**Department of Computer Science and Engineering**

**Amrita School of Engineering**

**Amrita Vishwa Vidyapeetham, Coimbatore**

**Lab : Basics of Go Programming**

**Date of Issue: 03/02/2022 – 2:10PM**

**Date of Submission : 09/02/2022 - 10PM**

1. GO environment

Solution:

package main

import "fmt"

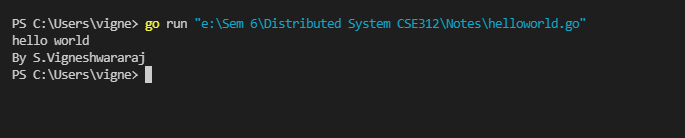
func main() {

    fmt.Println("hello world")

    fmt.Println("By S.Vigneshwararaj")

}

Output:



1. Datatypes supported in GO

Solution:

* Integers (Signed and UnSigned)
* Floats.
* Complex Numbers.
* Byte.
* Rune.
* String.
* Booleans.

1. Defining and declaring variables with different data types.

Solution:

package main

import "fmt"

func main() {

    var number int = 20

    var str string = "S Vigneshwararaj"

    var floatNum float64 = 40.25

    var bools bool = false

    var bytes byte = 1

    fmt.Printf("The value of number is : %d\t  and the data type of number is : %T\n", number, number)

    fmt.Printf("The value of str is : %s\t  and the data type of str is : %T\n", str, str)

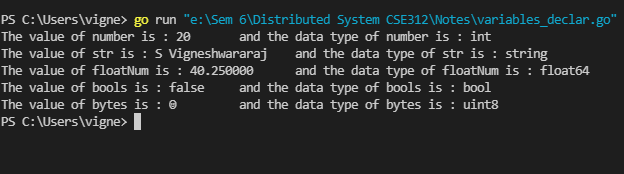
    fmt.Printf("The value of floatNum is : %f\t  and the data type of floatNum is : %T\n", floatNum, floatNum)

    fmt.Printf("The value of bools is : %t\t  and the data type of bools is : %T\n", bools, bools)

    fmt.Printf("The value of bytes is : %c\t  and the data type of bytes is : %T\n", bytes, bytes)

}

Output:



1. Defining, writing and calling functions.

Solution:

package main

import "fmt"

func addi(num1, num2 int) int {

    Sum := num1 + num2

    return Sum

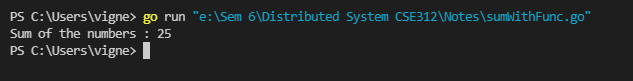
}

func main() {

    fmt.Printf("Sum of the numbers : %d", addi(15, 10))

}

Output:



1. Different general purpose packages and their necessity for different functions.
2. Defining and using arrays.

Solution:

package main

import "fmt"

func main() {

    arr := [5]float64{25.0, 45.36, 89.7, 12.3, 69.4}

    arrays(arr)

}

func arrays(arr [5]float64) {

    fmt.Println("Elements of the array:")

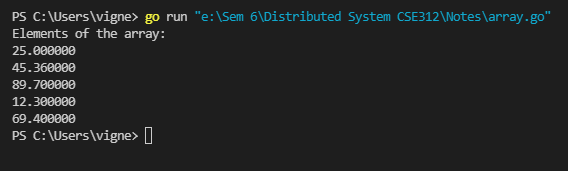
    for i := 0; i < 5; i++ {

        fmt.Printf("%f\n", arr[i])

    }

}

Output:



1. Writing different conditional statements

Solution:

package main

import "fmt"

func main() {

    var num1 int

    var num2 int

    var operator int

    fmt.Printf("Please Enter first Number\n")

    fmt.Scanln(&num1)

    fmt.Printf("Please Enter second Number\n")

    fmt.Scanln(&num2)

    fmt.Printf("Please Enter\n 1 - add\n 2 - sub\n 3 - mul\n 4 - div\n")

    fmt.Scanln(&operator)

    if operator == 1 {

        fmt.Printf("Sum of the numbers : %d", addi(num1, num2))

    } else if operator == 2 {

        fmt.Printf("Sub of the numbers : %d", subs(num1, num2))

    } else if operator == 3 {

        fmt.Printf("Multiplication of the numbers : %d", multi(num1, num2))

    } else if operator == 4 {

        fmt.Printf("Div of the numbers : %d", div(num1, num2))

    } else {

        fmt.Printf("Wrong operation choice")

    }

}

func addi(num1, num2 int) int {

    Sum := num1 + num2

    return Sum

}

func subs(num1, num2 int) int {

    Sub := num1 - num2

    return Sub

}

func multi(num1, num2 int) int {

    Mul := num1 \* num2

    return Mul

}

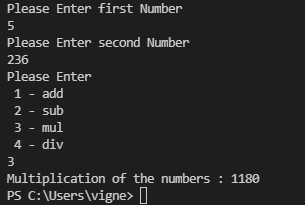
func div(num1, num2 int) int {

    Divi := num1 / num2

    return Divi

}

Output:



1. Writing different Loop structures for accessing arrays/ lists/ strings etc.

Solution:

package main

import (

    "container/list"

    "fmt"

)

func main() {

    arr := [5]float64{25.0, 45.36, 89.7, 12.3, 69.4}

    var str string = "Hai"

    ls := list.New()

    ls.PushBack(10)

    ls.PushBack(12)

    ls.PushBack("Hai")

    ls.PushBack(45.69)

    strs(str)

    arrays(arr)

    fmt.Println("Elements of the list:")

    for e := ls.Front(); e != nil; e = e.Next() {

        fmt.Println(e)

    }

}

func arrays(arr [5]float64) {

    fmt.Println("Elements of the array:")

    for i := 0; i < 5; i++ {

        fmt.Printf("%f\n", arr[i])

    }

}

func strs(str string) {

    for i := 0; i < len(str); i++ {

        fmt.Printf("Character at %d Index Position = %c\n", i, str[i])

    }

}

Output:

