225
Enter Length of Square : 20
Area of Square : 400
PS D:\go\sample\demo\strings>
```

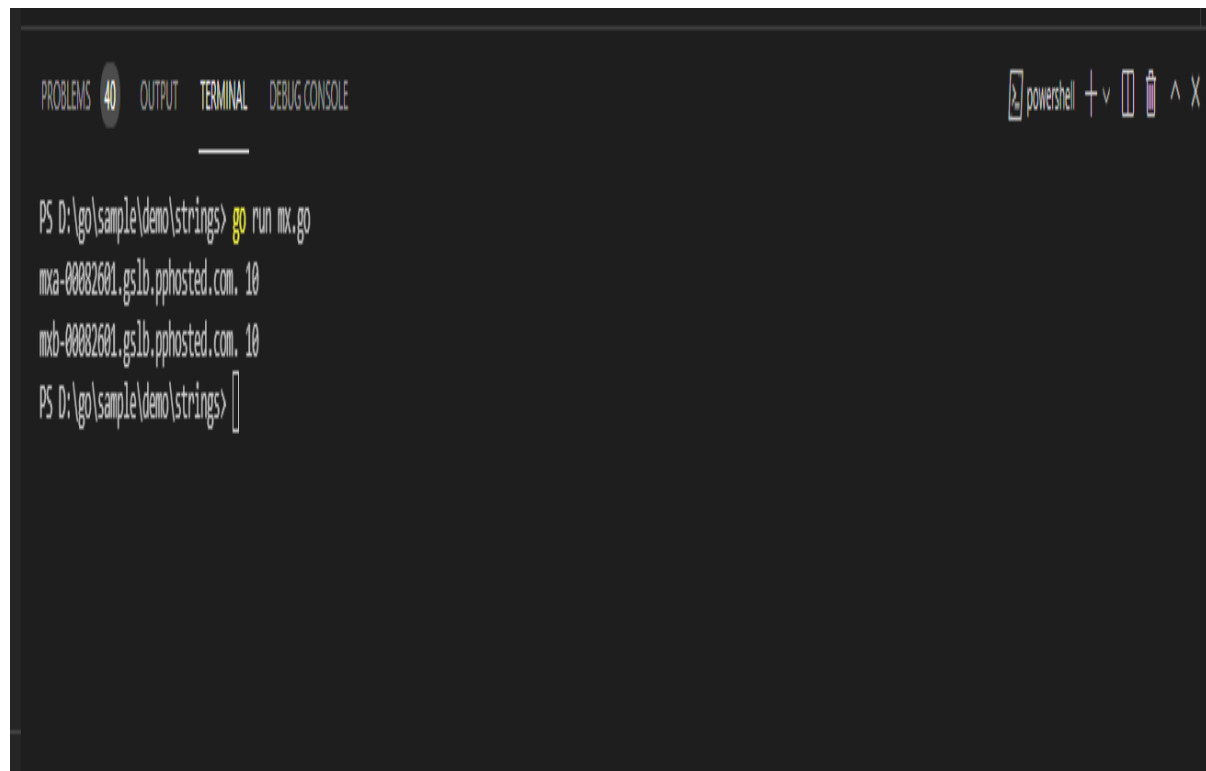
7. Go program to find mx records record of domain.

```
package main

import (
    "fmt"
    "net"
)

func main() {
    mxrecords, _ := net.LookupMX("instagram.com")
    for _, mx := range mxrecords {
        fmt.Println(mx.Host, mx.Pref)
    }
}
```

Output:



```
PROBLEMS 40 OUTPUT TERMINAL DEBUG CONSOLE powershell + v [ ] ^ X

PS D:\go\sample\demo\strings> go run mx.go
mx2-00002601.gslb.pphosted.com. 10
mx6-00002601.gslb.pphosted.com. 10
PS D:\go\sample\demo\strings> [ ]
```

8. Bubble sort using channel.

