Assignment 1

// Assi1 A1.WAP in go language to print Student name, rollno, division and college name.

```
package main
import "fmt"
func main(){
        nm:="NIlesh"
        rno:=4819
        div:="A"
        college:="H.V.D"

        fmt.Println("Name= ",nm)
        fmt.Println("Rno= ",rno)
        fmt.Println("Division= ",div)
        fmt.Println("College= ",college)
}
```

//Assi1 A2.WAP in go language to print whether the number is even or odd.

```
package main
import f"fmt"
func main(){
        var n int
        f.Print("Enter a number=")
        f.Scan(&n)
        if n%2==0{
        f.Print(n," is even")
        }else{
        f.Print(n," is odd")
    }
}
```

//Assi1 A3.WAP in go language to swap the number without a temporary variable.

```
package main
import f "fmt"
func main(){
        var a int
        var a int
```

```
        f.Print("Enter two numbers=")
        f.Scanf("%d%d",&a,&b)
        a,b=b,a
        f.Println("After swapping=")
        f.Println("First number=%d,a)
        f.Println("Second number=%d",b)
    }
```

//Assi1 A4.WAP in go Language to print the address of a variable.

```
    package main
    import f "fmt"
    func main(){
        var n int
        var s string
        n=20
        s="Hello"
        f.Println("Address of n",&n)
        f.Println("Address of s",&s)
    }
```

//Assi1 B1.WAP in go to the print table of given numbers.

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

//Assi1 B3.WAP in go language to print Fibonacci series of n terms.

```
package main
import f "fmt"
func main(){
    var a,b,c,n int
    a=0
    b=1
```

```
        f.Println("Enter any terms")
        f.Scanf("%d",&n)
        f.Printf("%d%d",a,b)
        for i:=3;i<=n;i++{
        c=a+b
        f.Printf(" %d",c)
        a=b
        b=c
}
}
```

