LABORATORY MANUAL

INTRODUCTION TO GO PROGRAMMING LAB MANUAL

Department of Software & Application Development

S.D.M COLLEGE

UJIRE

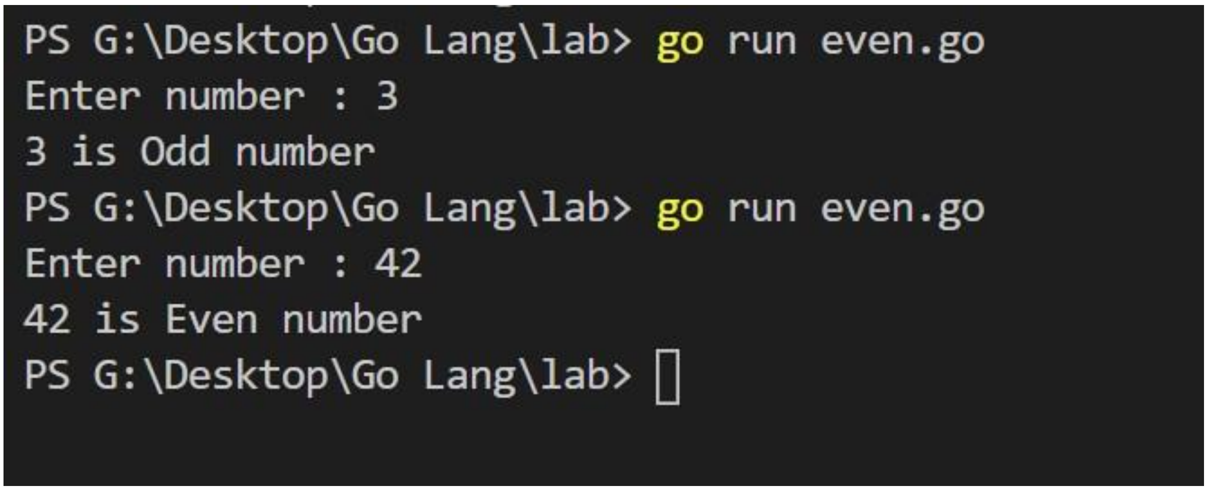
Prepared By:
Mr. Sammed jain
Assistant Professor
Dept of S&AD B.Voc
S.D.M College
Ujire

INDEX

1. GO Program to Check Whether a Number is Even or Odd.

```
package main
import "fmt"
func main() {
fmt.Print("Enter number : ")
var n int
fmt.Scanln(&n)
if n%2 == 0 {
fmt.Println(n, "is Even number")
} else {
fmt.Println(n, "is Odd number")
}
}
```

Output :

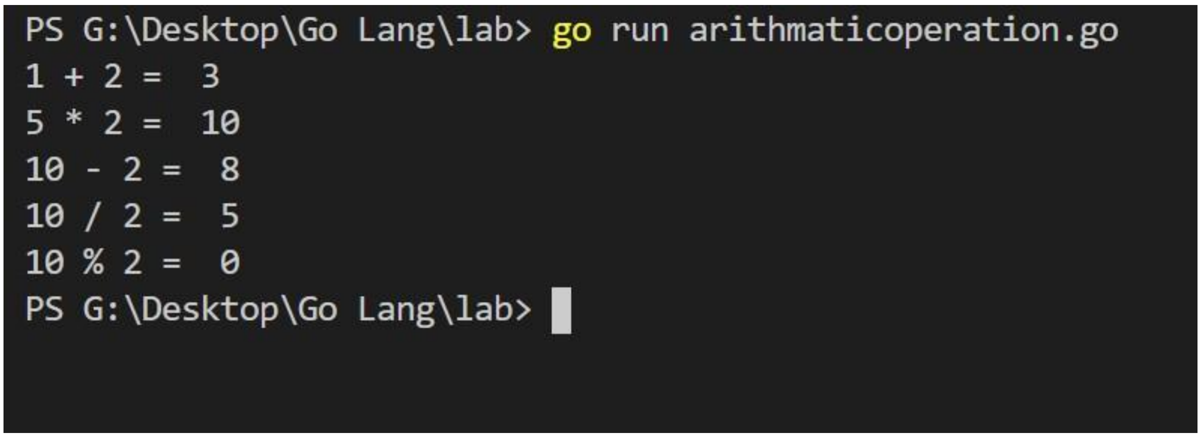


```
PS G:\Desktop\Go Lang\lab> go run even.go
Enter number : 3
3 is Odd number
PS G:\Desktop\Go Lang\lab> go run even.go
Enter number : 42
42 is Even number
PS G:\Desktop\Go Lang\lab> 
```

2. **GO program to display of standard arithmetic operators with 2 integer values.**

```
package main
import "fmt"
func main() {
fmt.Println("1 + 2 = ", 1+2)
fmt.Println("5 * 2 = ", 5*2)
fmt.Println("10 - 2 = ", 10-2)
fmt.Println("10 / 2 = ", 10/2)
fmt.Println("10 % 2 = ", 10%2)
}
```

Output :



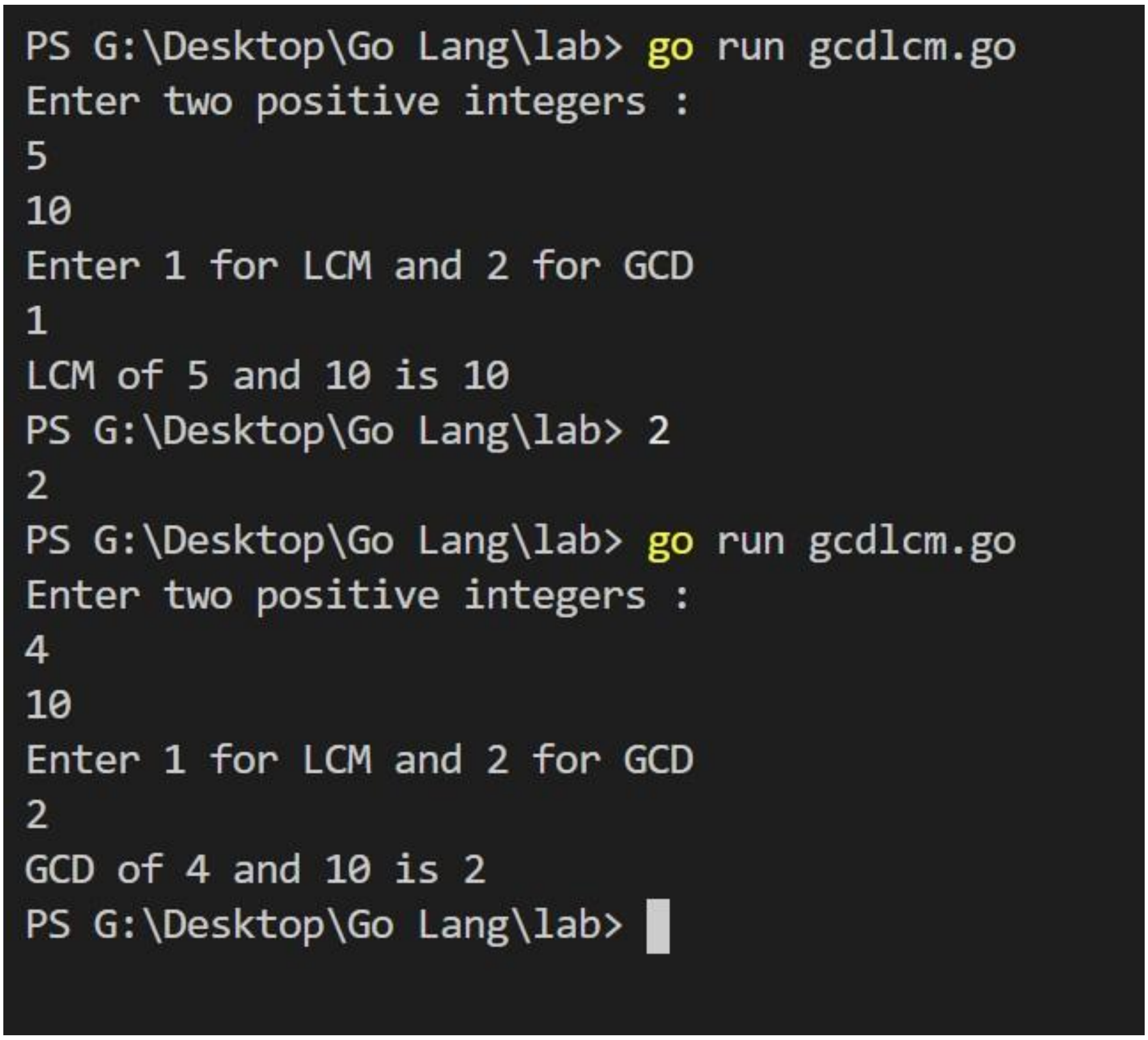
```
PS G:\Desktop\Go Lang\lab> go run arithmaticoperation.go
1 + 2 =  3
5 * 2 =  10
10 - 2 =  8
10 / 2 =  5
10 % 2 =  0
PS G:\Desktop\Go Lang\lab>
```

