**Practical slips Go and iot**

**Slip 1**

//A) Write a program in GO language to accept user choice and print answers

//using arithmetic operators.

package main

import (

    "fmt"

)

func main() {

    var num1, num2 float64

    var operator string

    // Accepting user input

    fmt.Print("Enter first number: ")

    fmt.Scanln(&num1)

    fmt.Print("Enter an operator (+, -, \*, /, %): ")

    fmt.Scanln(&operator)

    fmt.Print("Enter second number: ")

    fmt.Scanln(&num2)

    // Performing the operation

    switch operator {

    case "+":

        fmt.Printf("Result: %.2f\n", num1+num2)

    case "-":

        fmt.Printf("Result: %.2f\n", num1-num2)

    case "\*":

        fmt.Printf("Result: %.2f\n", num1\*num2)

    case "/":

        if num2 != 0 {

            fmt.Printf("Result: %.2f\n", num1/num2)

        } else {

            fmt.Println("Error: Division by zero is not allowed.")

        }

    case "%":

        if num2 != 0 {

            fmt.Printf("Result: %.0f\n", float64(int(num1)%int(num2)))

        } else {

            fmt.Println("Error: Modulo by zero is not allowed.")

        }

    default:

        fmt.Println("Invalid operator. Please use +, -, \*, /, or %.")

    }

}

OR

//B) Write a program 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 (

    "fmt"

)

type Student struct {

    RollNo   int

    Name     string

    Mark1    float64

    Mark2    float64

    Mark3    float64

    Total    float64

    Average  float64

}

func main() {

    var n int

    fmt.Print("Enter the number of students: ")

    fmt.Scanln(&n)

    students := make([]Student, n)

    for i := 0; i < n; i++ {

        fmt.Printf("\nEnter details for student %d:\n", i+1)

        fmt.Print("Roll No: ")

        fmt.Scanln(&students[i].RollNo)

        fmt.Print("Name: ")

        fmt.Scanln(&students[i].Name)

        fmt.Print("Mark 1: ")

        fmt.Scanln(&students[i].Mark1)

        fmt.Print("Mark 2: ")

        fmt.Scanln(&students[i].Mark2)

        fmt.Print("Mark 3: ")

        fmt.Scanln(&students[i].Mark3)

        // Calculate total and average

        students[i].Total = students[i].Mark1 + students[i].Mark2 + students[i].Mark3

        students[i].Average = students[i].Total / 3

    }

    // Display student details

    fmt.Println("\nStudent Details:")

    fmt.Println("RollNo\tName\tTotal\tAverage")

    for \_, student := range students {

        fmt.Printf("%d\t%s\t%.2f\t%.2f\n", student.RollNo, student.Name, student.Total, student.Average)

    }

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to blink LED. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

void setup() {

// initialize digital pin LED\_BUILTIN as an output. pinMode(LED\_BUILTIN, OUTPUT); } // the loop function runs over and over again forever

void loop() {

digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level) delay(1000); // wait for a second digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW delay(1000); // wait for a second

}

**Slip 2**

//A) Write a program in GO language to print Fibonacci series of n terms.

package main

import (

    "fmt"

)

func fibonacci(n int) {

    first, second := 0, 1

    for i := 0; i < n; i++ {

        fmt.Print(first, " ")

        temp := first + second

        first = second

        second = temp

    }

    fmt.Println()

}

func main() {

    var n int

    fmt.Print("Enter the number of terms: ")

    fmt.Scanln(&n)

    fmt.Println("Fibonacci Series:")

    fibonacci(n)

}

//B) Write a program in GO language to print file information.

package main

import (

    "fmt"

    "os"

)

func main() {

    var fileName string

    fmt.Print("Enter the file name: ")

    fmt.Scanln(&fileName)

    fileInfo, err := os.Stat(fileName)

    if err != nil {

        fmt.Println("Error:", err)

        return

    }

    fmt.Println("File Information:")

    fmt.Println("Name:", fileInfo.Name())

    fmt.Println("Size:", fileInfo.Size(), "bytes")

    fmt.Println("Mode:", fileInfo.Mode())

    fmt.Println("Last Modified:", fileInfo.ModTime())

    fmt.Println("Is Directory:", fileInfo.IsDir())

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to turn ON/OFF buzzer. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

// Define the buzzer pin const int buzzerPin = 9;

// Connect the piezo buzzer to pin 9

// Define the frequency for the "tick" sound (in Hz)

const int tickFrequency = 1000;

// Frequency of the tick sound (1000 Hz)

// Define the duration for the "tick" sound

const int tickDuration =200; // Duration of each tick sound (in milliseconds) void setup() { // Set the buzzer pin as an output pinMode(buzzerPin, OUTPUT); }

OR

const int buzzerPin = 9; // Connect the positive pin of the buzzer to pin 9 void setup() {

pinMode(buzzerPin, OUTPUT); }

void loop() {

tone(buzzerPin, 1000); // Play a 1 kHz tone delay(100); // Duration of the tick noTone(buzzerPin); // Stop the tone delay(900); //Wait before the next tick }

Slip 3

//A) Write a program in the GO language using function to check whether accepts number is palindrome or not

package main

import (

    "fmt"

)

// Function to check if a number is a palindrome

func isPalindrome(num int) bool {

    reverse, temp := 0, num

    for temp > 0 {

        digit := temp % 10

        reverse = reverse\*10 + digit

        temp /= 10

    }

    return num == reverse

}

func main() {

    var num int

    fmt.Print("Enter a number: ")

    fmt.Scanln(&num)

    if isPalindrome(num) {

        fmt.Println(num, "is a palindrome.")