```
package main

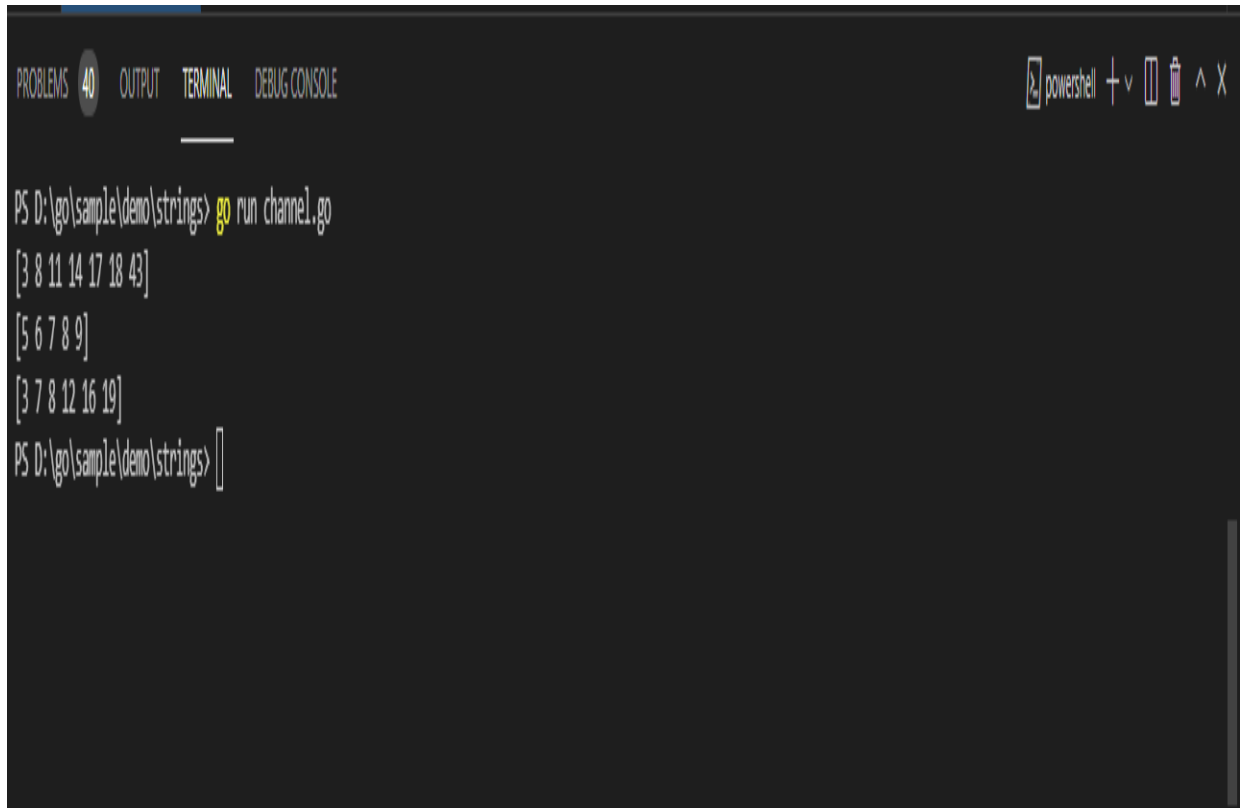
import "fmt"

func bubbleSort(array []int, c chan []int) []int {
    for i := 0; i < len(array)-1; i++ {
        for j := 0; j < len(array)-i-1; j++ {
            if array[j] > array[j+1] {
                array[j], array[j+1] = array[j+1], array[j]
            }
        }
    }
    c <- array
    return array
}