3. GO Program to Find LCM and GCD of given two numbers.

```
package main
import "fmt"
func lcm(temp1 int, temp2 int) {
var lcmnum int = 1
if temp1 > temp2 {
lcmnum = temp1
} else {
lcmnum = temp2
}
for {
if lcmnum%temp1 == 0 && lcmnum%temp2 == 0 {
fmt.Printf("LCM of %d and %d is %d", temp1, temp2, lcmnum)
break
}
lcmnum++
}
return
}
func gcd(temp1 int, temp2 int) {
var gcdnum int
for i := 1; i <= temp1 && i <= temp2; i++ {
if temp1%i == 0 && temp2%i == 0 {
gcdnum = i
}
}
fmt.Printf("GCD of %d and %d is %d", temp1, temp2, gcdnum)
return
}
func main() {
```

```
var n1, n2, action int

fmt.Println("Enter two positive integers : ")
fmt.Scanln(&n1)
fmt.Scanln(&n2)
fmt.Println("Enter 1 for LCM and 2 for GCD")
fmt.Scanln(&action)
switch action {
case  1:
lcm(n1, n2)
case 2:
gcd(n1, n2)
}
}
```

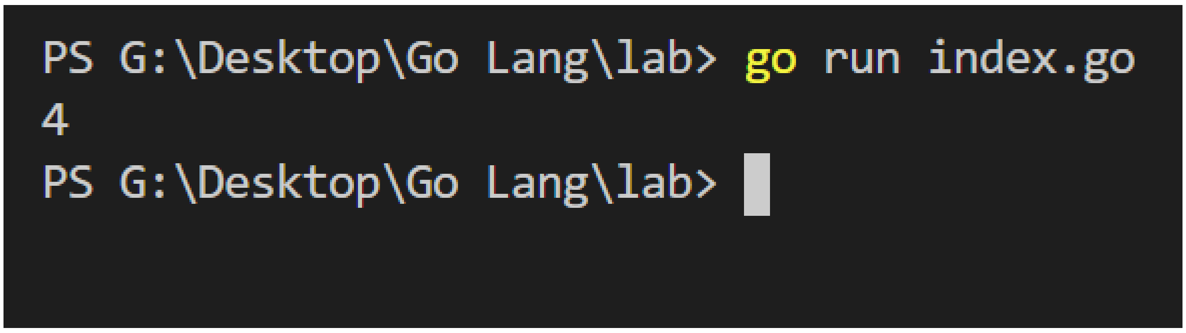
Output :


```
PS G:\Desktop\Go Lang\lab> go run gcdlcm.go
Enter two positive integers :
5
10
Enter 1 for LCM and 2 for GCD
1
LCM of 5 and 10 is 10
PS G:\Desktop\Go Lang\lab> 2
2
PS G:\Desktop\Go Lang\lab> go run gcdlcm.go
Enter two positive integers :
4
10
Enter 1 for LCM and 2 for GCD
2
GCD of 4 and 10 is 2
PS G:\Desktop\Go Lang\lab>
```

4. GO Program to find the index of first occurrence of a substring.

```
package main
import  (
"fmt"
"strings"
)
func main() {
str := "the cat in the hat"
i := strings.Index(str, "cat")
fmt.Println(i)
}
```

Output :

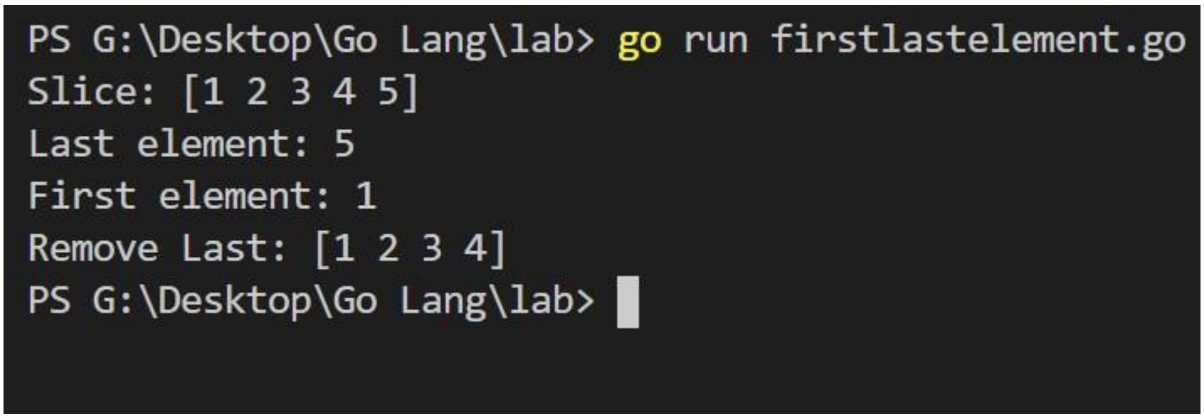


```
PS G:\Desktop\Go Lang\lab> go run index.go
4
PS G:\Desktop\Go Lang\lab>
```

5. GO Program to get first and last element of slice in Golang.

```
package main
import "fmt"
func main() {
intSlice := []int{1, 2, 3, 4, 5}
fmt.Println("slice: %v\n", intSlice)
last := intSlice[len(intSlice)-1]
fmt.Println("last element: %v\n", last)
first := intSlice[:0]
fmt.Println("first element: %d\n", first)
remove := intSlice[:len(intSlice)-1]
fmt.Println("remove last: %v\n", remove)
}
```

Output :

