q2 .Write a program in GO language to create an interface and display its values with the help of type assertion.

```
package main

import "fmt"
type tank interface {
	Tarea() float64
	Volume() float64
}

type myvalue struct {
	radius float64
	height float64
}

func (m myvalue) Tarea() float64 {

	return 2*m.radius*m.height +
		2*3.14*m.radius*m.radius
}

func (m myvalue) Volume() float64 {

	return 3.14 * m.radius * m.radius * m.height
}

func main() {
	var t tank
	t = myvalue{10, 14}
	fmt.Println("Area of tank :", t.Tarea())
```

```
	fmt.Println("Volume of tank:", t.Volume())
}
```

Q3.Write a program in GO language to check whether the accepted number is two digit or not.

```
package main

import (
	"fmt"
	"unicode"
)

func main() {

	val := []rune{'g', 'E', '3', 'K', '1'}

	for i := 0; i < len(val); i++ {

		if unicode.IsDigit(val[i]) == true {

			fmt.Println("It is a decimal digit")

		} else {

			fmt.Println("It is not a decimal digit")
		}
	}
}
```

q4.Write a program in GO language to swap two numbers using call by reference concept

```
package main
```

```
import "fmt"

func main() {
  /* local variable definition */
  var a int = 100
  var b int = 200

  fmt.Printf("Before swap, value of a : %d\n", a )
  fmt.Printf("Before swap, value of b : %d\n", b )

  swap(&a, &b)

  fmt.Printf("After swap, value of a : %d\n", a )
  fmt.Printf("After swap, value of b : %d\n", b )
}
func swap(x *int, y *int) {
  var temp int
  temp = *x
  *x = *y
  *y = temp
}
```

Write a program in GO language to print sum of all even and oddnumbers separately between 1 to 100.

```
package main

import "fmt"
```

```
func main() {

   var evnum, i,sum int

   fmt.Print("Enter the Number to Print Even's = ")
   fmt.Scanln(&evnum)

   fmt.Println("Even Numbers from 1 to ", evnum, " are = ")
   for i = 2; i <= evnum; i = i + 2 {
      fmt.Print(i, "\t")
   }
   fmt.Println()
}
```

Write a function in GO language to find the square of a number and write a benchmark for it.

```
package main

import "fmt"
import "math"

func main() {
        var x,result float64
  fmt.Print("Enter a number : ")
  fmt.Scan(&x)

        result = math.Sqrt(x)
        fmt.Println(result)
}
```

Write a program in GO language to demonstrate working of slices (like append, remove, copy etc.)

```
package main

import "fmt"

func main() {
   slice1 := []int{1,3,5,7}
   slice2 := []int{2,4,6,8}
   slice1=append(slice1,2)
   fmt.Println("Slice1 = ", slice1, "| Slice2 = ", slice2)
   copy(slice2, slice1[:])
   fmt.Println("Slice1 = ", slice1, "| Slice2 = ", slice2)
}
```

Write a program in GO language to demonstrate function return multiple values.

```
package main

import "fmt"

func myfunc(p, q int)(int, int, int ){
   return p - q, p * q, p + q
}

// Main Method
func main() {
  var myvar1, myvar2, myvar3 = myfunc(4, 2)
  fmt.Printf("Result is: %d", myvar1 )
  fmt.Printf("\nResult is: %d", myvar2)
  fmt.Printf("\nResult is: %d", myvar3)
}
```

Write a program in GO language that prints out the numbers from 0 to 10, waiting between 0 and 250 ms after each one using the delay function

```
package main

import (
  "fmt"
  "time"
  "math/rand"
)

func f(n int) {
  for i := 0; i < 10; i++ {
    fmt.Println(n, ":", i)
    amt := time.Duration(rand.Intn(250))
    time.Sleep(time.Millisecond * amt)
  }
}

func main() {
  for i := 0; i < 10; i++ {
    go f(i)
  }
  var input string
  fmt.Scanln(&input)
}
```

Write a program in GO language to print a multiplication table of number using function.

```
package main
import "fmt"
func main(){
```

```
  var n int
  fmt.Print("Enter the number to print the multiplication table:")
  fmt.Scanf("%d", &n)
  for i:=1; i<11; i++ {
    fmt.Println(n, "x", i, "=", n*i)
  }
}
```

Write a program in GO languageto illustrate the function returning multiple values(add, subtract).

```
package main
import "fmt"
func sumDiff(a int, b int) (int, int) {
   return (a + b), (a - b)
}

func main() {
   var a = 68
   var b = 100
   var sum, diff = sumDiff(a, b)
   fmt.Println("Sum = ", sum, "\nDifference = ", diff)
}
```

Write a program in Go language how to create a channel and illustrate how to close a channel using for range loop and close function.

```
package main

import "fmt"
func myfun(mychnl chan string) {

   for v := 0; v < 4; v++ {
      mychnl <- "GeeksforGeeks"
```

```
    }
    close(mychnl)
}

func main() {
    c := make(chan string)
    go myfun(c)
    for {
        res, ok := <-c
        if ok == false {
            fmt.Println("Channel Close ", ok)
            break
        }
        fmt.Println("Channel Open ", res, ok)
    }
}
```