Q.Write a program in Go language to add or append content at the end of a text file.

```
package main
import (
"fmt"
"os"
)
 func main() {
message := "Add this content at end"
filename := "test.txt"
 f, err := os.OpenFile(filename, os.O_RDWR|os.O_APPEND|os.O_CREATE, 0660)
 if err != nil {
fmt.Println(err)
os.Exit(-1)
}
defer f.Close()
fmt.Fprintf(f, "%s\n", message)
}
```

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

```
package main
import "fmt"
func main() {
   evenSum := 0
   oddSum := 0
   for i := 1; i <= 100; i++ {
      if i % 2 == 0 {
         evenSum += i
      } else {
         oddSum += i
      }
   }
  fmt.Println("Sum of even numbers from 1 to 100:", evenSum)
   fmt.Println("Sum of odd numbers from 1 to 100:", oddSum)
}
```

Q.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, 2, 3, 4, 5}
slice1 = append(slice1, 6, 7, 8)
fmt.Println("After appending elements:", slice1)
slice1 = append(slice1[:2], slice1[4:]...)
fmt.Println("After removing element:", slice1)
slice2 := make([]int, len(slice1))
```

```
copy(slice2, slice1)
fmt.Println("Copied slice:", slice2)
}
```

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

```
package main
import "fmt"
func swap(x, y string) (string, string) {
   return y, x
}
func main() {
   a, b := swap("hello", "world")
   fmt.Println(a, b)
}
```

Q.Write a program in GO language using a function to checkwhether the accepted number is palindrome or not.

```
package main
import "fmt"
func checkPalindrome(num int) string{
input_num := num
var remainder int
res := 0
for num>0 {
remainder = num % 10
res = (res * 10) + remainder
num = num / 10
}
if input_num == res {
return "Palindrome"
} else {
return "Not a Palindrome"
}
}
func main(){
fmt.Println(checkPalindrome(121))
fmt.Println(checkPalindrome(123))
fmt.Println(checkPalindrome(1331))
fmt.Println(checkPalindrome(1231))
}
```

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

```
package main
import "fmt"
func myfunc(p, q int)(int, int, int ){
```

```
return p - q, p * q, p + q
}
  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)
}
```

Q. Write a program in GO language to print a multiplication table ofnumber using function

```
package main
import "fmt"
func multiplicationTable(num int) {
 for i := 1; i <= 10; i++ {
 fmt.Printf("%d x %d = %d\n", num, i, num*i)
 }
 }
 func main() {
 var num int
 fmt.Print("Enter a number: ")
 fmt.Scanln(&num)
 multiplicationTable(num)
 }
```

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

```
package main
  import (
"fmt"
)
func main() {
var value interface{} = 20024
 var value1 int = value.(int)
 fmt.Println(value1)
 value2, test := value.(string)
 if test {
 fmt.Println("String Value found!")
 fmt.Println(value2)
 } else {
  fmt.Println("String value not found!")
 }
 }
```

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

```
package main
```

```
 import (
"fmt"
)
 func main() {
var num int
fmt.Print("Enter a number: ")
fmt.Scan(&num)
if num >= 10 && num < 100 {
fmt.Println("The number is two digits.")
} else {
fmt.Println("The number is not two digits.")
}
}
```

Q. Write a program in GO language to create a buffered channel, store few values in it and find channel capacity and length. Read values from channel and find modified length of a channel