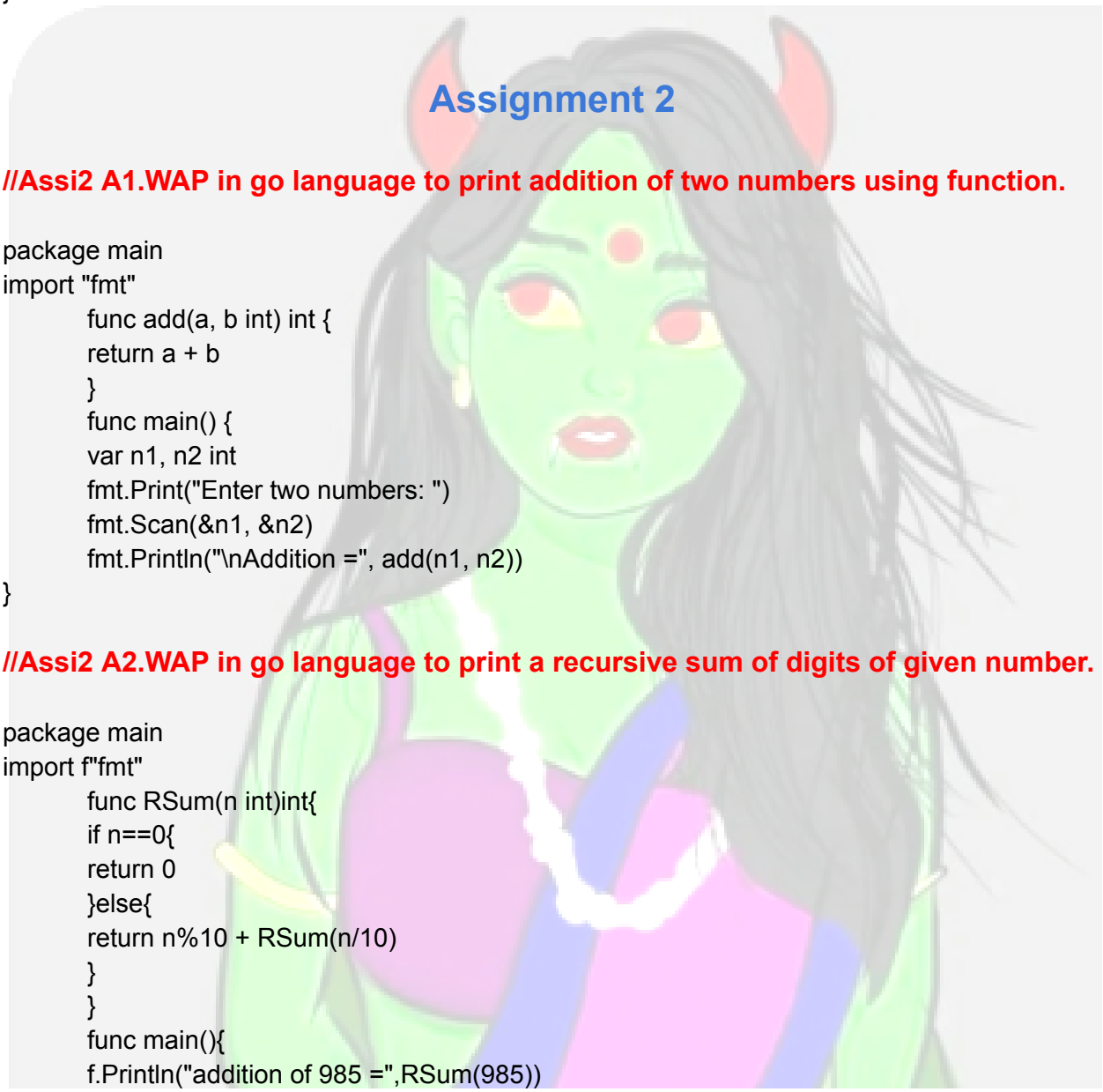
//Assi1 C3.WAP in go language to accept user choice and print answers using arithmetic operators.

```
package main
import f "fmt"
func main(){
        var n1,n2 int
        var ch string
        f.Print("Enter two number : ")
        f.Scan(&n1,&n2)
        f.Print("\n 1.Addition \n 2.Substraction \n 3.Multiplication \n 4.Division ")
        f.Print("\nEnter your choice (1-4) :")
        f.Scan(&ch)
        switch ch{
        case "+":f.Print("\n Addition = ",n1+n2)
        case "-":f.Print("\n Substraction = ",n1-n2)
        case "*":f.Print("\n Multiplication = ",n1*n2)
        case "/":f.Print("\n Division = ",n1/n2)
        default : f.Print("Invalid")
}
}
```

//Assi1 C4.WAP in go language to check whether the accepted number is single digit or not.

```
package main
import f "fmt"
func main(){
        var n int
        f.Print("Enter a number=")
```

```
        f.Scanf("%d",&n)
        if n>9 && n<100{
        f.Println("Number is 2 digit")
        }else{
        f.Println("Number is not 2 digit")
}
}
```



Assignment 2

//Assi2 A1.WAP in go language to print addition of two numbers using function.

```
package main
import "fmt"
        func add(a, b int) int {
        return a + b
        }
        func main() {
        var n1, n2 int
        fmt.Print("Enter two numbers: ")
        fmt.Scan(&n1, &n2)
        fmt.Println("\nAddition =", add(n1, n2))
}
```

//Assi2 A2.WAP in go language to print a recursive sum of digits of given number.

```
package main
import f"fmt"
        func RSum(n int)int{
        if n==0{
        return 0
        }else{
        return n%10 + RSum(n/10)
        }
        }
        func main(){
        f.Println("addition of 985 =",RSum(985))
        }
```

//Assi2 A3.WAP in go language using a function to check whether the accepted number is palindrome or not.

```
package main
import f"fmt"
	func Palindrome(n int){
	t:=n
	rev:=0
	for n>0{
	rem:=n%10
	rev=rev*10+rem
	n=n/10
	}
	if(t==rev){
	f.Print("No. is Palindrome")
	}else{
	f.Print("No. isn't Palindrome")
	}
	}
	func main(){
	var num int
	f.Print("Enter a Number=")
	f.Scan(&num)
	Palindrome(num)
}
```

