Go:

package main // package Declaration

import "fmt"

func main() {

    fmt.Printf("Hello.. I am Here")

}

To run Go Program:

Go run <file name>.go

Variable :

To Store Value

It is like container

Syntax –

Var <Variable name> = <Value>

Var name = “Suraj Mandhane”

Note : if variable is declared and not used then Go lang shows error. Same for Package

Constant Keyword:

Const <variable name> = <value>

Const confTicket = 50

Print Formatted Data:

fmt.Printf("There are %v total ticket and %v are remaining", confTicket, remainingTicket)

Output:

There are 50 total ticket and 50 are remaining

Data Type:

String - Integers – Maps – Boolean – Arrays

Syntax-

Var <variable\_name> Data\_Type

var userName string

var userTicket int

Array Program:

func main() {

    var arr1 = [4]int{1, 3, 4, 5}

    fmt.Println(arr1)

    for i := 0; i < len(arr1); i++ {

        println(arr1[i]) // It will print all Element from Array arr1

    }

    fmt.Println("Hello")

}

Getting User Input:

Array and Slice :

Array -

var arr1 = [10]int{1, 2, 3, 4, 5, 5, 5, 7, 8, 9}

    arr2 := [20]string{"aaa", "bbb", "ccc"}

    for i := 0; i < len(arr1); i++ {

        fmt.Print(arr1[i])

    }

    fmt.Print("\n")

    for i := 0; i < len(arr2); i++ {

        fmt.Print(arr2[i] + " ")

    }

Slice –

Slice is an Abstraction of Array

Slices are more fliexible and Poweful.

Variable Length or get an Sub Array of its own.

Slices are also INDEX BASED and Have a Size, but Resized when Needed.

var slice1 = []int{1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 0}

    slice2 := []string{"qqq", "rrr", "ttt", "ppp"}

    slice2 = append(slice2, "zzz")

    for i := 0; i < len(slice1); i++ {

        fmt.Print(slice1[i])

    }

    for i := 0; i < len(slice2); i++ {

        fmt.Print(slice2[i])

    }

Loops:

for1 := []int{1, 2, 2, 3, 4, 5, 6}

var for2 []int

    for i := 1; i < 5; i++ {

        var in int

        fmt.Scanln(&in)

        for2 = append(for2, in)

    }

    for \_, i := range for1 {

        fmt.Print(i)

    }

    fmt.Println()

    for \_, j := range for2 {

        fmt.Print(j)

    }

IF/Else:

Strings :

var name = "Suraj Mandhane"

    var re = strings.Split(name, " ")

    fmt.Println(name)

    fmt.Println(re[0])

    fmt.Println(re[1])

    re1 := strings.Fields(name) // This Function Split the the String with WHITE SPACE as Seperator

    fmt.Print(re1)

Switch Case:

func main() {

    fmt.Println("---------------------Welcome to SWITCH CASE----------------")

    city := "Pune"

    switch city {

    case "Raipur":

        fmt.Println("Welcome to Raipur")

    case "Rampur":

        fmt.Println("Welcome to Rampur")

    case "Nagar":

        fmt.Println("Welcome to Nagar")

    case "Pune":

        fmt.Println("Welcome to Pune")

    default:

        fmt.Print("Default Statement")

    }

}

Functions:

* Encapsulation code into own container (= function) . which logically belong Together
* Like Variable Name, You should give a function a Descriptive Name’
* Call the Function by its Name, whenever you want to execute this Block of Code
* Every Function has Al least one function which is the **main()** functions

Functions parameter:

func main() {

    fmt.Println("Functions")

    name := "Suraj"

    greet(name)

}

func **greet(name string)** {

    fmt.Println("This is GREET() Function")

    fmt.Println(name)

}

**Multiple Return Statement:**

func main() {

    var name string

    fmt.Print("Enter Name : ")

    fmt.Scan(&name)

    var email string

    fmt.Print("Enter Email : ")

    fmt.Scan(&email)

    fmt.Println(Validate(name, email))

    //fmt.Println(isEmailValid)

}

func Validate(name string, email string) (bool, bool) {

    isNameValid := len(name) >= 2 && len(name) < 10

    isEmailValid := strings.Contains(email, ".com") && strings.Contains(email, "@")

    return isNameValid, isEmailValid

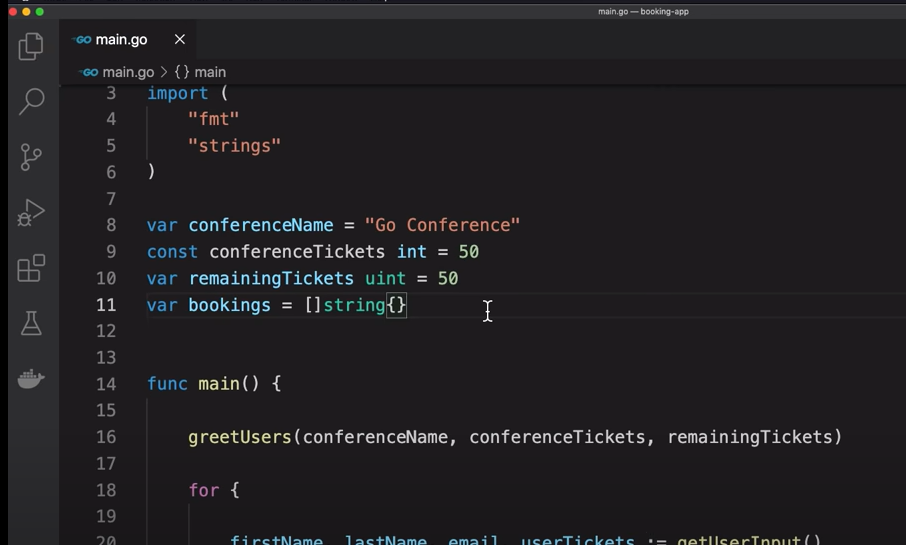
}

Package Level Variables:

Defined at the Top outside of all functions

They can be accessed inside any of the Functions

And in all files, which are in the Same Package



PACKAGES:

Go Programs are Organized into Packages.

A Packages is a Collection of Go Files.

VARIABLE SCOPE:

**Local Package Global**

**Local:**

Declaration within Function

Declaration within Block e.g in For Loop, If/Else

**Package:**

Declaration outside all functions

(Can be used everywhere in same Package)

**Global:**

Declaration outside all functions and UPPERCASE first Letter

Variable Scope:

Scope is the region a program , where a defined variable can be Accessed.

MAPs:

Map key to Unique Key Values.

You can retrieve the Value by using its key Later

Note:

All keys have the Same Data Type

All values have the Same Data Type

To Make an EMPTY MAP,

Use Make() Function

var name = make(map[string]string)