func printOutput(arr []int) []int {
    fmt.Println(arr)
    return arr
}

func main() {
    c := make(chan []int, 7)
    array := []int{11, 14, 3, 8, 18, 17, 43}
    array2 := []int{3, 12, 16, 7, 19, 8}
    array3 := []int{9, 8, 7, 6, 5}
    go bubbleSort(array, c)
    go bubbleSort(array2, c)
    go bubbleSort(array3, c)
    sorted := <-c
    sorted1 := <-c
    sorted2 := <-c
    fmt.Println(sorted)
    fmt.Println(sorted1)
    fmt.Println(sorted2)
}
```


Output:



The screenshot shows a PowerShell terminal window with a dark background. At the top, there is a tab bar with 'PROBLEMS' (containing a '40' badge), 'OUTPUT', 'TERMINAL', and 'DEBUG CONSOLE'. The 'TERMINAL' tab is active. In the top right corner, there are icons for 'powershell', a window management icon, a trash icon, and a close icon. The terminal content shows a PowerShell prompt 'PS D:\go\sample\demo\strings>' followed by the command 'go run channel.go'. The output consists of three lines of integers in square brackets: '[3 8 11 14 17 18 43]', '[5 6 7 8 9]', and '[3 7 8 12 16 19]'. The prompt returns to 'PS D:\go\sample\demo\strings>'.

```
PS D:\go\sample\demo\strings> go run channel.go
[3 8 11 14 17 18 43]
[5 6 7 8 9]
[3 7 8 12 16 19]
PS D:\go\sample\demo\strings>
```

9. Go program to find standard deviation.

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var num [10]float64
    var sum, mean, sd float64
    fmt.Println("***** Enter 10 elements *****")
    for i := 1; i <= 10; i++ {
        fmt.Printf("Enter %d element : ", i)
        fmt.Scan(&num[i-1])
        sum += num[i-1]
    }
    mean = sum / 10

    for j := 0; j < 10; j++ {
        sd += math.Pow(num[j]-mean, 2)
    }
    sd = math.Sqrt(sd / 10)

    fmt.Println("The Standard Deviation is : ", sd)
}
```

Output:

```
PS D:\go\sample\demo\strings> go run sd.go
***** Enter 10 elements *****
Enter 1 element : 2
Enter 2 element : 4
Enter 3 element : 6
Enter 4 element : 8
Enter 5 element : 10
Enter 6 element : 12
Enter 7 element : 14
Enter 8 element : 16
Enter 9 element : 18
Enter 10 element : 20
The Standard Deviation is : 5.744562646538829
PS D:\go\sample\demo\strings> |
```

10. Go program for writing a json file.

```
package main

import (
    "encoding/json"
    "io/ioutil"
)

type Salary struct {
    Basic, HRA, TA float64
}

type Employee struct {
    FirstName, LastName, Email string
    Age                       int
    MonthlySalary             []Salary
}

func main() {
    data := Employee{
        FirstName: "Vikas",
        LastName:  "S",
        Email:     "vikas@gmail.com",
        Age:       25,
        MonthlySalary: []Salary{
            Salary{
                Basic: 115000.00,
                HRA:   15000.00,
                TA:    12000.00,
            },
            Salary{
                Basic: 16000.00,
                HRA:   25000.00,
                TA:    21000.00,
            },
            Salary{
                Basic: 17000.00,
                HRA:   50000.00,
                TA:    22000.00,
            },
        },
    }
    file, _ := json.MarshalIndent(data, "", " ")
    _ = ioutil.WriteFile("test.json", file, 0644)
}
```

Output:

```
strings > {} testjson > ...
1  {}
2  "FirstName": "Vikas",
3  "LastName": "S",
4  "Email": "vikas@gmail.com",
5  "Age": 25,
6  "MonthlySalary": [
7  {
8    "Basic": 115000,
9    "HRA": 15000,
10   "TA": 12000
11  },
12  {
13    "Basic": 16000,
14    "HRA": 25000,
15    "TA": 21000
16  },
17  {
18    "Basic": 170000,
19    "HRA": 50000,
20    "TA": 22000
21  }
22  ]
23  {}

PROBLEMS 43 OUTPUT TERMINAL DEBUG CONSOLE
PS D:\go\sample\demo\strings> go run json.go
PS D:\go\sample\demo\strings> {}
```