//Assi2 B1.WAP in go language to swap two numbers using call by reference concept.

```
package main
import f"fmt"
	func swap(a,b *int){
	x:=*a
	*a=*b
	*b=x
	}
	func main(){
	var a,b int
	f.Print("Enter two numbers=")
	f.Scan(&a,&b)
	f.Println("Original Value=",a,b)
	swap(&a,&b)
	f.Println("After Swap=",a,b)
}
```

//Assi2 B2.WAP in go language to demonstrate use of names returns variables.

```
package main
import f"fmt"
	func add(a,b int)(sum int){
	sum=a+b
	return
	}
	func main(){
	var x,y int
	f.Print("Enter two numbers=")
	f.Scan(&x,&y)
	f.Println("Addition=",add(x,y))
}
```

//Assi2 C3.WAP in go language to illustrate the concept of returning multiple values from a function.

```
package main
import f"fmt"
	func calculate(a,b float32) (sum,sub,Multiply,division float32) {
		sum = a + b
		sub = a-b
		Multiply = a * b
		division = a/b
		return
	}
	func main() {
	var x,y float32
	f.Print("Enter two numbers=")
	f.Scan(&x,&y)
	s1,s2,s3,s4:= calculate(x,y)
	f.Print("addition=",s1)
	f.Print("\nsubtraction=",s2)
	f.Print("\nmultiplication=",s3)
	f.Print("\ndivision=",s4)
}
```

//EXTRA.Multiplication table using function.

```
package main
import f"fmt"
func table(n int) {
	f.Printf("Multiplication Table for %d:\n", n)
```

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

//EXTRA.Function to print square and cube.

```
package main
import f"fmt"
   func squarecube(a int) (square,cube int){
      square=a*a
      cube=a*a*a
      return
   }

   func main(){
      var n int
      f.Print("Enter a number=")
      f.Scan(&n)
      s1,s2:=squarecube(n)
      f.Print("Square=",s1)
      f.Print("\nCube=",s2)

   }
```

//Assi B3.WAP in go language to show the compiler throws an error if a variable is declared but not used.

//Assi C1.WAP in go language to illustrate the concept of call by value.

//Assi C2.WAP in go language to create a file and write hello world in it and close the file by using defer statement.

Assignment 3

//Assi3 A1.WAP in go language to find the largest and smallest number in an array.

//Assi3 A2.WAP in go language to accept the book details such as BookID, Title, Author, Price. Read and display the details of n number of books.

```
package main
import f "fmt"
        type book struct{
        bid int
        title,author string
        price float32
}
        func main(){
        var n int
        f.Print("Enter how many books ?=")
        f.Scan(&n)

        b:=make([]book,n)
        for i:=0;i<n;i++{
        f.Print("\n Enter book id,title,author,price for book ",i+1,"=")
        f.Scan(&b[i].bid,&b[i].title,&b[i].author,&b[i].price)
}
        f.Println("\n Book Details ")
        for i:=0;i<n;i++{
        f.Println(b[i].bid,"\t",b[i].title,"\t",b[i].author,"\t",b[i].price)
}
}
```

//Assi3 A3.WAP in go language to Initialize a Slice using Multi-Line Syntax and display

```
package main
import f "fmt"
        func main(){
        temp:=[] float32{
```

```
	25.6,26.6,52.8}

	f.Print("Slice = ",temp)
}
```

//Assi3 B1.WAP in go language to create and print multidimensional Slice.