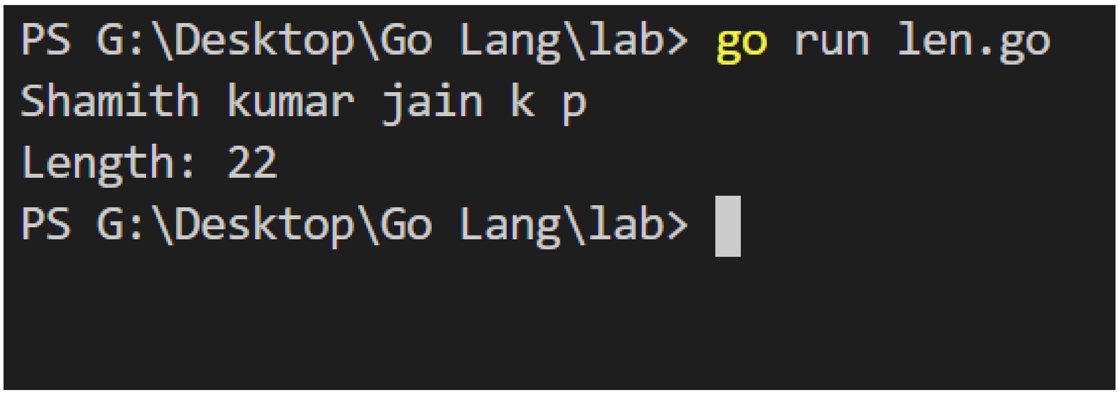


```
PS G:\Desktop\Go Lang\lab> go run firstlastelement.go
Slice: [1 2 3 4 5]
Last element: 5
First element: 1
Remove Last: [1 2 3 4]
PS G:\Desktop\Go Lang\lab>
```

6. GO Program to get total number of characters in a string.

```
package main
import "fmt"
func main() {
str := "Shamith kumar jain k p"
fmt.Println(str)
len := len(str)
fmt.Println(len)
}
```

Output :

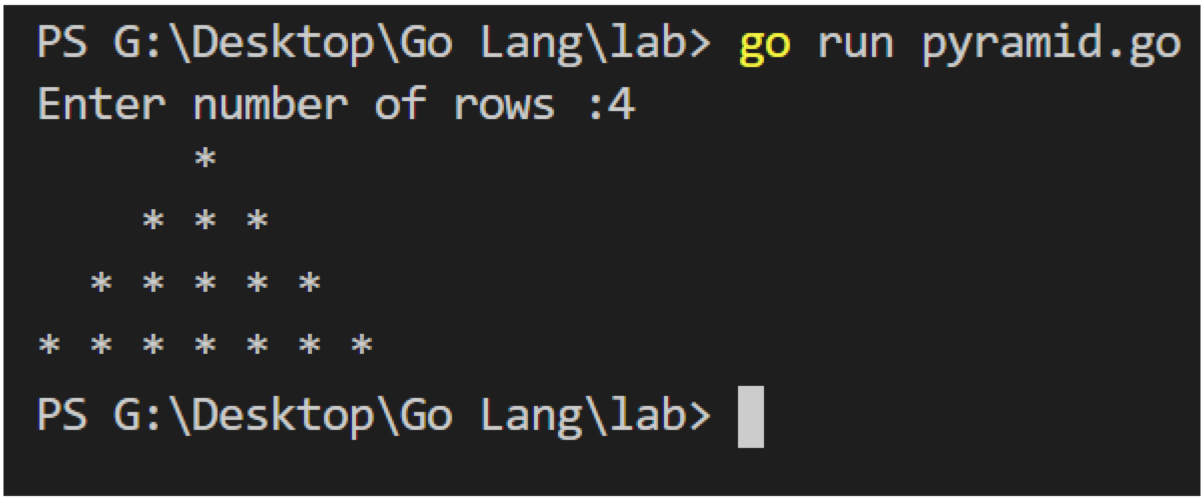


```
PS G:\Desktop\Go Lang\lab> go run len.go
Shamith kumar jain k p
Length: 22
PS G:\Desktop\Go Lang\lab>
```

7. GO Program to print full Pyramid using STAR.

```
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 :

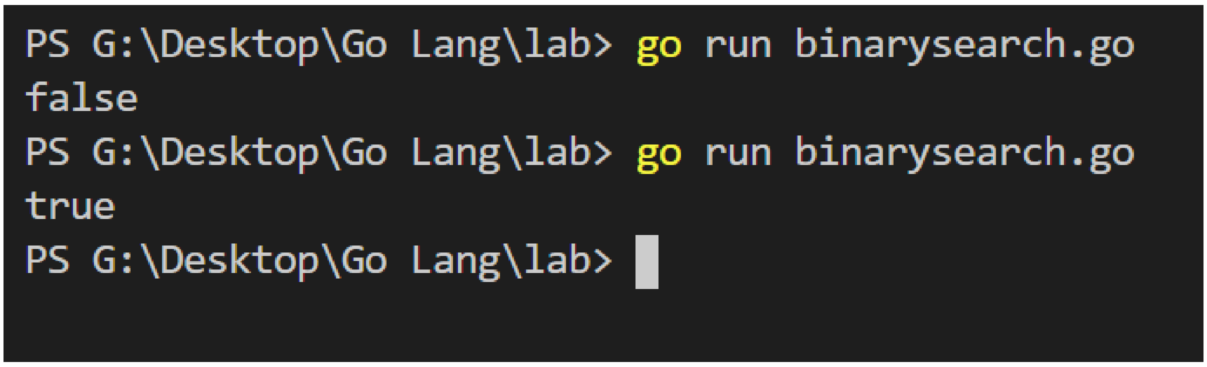


```
PS G:\Desktop\Go Lang\lab> go run pyramid.go
Enter number of rows :4
      *
    * * *
  * * * * *
* * * * * * *
PS G:\Desktop\Go Lang\lab>
```

8. GO program for implementation of Binary Search.

```
package main
import "fmt"
func binarySearch(needle int, haystack []int) bool {
low := 0
high := len(haystack)
for low <= high {
median := (low + high) / 2
if haystack[median] < needle {
low = median + 1
} else {
high = median - 1
}
}
if low == len(haystack) || haystack[low] != needle {
return false
}
return true
}
func main() {
items := []int{1, 2, 9, 20, 31, 45, 63, 70, 100}
fmt.Println(binarySearch(100, items))
}
```

Output :

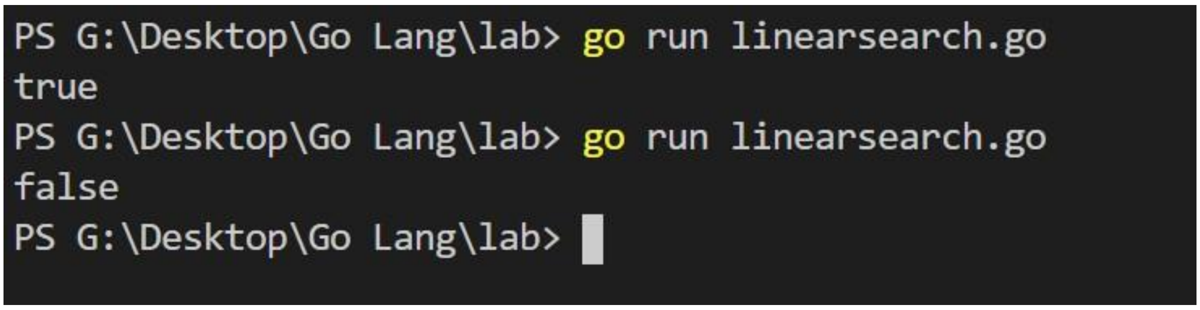


```
PS G:\Desktop\Go Lang\lab> go run binarysearch.go
false
PS G:\Desktop\Go Lang\lab> go run binarysearch.go
true
PS G:\Desktop\Go Lang\lab> 
```

9. GO program for implementation of Linear Search.