```
package main
import "fmt"
func main() {
ch := make(chan int, 3)
ch <- 10
ch <- 20
ch <- 30
fmt.Println("Channel capacity:", cap(ch))
fmt.Println("Channel length:", len(ch))
fmt.Println("Values from channel:")
fmt.Println(<-ch)
fmt.Println(<-ch)
fmt.Println("Modified channel length:", len(ch))
}
```

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

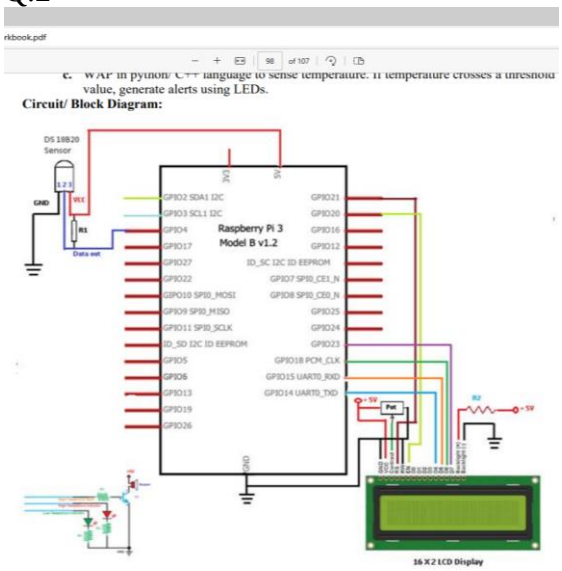
```
package main
import "fmt"
func main()  {
var a, b, temp int
```

```
fmt.Print("Enter the First  = ")
fmt.Scanln(&a)
fmt.Print("Enter the Second  = ")
fmt.Scanln(&b)
temp = a
a = b
b = temp
fmt.Println("The First Number after  = ", a)
fmt.Println("The Second Number after = ", b)
}
```

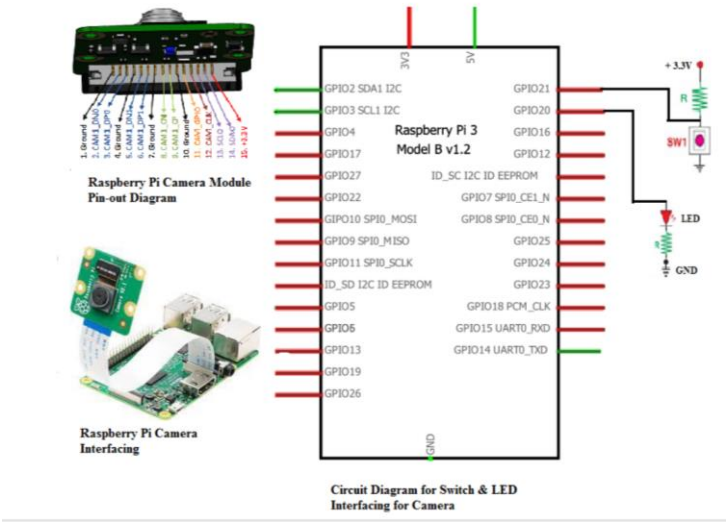
Q.Write a program in GO language to print sum of all even and odd numbers separately between 1 to 100

```
package main
import "fmt"
func main() {
var evenSum, oddSum int
for i := 1; i <= 100; i++ {
if i%2 == 0 {
evenSum += i
} else {
oddSum += i
}
}
fmt.Println("Sum of even numbers between 1 to 100:", evenSum)
fmt.Println("Sum of odd numbers between 1 to 100:", oddSum)
}
```

Q.2



Circuit/Block Diagram:



Circuit Diagram for Switch & LED
Interfacing for Camera

```
import serial
import time

# Define the serial port and baud rate
ser = serial.Serial('COM3', 9600)

def turn_on_buzzer():
    # Send the command to turn on the buzzer
    ser.write(b'on\n')

def turn_off_buzzer():
    # Send the command to turn off the buzzer
    ser.write(b'off\n')

# Wait for 2 seconds before starting the loop
time.sleep(2)

while True:
    # Wait for user input
    user_input = input("Enter 'on' to turn on the buzzer, 'off' to turn it off, or 'exit' to quit: ")

    # Check if the user wants to exit the program
    if user_input == "exit":
        break

    # Turn on the buzzer
    if user_input == "on":
        turn_on_buzzer()

    # Turn off the buzzer
    elif user_input == "off":
        turn_off_buzzer()

    # Invalid input
    else:
        print("Invalid input. Please enter 'on', 'off', or 'exit'.")
```