//Assi3 B2.WAP in go language to sort array elements in ascending order.

```
package main
import f "fmt"
	func main(){
	var arr=[5] int{10,50,40,30,20}
	for i:=0;i<5;i++{
	for j:=i+1;j<5;j++{
	if(arr[i]>arr[j]){
	arr[i],arr[j]=arr[j],arr[i]
	}
	}
	}
	f.Println("\n Sorted array = ",arr)
}
```

//Assi3 B3.WAP in go language to accept n student details like roll_no, stud_name, mark1, mark2, mark3. Calculate the total and average of marks using structure

```
package main
import f "fmt"
	type student struct{
	rno int
	sname string
	m1,m2,m3,avg,total float32
	}
	func main(){
	var n int
	f.Print("Enter how many Student details=")
	f.Scan(&n)
```

```
        stud:=make([]student,n)
        for i:=0;i<n;i++{
        f.Print("Enter Student Rollno,Sanme,Marks m1,m2,m3=")
        f.Scan(&stud[i].rno,&stud[i].sname,&stud[i].m1,&stud[i].m2,&stud[i].m3)
        }
        f.Println("\n===Student Details===")
        for i:=0;i<n;i++{
        f.Println(stud[i].rno,"\t",stud[i].sname,"\t",stud[i].m1,"\t",stud[i].m2,"\t",stud[i].m3)
        stud[i].total=stud[i].m1+stud[i].m2+stud[i].m3
        stud[i].avg=stud[i].total/3
        f.Println("Total marks of student=",stud[i].total)
        f.Println("Average of marks=",stud[i].avg)
}
}
```

//Assi3 C1.WAP in go language to accept two matrices and display it's multiplication.

```
package main
import f "fmt"
        func main(){
        var r int
        var c int
        var i,j int
        f.Println("Enter How many rows and col ? ")
        f.Scan(&r,&c)
        mat:=make([][]int,r)
        for i=0;i<r;i++{
        mat[i]=make([]int,c)
        }
        trans:=make([][]int,c)
        for i=0;i<c;i++{
        trans[i]=make([]int,r)
        }
        f.Println("Enter matrix element ")
        for i=0;i<r;i++{
        for j=0;j<c;j++{
        f.Scan(&mat[i][j])
        trans[j][i]=mat[i][j]
        }
        }
        f.Println("Orignal matrix = ")
        for i=0;i<r;i++{
        for j=0;j<c;j++{
```

```
        f.Print(mat[i][j],"\t")
        }
        f.Println()
        }

        f.Println("Transpose matrix = ")
        for i=0;i<c;i++{
        for j=0;j<r;j++{
        f.Print(trans[i][j],"\t")
        }
        f.Println()
}
}
```

//Assi3 C2.WAP in go language to accept n records of employee information (eno,ename, salary) and display record of employees having maximum salary.

```
package main
import f "fmt"
        type employee struct{
        eno int
        ename string
        sal int
        }
        func main(){
        var n,j int
        max:=0
        f.Print("Enter how many Employee details=")
        f.Scan(&n)

        emp:=make([]employee,n)
        for i:=0;i<n;i++{
        f.Println("Enter Employee number,name,salary=")
        f.Scan(&emp[i].eno,&emp[i].ename,&emp[i].sal)
        if(max<emp[i].sal){
        max=emp[i].sal
        j=i
        }
        }
        f.Println("Employee with maximum salary = ",emp[j])
}
```

//Assi3 C3.WAP in go language to demonstrate working of slices (like append, remove, copy etc.)

```
package main
import f "fmt"
func main(){
	var slice=[] int{10,20,30}
	f.Println("Orignal slice = ",slice)
	slice=append(slice,40,50,60)
	f.Println("After append slice = ",slice)
	index:=2
	slice=append(slice[:index],slice[index+1:]...)

	f.Println("After remove value at ",index," \nslice = ",slice)
	slice2:=make([]int,len(slice))
	copy(slice2,slice)
	f.Println("Copied slice = ",slice2)
}
```

Assignment 4

//Assi4 A3. Write a program in go language to create structure author. Write a method show() whose receiver is struct author.

```
package main
import f "fmt"

type author struct {
	ano int
	anm string
	city string
}

func (a author) show() {
	f.Println("Author number = ", a.ano)
	f.Println("Author Name = ", a.anm)
	f.Println("Author City = ", a.city)
}

func main() {
	var a1 author
	f.Println("Enter autor number,name,city")
	f.Scan(&a1.ano, &a1.anm, &a1.city)
	a1.show()
}
```