```
package main
import "fmt"
func linearSearch(datalist []int, key int) bool {
for _, item := range datalist {
if item == key {
return true
}
}
return false
}
func main() {
items := []int{95, 78, 56, 84, 25, 35, 15, 26}
fmt.Println(linearSearch(items, 96))
}
```

Output :

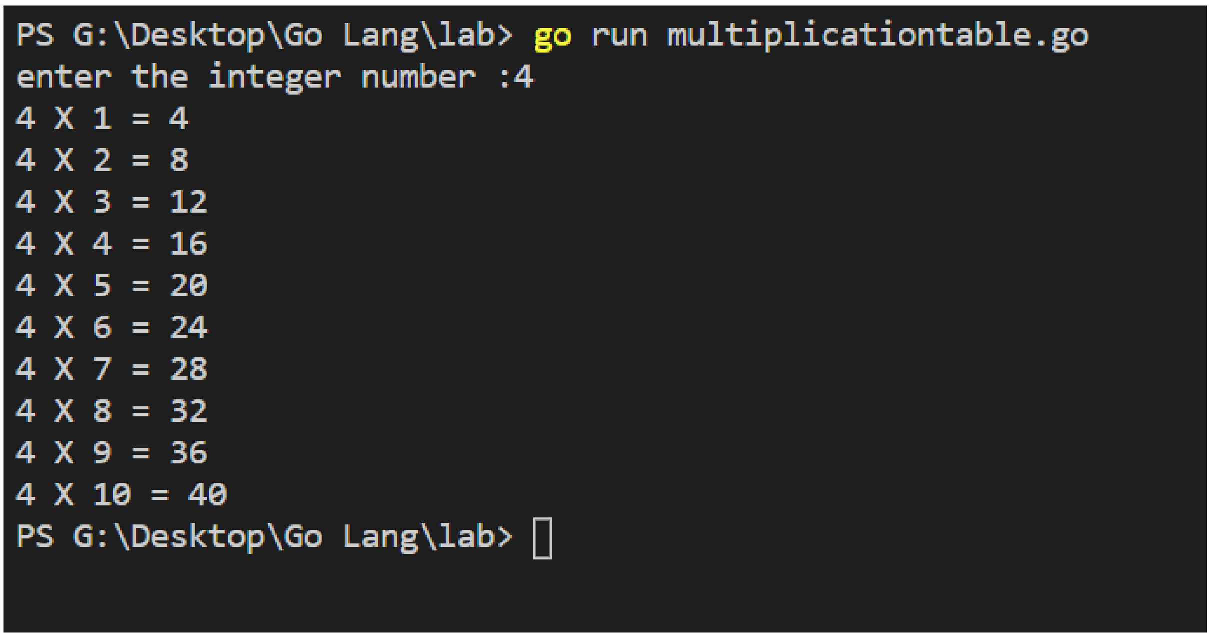


```
PS G:\Desktop\Go Lang\lab> go run linearsearch.go
true
PS G:\Desktop\Go Lang\lab> go run linearsearch.go
false
PS G:\Desktop\Go Lang\lab>
```

10. GO Program to Generate Multiplication Table.

```
package main
import "fmt"
func main() {
var n int
fmt.Print("enter the integer number :")
fmt.Scan(&n)
i := 1
for {
if i > 10 {
break
}
fmt.Println(n, "X", i, "=", n*i)
i++
}
}
```

Output :

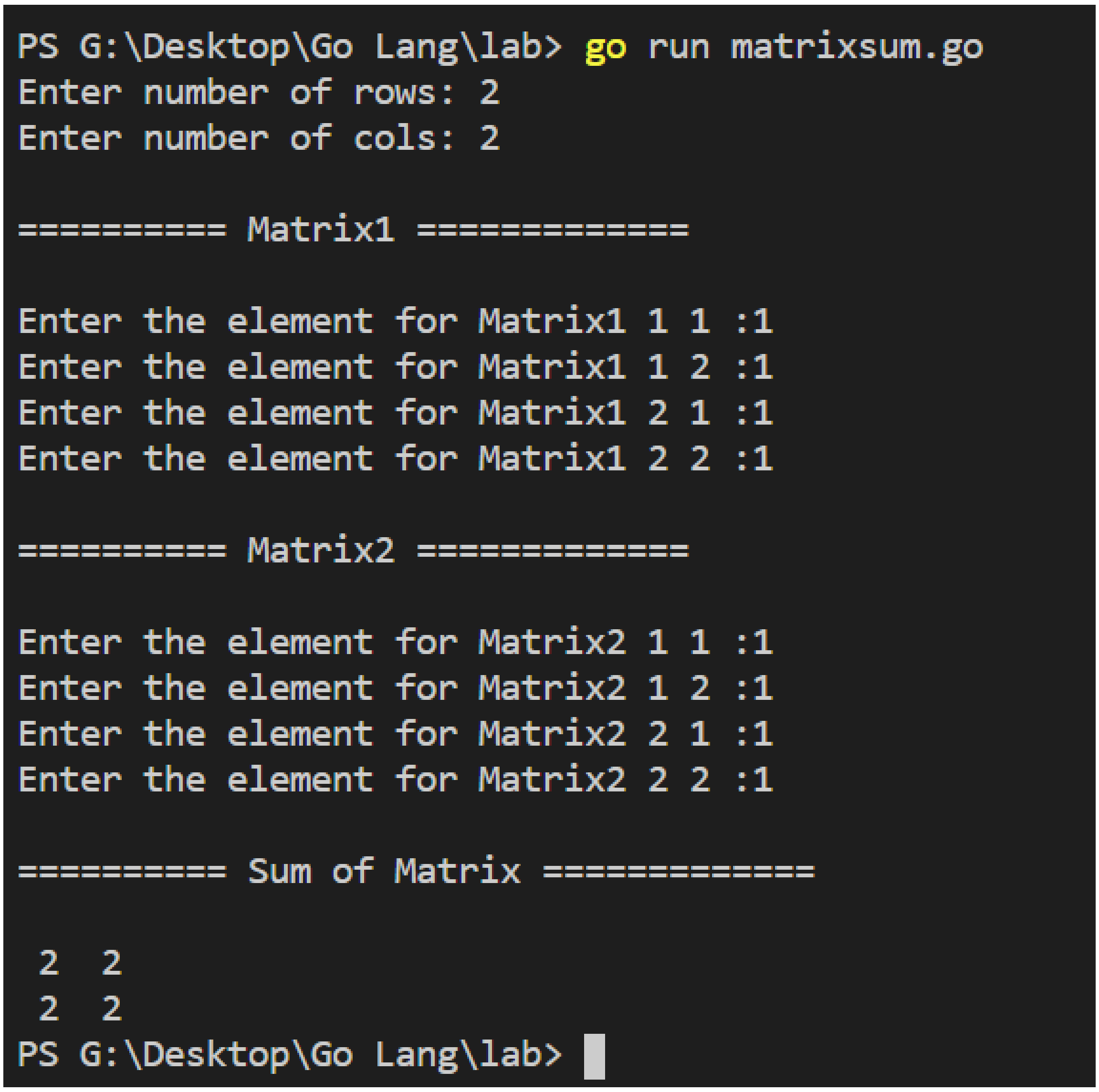


```
PS G:\Desktop\Go Lang\lab> go run multiplicationtable.go
enter the integer number :4
4 X 1 = 4
4 X 2 = 8
4 X 3 = 12
4 X 4 = 16
4 X 5 = 20
4 X 6 = 24
4 X 7 = 28
4 X 8 = 32
4 X 9 = 36
4 X 10 = 40
PS G:\Desktop\Go Lang\lab>
```

11. GO Program to Add Two Matrix Using Multi-dimensional Arrays.