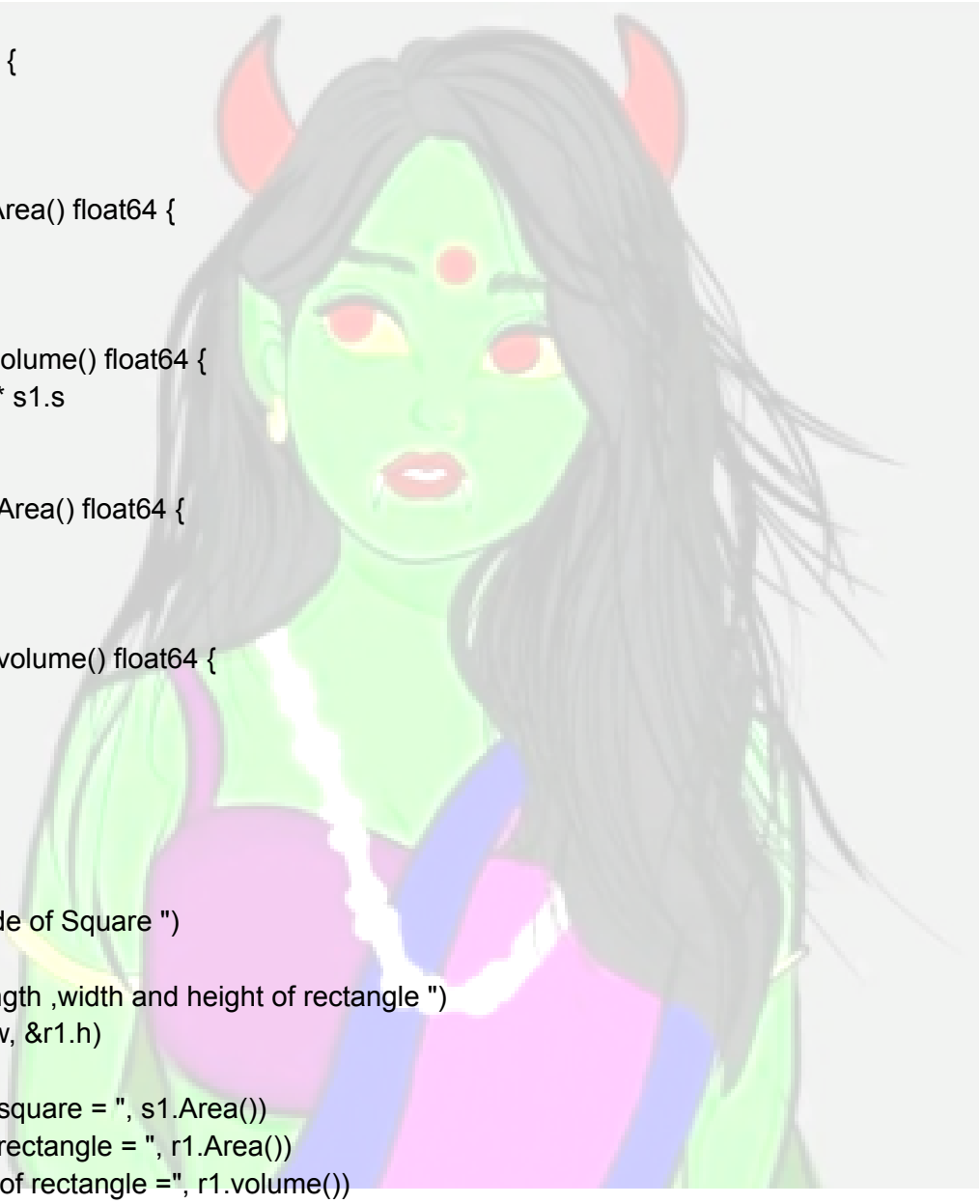
//Assi4 B2: Write a program in go language to demonstrate working type switch in interface.

```
package main
import f "fmt"

type date1 interface {
        d int
        m int
        yy int
}

type emp interface {
        eid int
        enm string
        dob date1
}

func main() {
        var e emp
        f.Println("Enter eid and name")
        f.Scan(&e.eid, &e.enm)
        f.Println("Enter Dob(dd,mm,yy) ")
        type myDate struct { // Added a concrete type for date
        d, m, yy int
}
var dob myDate
        f.Scan(&dob.d, &dob.m, &dob.yy)

        e = struct { // Composite literal to satisfy the interface
        eid int
        enm string
        dob date1
        }{e.eid, e.enm, dob}

        f.Println("Employee details = ", e)
}
```

//EXTRA: Write a program in go language to create an interface shape that includes area and volume. Implements these methods in circle and rectangle type.

```
package main
import f "fmt"

type shape interface {
```

```
Area() float64
volume() float64
}

type rectangle struct {
l, w, h float64
}

type square struct {
s float64
}

func (s1 square) Area() float64 {
return s1.s * s1.s
}

func (s1 square) volume() float64 {
return s1.s * s1.s * s1.s
}

func (r rectangle) Area() float64 {
return r.l * r.w
}

func (r rectangle) volume() float64 {
return r.l * r.w * r.h
}

func main() {
var r1 rectangle
var s1 square
f.Println("Enter Side of Square ")
f.Scan(&s1.s)
f.Println("Enter length ,width and height of rectangle ")
f.Scan(&r1.l, &r1.w, &r1.h)

f.Println(" Area of square = ", s1.Area())
f.Println(" Area of rectangle = ", r1.Area())
f.Println(" Volume of rectangle =", r1.volume())
f.Println("Volume of square =", s1.volume())

}
```



//Assi4 B3: Write a program in Go language to copy all elements of one array into another using a method.

```
package main
import f "fmt"

type array struct {
arr1 [5]int
}

func (a *array) copyArr(a1 [5]int) { // Changed to pointer receiver
a.arr1 = a1
f.Println("Copied Array = ", a.arr1)
}

func main() {
var a2 = [5]int{10, 20, 30, 40, 50}
var a1 array
a1.copyArr(a2)
}
```

//Assi4 B2: Write a program in go language to demonstrate working type switch in interface.

```
package main
import f "fmt"

func main() {
var value interface{} = true
switch t := value.(type) {
case int:
f.Println("Type is integer", t)

case float64:
f.Println("Type is float", t)

case string:
f.Println("Type is String", t)

case bool:
f.Println("Type is Boolean", t)

default:
f.Printf("Type is %T", t) // Added default case
}
}
```

//Assi4 A1. Write a program in go language to create an interface shape that includes area and perimeter. Implements these methods in circle and rectangle type.

```
package main
import f "fmt"

type shape interface {
Area() float64
perimeter() float64
}

type rectangle struct {
l, w float64
}

type circle struct {
r float64
}

func (r rectangle) Area() float64 {
return r.l * r.w
}

func (r rectangle) perimeter() float64 {
return 2 * (r.l + r.w)
}

func (c circle) Area() float64 {
return 3.14 * c.r * c.r
}

func (c circle) perimeter() float64 {
return 2 * 3.14 * c.r
}

func main() {
var c circle
var r rectangle
f.Println("Enter length and width of rectangle")
f.Scan(&r.l, &r.w)
f.Println("Enter radius of circle ")
f.Scan(&c.r)
f.Println("Area of Rectangle = ", r.Area())
f.Println("Perimeter of Rectangle  = ", r.perimeter())
```

```
f.Println("Area of circle", c.Area())
f.Println("Perimeter of circle ", c.perimeter())
}
```

//Assi4 B1. Write a program in go language to create structure student. Write a method show() whose receiver is a pointer of struct student.

```
package main
import f "fmt"

type student struct {
sno int
snm string
city string
}

func (s *student) show() {
f.Println("Student number = ", s.sno)
f.Println("Student Name = ", s.snm)
f.Println("Student City = ", s.city)
}

func main() {
var s1 student
f.Println("Enter Student number,name,city")
f.Scan(&s1.sno, &s1.snm, &s1.city)
s1.show() // No need for & here
}
```