```
package main
import "fmt"
func main() {
var matrix1 [100][100]int
var matrix2 [100][100]int
var sum  [100][100]int
var row, col int
fmt.Println("enter number of rows :")
fmt.Scanln(&row)
fmt.Println("enter number of cols :")
fmt.Scanln(&col)
fmt.Println()
fmt.Println("=================matrix1=================")
fmt.Println()
for i := 0; i < row; i++ {
for j := 0; j < col; j++ {
fmt.Printf("enter the element for matrix1 %d %d :", i+1, j+1)
fmt.Scanln(&matrix1[i][j])
}
}
fmt.Println()
fmt.Println("=================matrix2=================")
fmt.Println()
for i := 0; i < row; i++ {
for j := 0; j < col; j++ {
fmt.Printf("enter the element for matrix2 %d %d :", i+1, j+1)
fmt.Scanln(&matrix2[i][j])
}
}
for i := 0; i < row; i++ {
for j := 0; j < col; j++ {
sum[i][j] = matrix1[i][j] + matrix2[i][j]
}
}
fmt.Println()
fmt.Println("========== Sum of Matrix =============")
fmt.Println()
for i := 0; i < row; i++ {
for j := 0; j < col; j++ {
fmt.Printf(" %d ", sum[i][j])
if j == col-1 {
fmt.Println("")
}
}
}
}
```

Output :

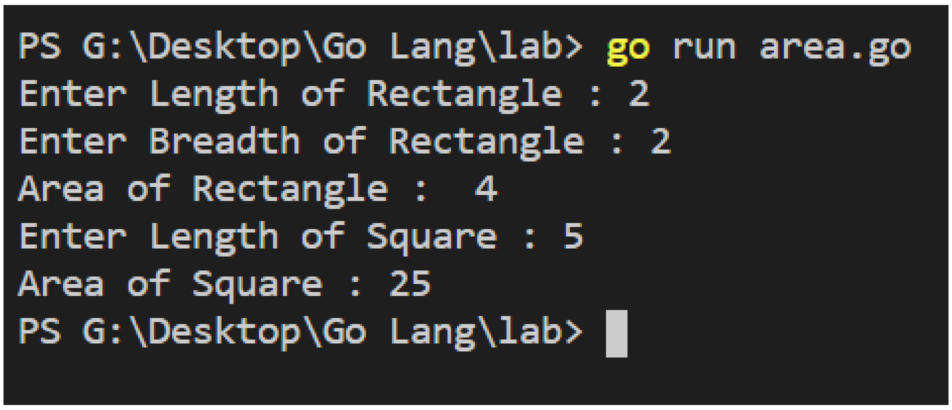


```
PS G:\Desktop\Go Lang\lab> go run matrixsum.go
Enter number of rows: 2
Enter number of cols: 2

========== Matrix1 =============

Enter the element for Matrix1 1 1 :1
Enter the element for Matrix1 1 2 :1
Enter the element for Matrix1 2 1 :1
Enter the element for Matrix1 2 2 :1

========== Matrix2 =============

Enter the element for Matrix2 1 1 :1
Enter the element for Matrix2 1 2 :1
Enter the element for Matrix2 2 1 :1
Enter the element for Matrix2 2 2 :1

========== Sum of Matrix =============

 2  2
 2  2
PS G:\Desktop\Go Lang\lab> 
```

12. GO Program to Calculate Area of Rectangle and Square.

```
package main
import "fmt"
var area int
func main() {
var l, b int
fmt.Println("enter length of rectangle :")
fmt.Scanln(&l)
fmt.Println("enter the breadth of rectangle :")
fmt.Scanln(&b)
area = l * b
fmt.Println("area of rectangle :", area)
fmt.Println("enter length of square :")
fmt.Scanln(&l)
area = l * l
fmt.Println("area of square :", area)
}
```

Output :

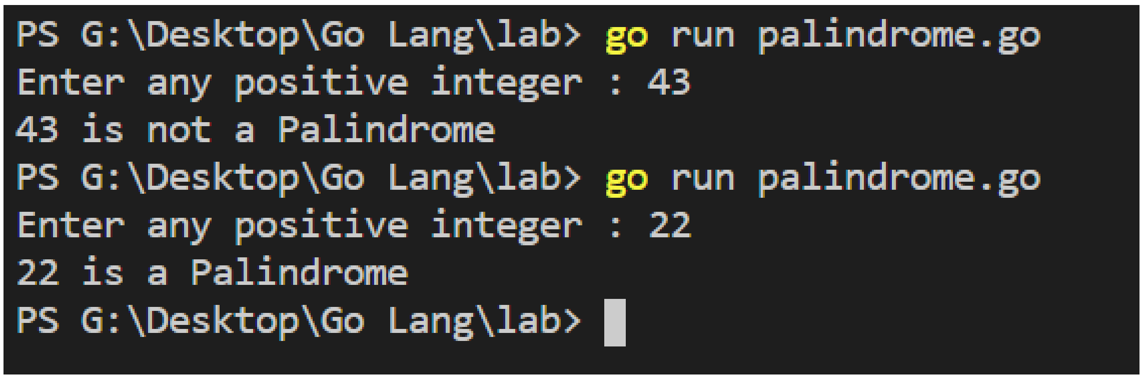


```
PS G:\Desktop\Go Lang\lab> go run area.go
Enter Length of Rectangle : 2
Enter Breadth of Rectangle : 2
Area of Rectangle :  4
Enter Length of Square : 5
Area of Square : 25
PS G:\Desktop\Go Lang\lab>
```

13. GO Program to Check Whether a Number is Palindrome or Not.

```
package main
import "fmt"
func main() {
var number, remainder, temp int
var reverse int = 0
fmt.Println("enter any positive integer :")
fmt.Scanln(&number)
temp = number
for {
remainder = number % 10
reverse = reverse*10 + remainder
number /= 10

if number == 0 {
break
}
}
if temp == reverse {
fmt.Println("%d is a palindrome", temp)
} else {
fmt.Println("%D is not a palindrome", temp)
}
}
```

Output :



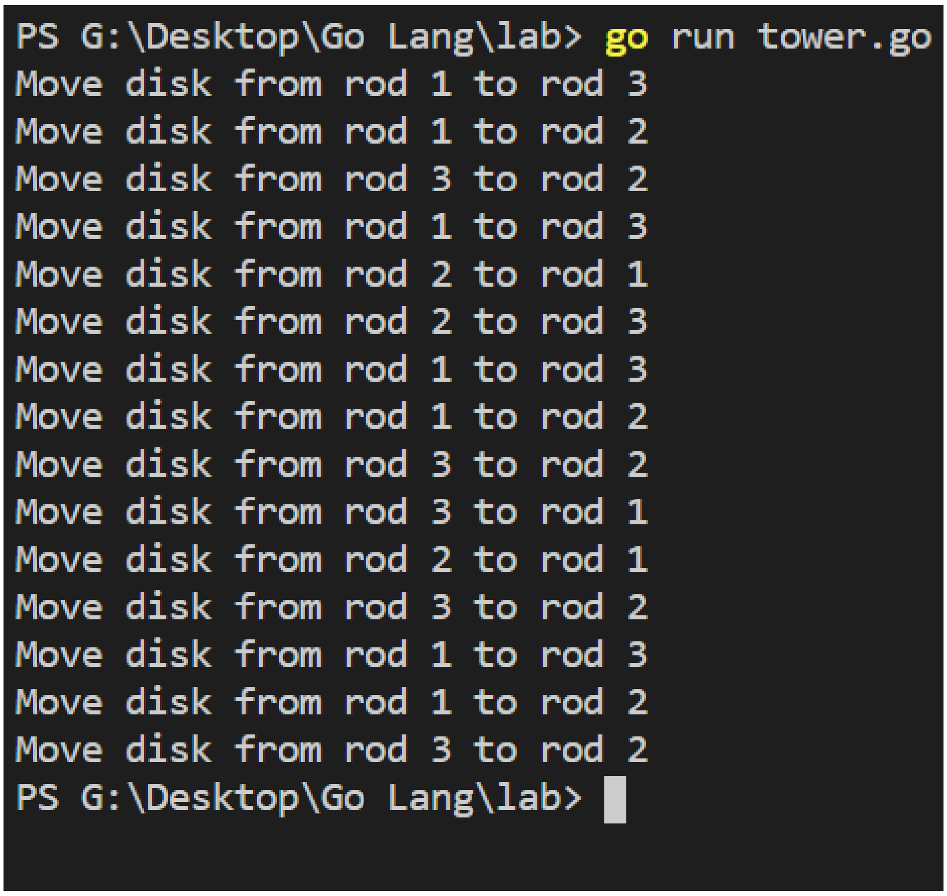
```
PS G:\Desktop\Go Lang\lab> go run palindrome.go
Enter any positive integer : 43
43 is not a Palindrome
PS G:\Desktop\Go Lang\lab> go run palindrome.go
Enter any positive integer : 22
22 is a Palindrome
PS G:\Desktop\Go Lang\lab>
```

14. GO program to implementation of Tower of Hanoi Algorithm.

```
package main

import "fmt"

type solver interface {
        play(int)
}

type towers struct {
}

func (t *towers) play(n int) {
        t.moveN(n, 1, 2, 3)
}
func (t *towers) moveN(n, from, to, via int) {
        if n > 0 {
                t.moveN(n-1, from, via, to)
                t.moveM(from, to)
                t.moveN(n-1, via, to, from)
        }
}
func (t *towers) moveM(from, to int) {
        fmt.Println("move disk from rod", from, "to rod", to)
}

func main() {
        var t solver
        t = new(towers)
        t.play(4)
```

}

Output :

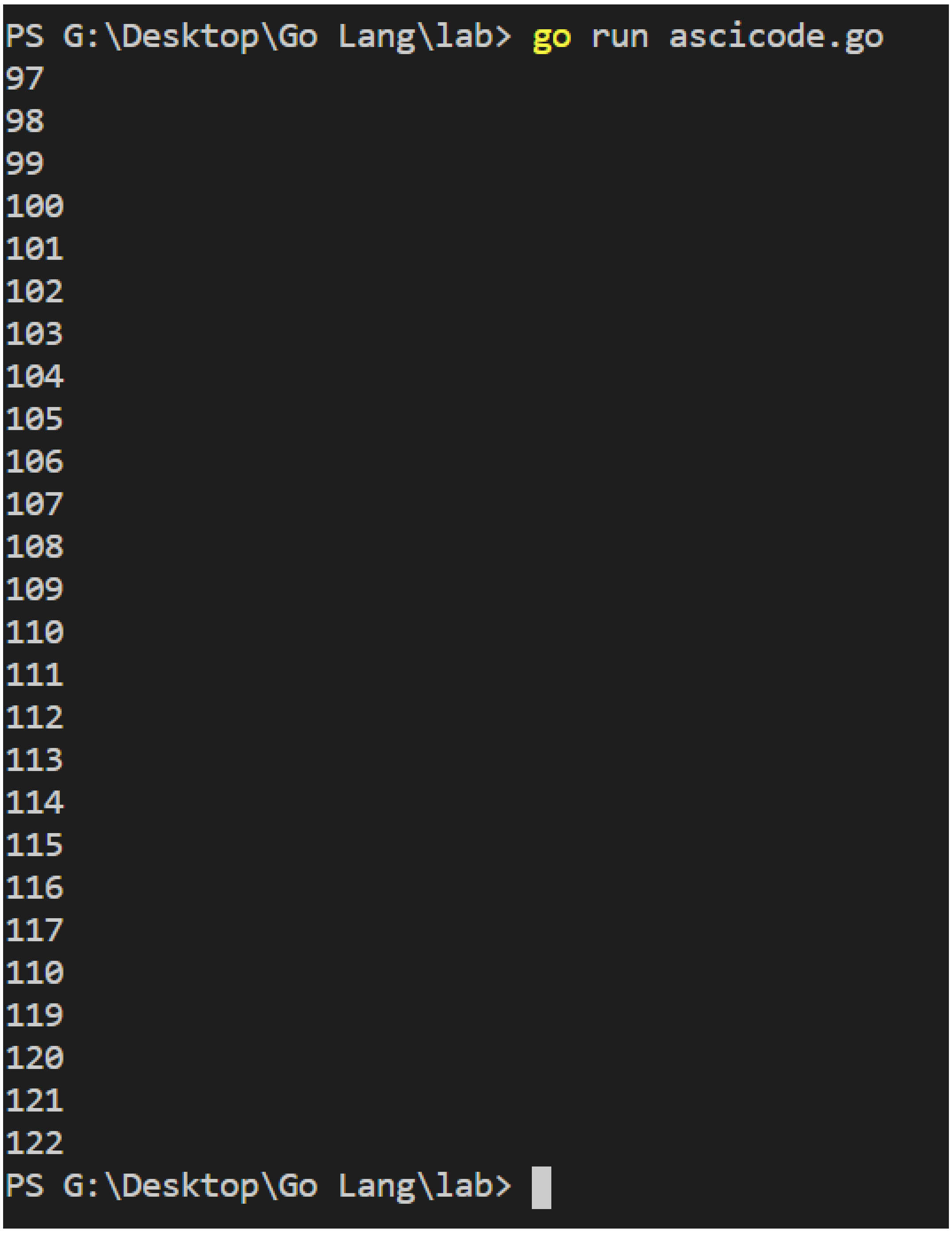


```
PS G:\Desktop\Go Lang\lab> go run tower.go
Move disk from rod 1 to rod 3
Move disk from rod 1 to rod 2
Move disk from rod 3 to rod 2
Move disk from rod 1 to rod 3
Move disk from rod 2 to rod 1
Move disk from rod 2 to rod 3
Move disk from rod 1 to rod 3
Move disk from rod 1 to rod 2
Move disk from rod 3 to rod 2
Move disk from rod 3 to rod 1
Move disk from rod 2 to rod 1
Move disk from rod 3 to rod 2
Move disk from rod 1 to rod 3
Move disk from rod 1 to rod 2
Move disk from rod 3 to rod 2
PS G:\Desktop\Go Lang\lab>
```

15. GO Program to print the ascii code for each letter in the alphabet.