//Assi4 C2. Write a program in go language to store n student information (rollno, name, percentage) and write a method to display student information in descending order of percentage.

```
package main
import f "fmt"

type student struct {
sro int
snm string
per float64
}

func main() {
var n int
var i, j int
f.Println("Enter how many student ")
```

```
f.Scan(&n)
stud := make([]student, n)
for i = 0; i < n; i++ {
f.Println("Enter student Rollno,name,Percentage")
f.Scan(&stud[i].sro, &stud[i].snm, &stud[i].per)
}
f.Println("Student Details ")
for i = 0; i < n; i++ {
f.Println("Student rollno = ", stud[i].sro)
f.Println("Student Name = ", stud[i].snm)
f.Println("Student marks = ", stud[i].per)
}

for i = 0; i < n; i++ {
for j = i + 1; j < n; j++ {
if stud[i].per < stud[j].per {
temp := stud[i]
stud[i] = stud[j]
stud[j] = temp
}
}
}
f.Println("Details in descending order")
f.Println("Student Percentage \t Student name \t Student Roll number")

for i = 0; i < n; i++ {
f.Println(stud[i].per, "\t", stud[i].snm, "\t", stud[i].sro)
}
}
```

//Assi4 C1. Write a program in go language to demonstrate working type assertion.

```
package main
import f "fmt"

func main() {
var anyval interface{}
anyval = 45
v, ok := anyval.(int)
if ok {
f.Println("Type of ", v, "is Integer ")
}

v1, ok := anyval.(float64)
if ok {
```

```
f.Println("Type of ", v1, "is Float ")
}

v2, ok := anyval.(string)
if ok {
f.Println("Type of ", v2, "is String ")
}

v3, ok := anyval.(bool)
if ok {
f.Println("Type of ", v3, "is Boolean ")
}
}
```

Assignment 5: Goroutines and Channels

- SET A

1. Write a go program using go routine and channel that will print the sum of the squares and cubes of the individual digits of a number. Example if number is 123 then squares = (1 * 1) + (2 * 2) + (3 * 3) cubes = (1 * 1 * 1) + (2 * 2 * 2) + (3 * 3 * 3). Output: Sum of squares= 170, Sum of cubes= 1366, Final sum of squares and cubes = 1536.

2. WAP in GO program that executes 5 goroutines simultaneously which generates numbers from 0 to 10, waiting between 0 and 250 ms after each go routine.

3. Write a go program that creates a slice of integers, checks numbers from the slice are even or odd and further sent to respective go routines through channel and display values received by goroutines.

- SET B

1. WAP in Go to create buffered channels, store few values in it and find channel capacity and length. Read values from the channel and find modified length of a channel.

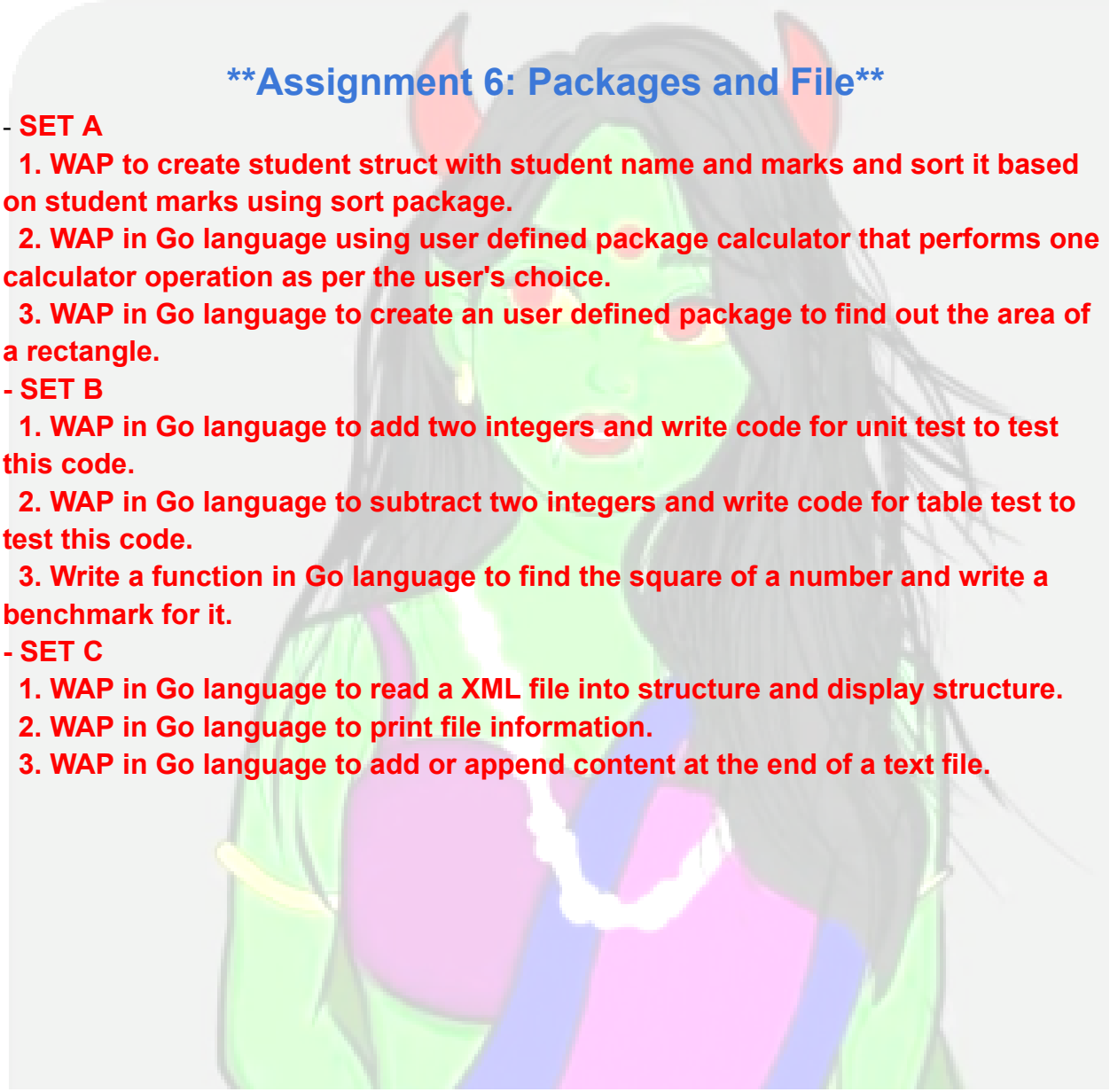
2. WAP in Go main go routine to read and write Fibonacci series to the channel.

3. WAP in Go how to create a channel and illustrate how to close a channel using for range loop and close function.

- SET C

1. Write a go program to implement the checkpoint synchronization problem which is a problem of synchronizing multiple tasks. Consider a workshop where

several workers assemble details of some mechanism. When each of them completes his work, they put the details together. There is no store, so a worker who finished its part first must wait for others before starting another one. Putting details together is the checkpoint at which tasks synchronize themselves before going their paths apart.



Assignment 6: Packages and File

- SET A
  1. WAP to create student struct with student name and marks and sort it based on student marks using sort package.
  2. WAP in Go language using user defined package calculator that performs one calculator operation as per the user's choice.
  3. WAP in Go language to create an user defined package to find out the area of a rectangle.
- SET B
  1. WAP in Go language to add two integers and write code for unit test to test this code.
  2. WAP in Go language to subtract two integers and write code for table test to test this code.
  3. Write a function in Go language to find the square of a number and write a benchmark for it.
- SET C
  1. WAP in Go language to read a XML file into structure and display structure.
  2. WAP in Go language to print file information.
  3. WAP in Go language to add or append content at the end of a text file.