```
package main

import "fmt"

func main() {
        var str = "abcdefghijklmnopqrstuvwxyz"

        for _, c := range str {
                fmt.Println(c)
        }
}
```

Output :



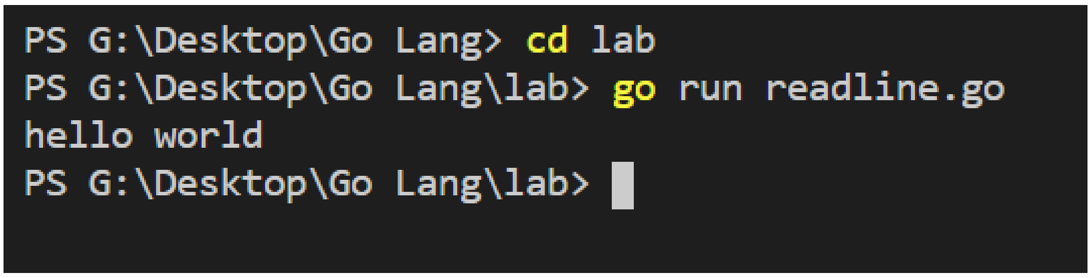
```
PS G:\Desktop\Go Lang\lab> go run ascicode.go
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
110
119
120
121
122
PS G:\Desktop\Go Lang\lab>
```

}

16.GO Program to read file line by line to string.

```
package main

import (
        "bufio"
        "fmt"
        "log"
        "os"
)

func main() {
        file, err := os.Open("kotlin.txt")

        if err != nil {
                log.Fatalf("failed opening file: %s", err)
        }
        scanner := bufio.NewScanner(file)
        scanner.Split(bufio.ScanLines)
        var txtlines []string

        for scanner.Scan() {
                txtlines = append(txtlines, scanner.Text())
        }

        file.Close()

        for _, eachline := range txtlines {
                fmt.Println(eachline)
        }
}
```

Output :

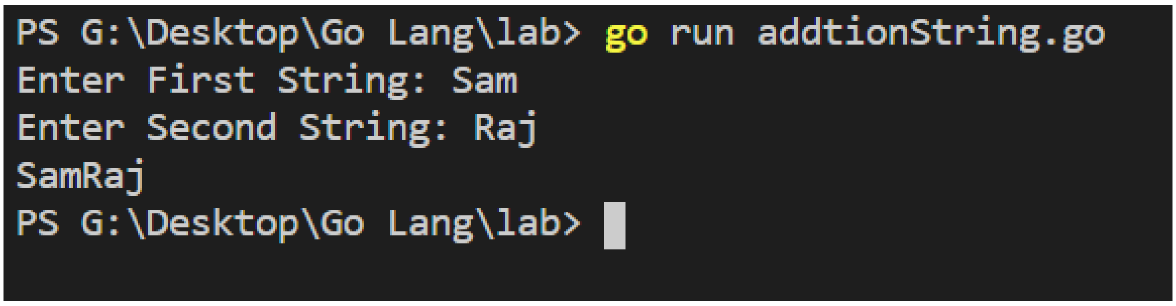


```
PS G:\Desktop\Go Lang> cd lab
PS G:\Desktop\Go Lang\lab> go run readline.go
hello world
PS G:\Desktop\Go Lang\lab> 
```

17. GO Program to take user input and addition of two strings.

```
package main

import "fmt"

func main() {
        fmt.Println("enter first string: ")
        var first string
        fmt.Scanln(&first)
        fmt.Println("enter second string: ")
        var second string
        fmt.Scanln(&second)
        fmt.Println(first + second)
}
```

Output :



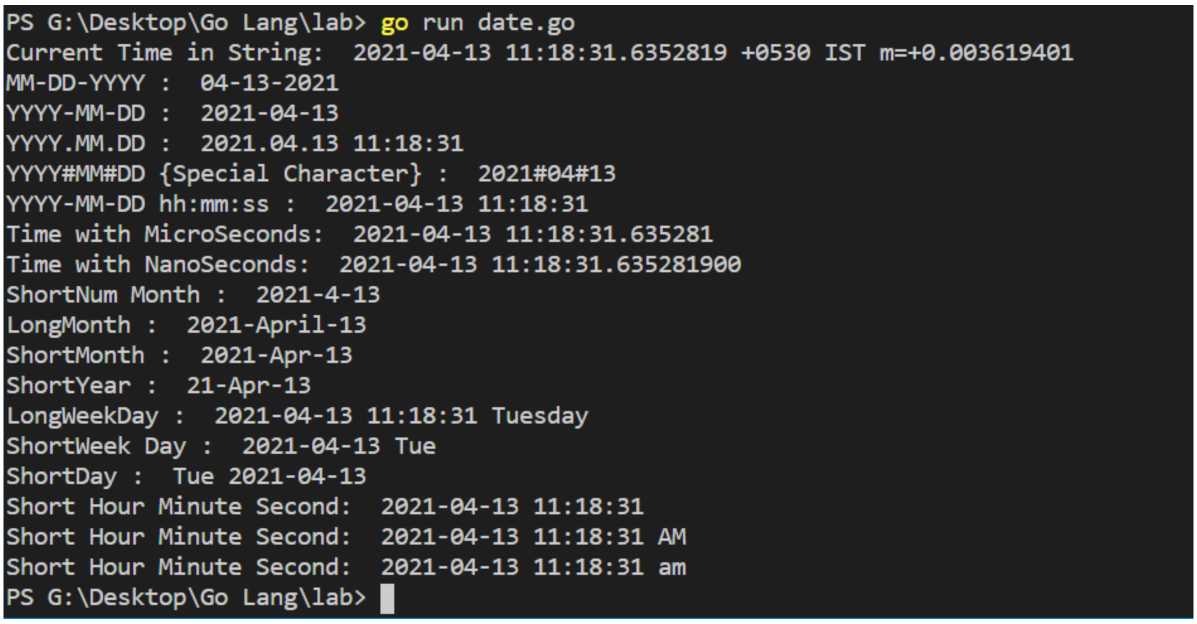
```
PS G:\Desktop\Go Lang\lab> go run addtionString.go
Enter First String: Sam
Enter Second String: Raj
SamRaj
PS G:\Desktop\Go Lang\lab>
```

18. GO Program to Get current date and time in various format in golang.

```go
package main

import (
	"fmt"
	"time"
)

func main() {
	currentTime := time.Now()

	fmt.Println("Current Time in String: ", currentTime.String())

	fmt.Println("MM-DD-YYYY : ", currentTime.Format("01-02-2006"))

	fmt.Println("YYYY-MM-DD : ", currentTime.Format("2006-01-02"))

	fmt.Println("YYYY.MM.DD : ", currentTime.Format("2006.01.02 15:04:05"))

	fmt.Println("YYYY#MM#DD {Special Character} : ",
	currentTime.Format("2006#01#02"))

	fmt.Println("YYYY-MM-DD hh:mm:ss : ", currentTime.Format("2006-01-02
	15:04:05"))

	fmt.Println("Time with MicroSeconds: ", currentTime.Format("2006-01-02
	15:04:05.000000"))
```

```
        fmt.Println("Time with NanoSeconds: ", currentTime.Format("2006-01-02
        15:04:05.000000000"))

        fmt.Println("ShortNum Month : ", currentTime.Format("2006-1-02"))

        fmt.Println("LongMonth : ", currentTime.Format("2006-January-02"))

        fmt.Println("ShortMonth : ", currentTime.Format("2006-Jan-02"))

        fmt.Println("ShortYear : ", currentTime.Format("06-Jan-02"))

        fmt.Println("LongWeekDay : ", currentTime.Format("2006-01-02 15:04:05
        Monday"))

        fmt.Println("ShortWeek Day : ", currentTime.Format("2006-01-02 Mon"))

        fmt.Println("ShortDay : ", currentTime.Format("Mon 2006-01-2"))

        fmt.Println("Short Hour Minute Second: ", currentTime.Format("2006-01-02
        3:4:5"))

        fmt.Println("Short Hour Minute Second: ", currentTime.Format("2006-01-02
        3:4:5 PM"))

        fmt.Println("Short Hour Minute Second: ", currentTime.Format("2006-01-02
        3:4:5 pm"))
}
```

Output :

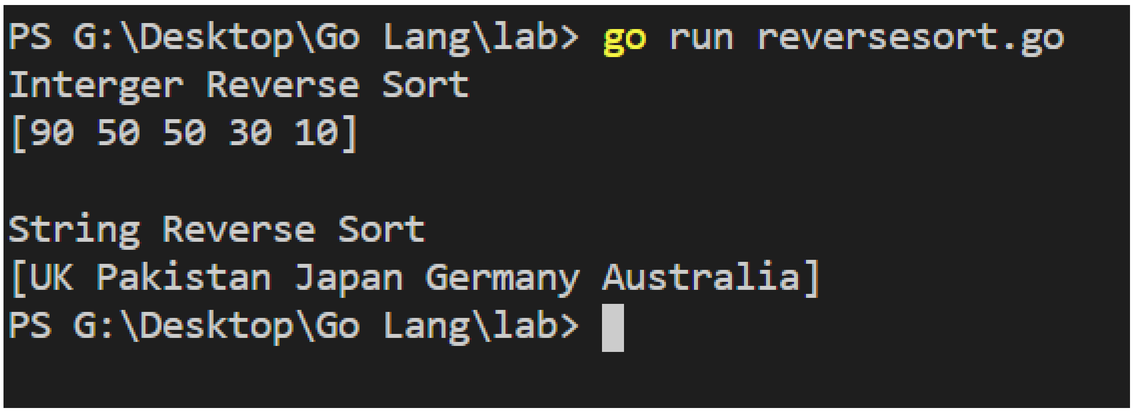


```
PS G:\Desktop\Go Lang\lab> go run date.go
Current Time in String:  2021-04-13 11:18:31.6352819 +0530 IST m=+0.003619401
MM-DD-YYYY :  04-13-2021
YYYY-MM-DD :  2021-04-13
YYYY.MM.DD :  2021.04.13 11:18:31
YYYY#MM#DD {Special Character} :  2021#04#13
YYYY-MM-DD hh:mm:ss :  2021-04-13 11:18:31
Time with MicroSeconds:  2021-04-13 11:18:31.635281
Time with NanoSeconds:  2021-04-13 11:18:31.635281900
ShortNum Month :  2021-4-13
LongMonth :  2021-April-13
ShortMonth :  2021-Apr-13
ShortYear :  21-Apr-13
LongWeekDay :  2021-04-13 11:18:31 Tuesday
ShortWeek Day :  2021-04-13 Tue
ShortDay :  Tue 2021-04-13
Short Hour Minute Second:  2021-04-13 11:18:31
Short Hour Minute Second:  2021-04-13 11:18:31 AM
Short Hour Minute Second:  2021-04-13 11:18:31 am
PS G:\Desktop\Go Lang\lab>
```

19. GO program with example of Array Reverse Sort Functions for integer and strings.

```
package main

import (
	"fmt"
	"sort"
)

func main() {
	fmt.Println("integer reverse sort")
	num := []int{50, 40, 60, 9, 80}
	sort.Sort(sort.Reverse(sort.IntSlice(num)))
	fmt.Println(num)

	fmt.Println()

	fmt.Println("string reverse sort")
	text := []string{"Japan", "UK", "Germany", "Australia", "Pakistan"}
	sort.Sort(sort.Reverse(sort.StringSlice(text)))
	fmt.Println(text)
}
```

Output :

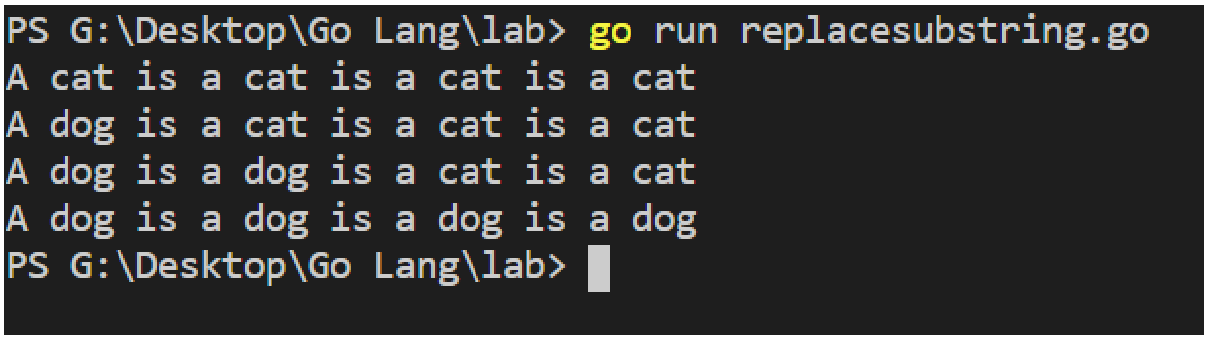


```
PS G:\Desktop\Go Lang\lab> go run reversesort.go
Interger Reverse Sort
[90 50 50 30 10]

String Reverse Sort
[UK Pakistan Japan Germany Australia]
PS G:\Desktop\Go Lang\lab>
```

20. GO Program to replace substrings in a string.

```
package main

import (
	"fmt"
	"strings"
)

func main() {
	str1 := "A cat is a cat is a cat is a cat"
	fmt.Println(str1)

	str2 := strings.Replace(str1, "cat", "dog", 1)
	fmt.Println(str2)

	str3 := strings.Replace(str1, "cat", "dog", 2)
	fmt.Println(str3)

	str4 := strings.Replace(str1, "cat", "dog", -1)
	fmt.Println(str4)
}
```

Output :



```
PS G:\Desktop\Go Lang\lab> go run replacesubstring.go
A cat is a cat is a cat is a cat
A dog is a cat is a cat is a cat
A dog is a dog is a cat is a cat
A dog is a dog is a dog is a dog
PS G:\Desktop\Go Lang\lab> 
```