    } else {

        fmt.Println(num, "is not a palindrome.")

    }

}

//B) Write a Program in GO language to accept n records of employee

//information (eno,ename,salary) and display record of employees

//having maximum salary.

package main

import (

    "fmt"

)

type Employee struct {

    Eno    int

    Ename  string

    Salary float64

}

func main() {

    var n int

    fmt.Print("Enter the number of employees: ")

    fmt.Scanln(&n)

    employees := make([]Employee, n)

    var maxSalary float64

    for i := 0; i < n; i++ {

        fmt.Printf("\nEnter details for employee %d:\n", i+1)

        fmt.Print("Employee Number: ")

        fmt.Scanln(&employees[i].Eno)

        fmt.Print("Employee Name: ")

        fmt.Scanln(&employees[i].Ename)

        fmt.Print("Salary: ")

        fmt.Scanln(&employees[i].Salary)

        if employees[i].Salary > maxSalary {

            maxSalary = employees[i].Salary

        }

    }

    fmt.Println("\nEmployee(s) with Maximum Salary:")

    for \_, emp := range employees {

        if emp.Salary == maxSalary {

            fmt.Printf("Eno: %d, Name: %s, Salary: %.2f\n", emp.Eno, emp.Ename, emp.Salary)

        }

    }

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to blink LED. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

void setup() { // initialize digital pin LED\_BUILTIN as an output. pinMode(LED\_BUILTIN, OUTPUT); } // the loop function runs over and over again forever void loop() { digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level) delay(1000); // wait for a second digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW delay(1000); // wait for a second }

Slip 4

//. A) Write a program in GO language to print a recursive sum of digits of a given number.

package main

import (

    "fmt"

)

// Recursive function to calculate the sum of digits

func sumOfDigits(n int) int {

    if n == 0 {

        return 0

    }

    return (n % 10) + sumOfDigits(n / 10)

}

func main() {

    var num int

    fmt.Print("Enter a number: ")

    fmt.Scanln(&num)

    sum := sumOfDigits(num)

    fmt.Println("Sum of digits:", sum)

}

//B) Write a program in GO language to sort array elements in ascending order

package main

import (

    "fmt"

    "sort"

)

func main() {

    var n int

    fmt.Print("Enter the number of elements: ")

    fmt.Scanln(&n)

    numbers := make([]int, n)

    fmt.Println("Enter the elements:")

    for i := 0; i < n; i++ {

        fmt.Scanln(&numbers[i])

    }

    sort.Ints(numbers)

    fmt.Println("Sorted array in ascending order:", numbers)

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to toggle two LED’s. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

/ Define the LED pins const int led1 = 13; // Built-in LED (usually on pin 13) const int led2 = 2; // External LED connected to pin 2 void setup() { // Initialize both pins as output pinMode(led1, OUTPUT); pinMode(led2, OUTPUT); } void loop() { // Turn on LED1 and turn off LED2 digitalWrite(led1, HIGH); digitalWrite(led2, LOW); delay(1000); // Wait for 1 second // Turn off LED1 and turn on LED2 digitalWrite(led1, LOW); digitalWrite(led2, HIGH); delay(1000); // Wait for 1 second }

Slip 5

// A) Write a program in GO language program to create Text file

package main

import (

    "fmt"

    "os"

)

func main() {

    var fileName string

    fmt.Print("Enter the file name to create: ")

    fmt.Scanln(&fileName)

    // Create the file

    file, err := os.Create(fileName)

    if err != nil {

        fmt.Println("Error creating file:", err)

        return

    }

    defer file.Close()

    fmt.Println("File created successfully.")

}

//B) Write a program in GO language to accept n records of employee

//information (eno,ename,salary) and display records of employees having minimum salary.

package main

import (

    "fmt"

)

type Employee struct {

    Eno    int

    Ename  string

    Salary float64

}

func main() {

    var n int

    fmt.Print("Enter the number of employees: ")

    fmt.Scanln(&n)

    employees := make([]Employee, n)

    var minSalary float64 = -1

    for i := 0; i < n; i++ {

        fmt.Printf("\nEnter details for employee %d:\n", i+1)

        fmt.Print("Employee Number: ")

        fmt.Scanln(&employees[i].Eno)

        fmt.Print("Employee Name: ")

        fmt.Scanln(&employees[i].Ename)

        fmt.Print("Salary: ")

        fmt.Scanln(&employees[i].Salary)

        if minSalary == -1 || employees[i].Salary < minSalary {

            minSalary = employees[i].Salary

        }

    }

    fmt.Println("\nEmployee(s) with Minimum Salary:")

    for \_, emp := range employees {

        if emp.Salary == minSalary {

            fmt.Printf("Eno: %d, Name: %s, Salary: %.2f\n", emp.Eno, emp.Ename, emp.Salary)

        }

    }

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to blink LED. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

void setup() { // initialize digital pin LED\_BUILTIN as an output. pinMode(LED\_BUILTIN, OUTPUT); } // the loop function runs over and over again forever void loop() { digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level) delay(1000); // wait for a second digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW delay(1000); // wait for a second }

Slip 6

//Write a program in GO language to accept two matrices and display its multiplication

package main

import (

    "fmt"

)

func main() {

    var r1, c1, r2, c2 int

    fmt.Print("Enter rows and columns for first matrix: ")

    fmt.Scanln(&r1, &c1)

    fmt.Print("Enter rows and columns for second matrix: ")

    fmt.Scanln(&r2, &c2)

    if c1 != r2 {

        fmt.Println("Matrix multiplication is not possible with given dimensions.")

        return

    }

    // Initialize matrices

    matrix1 := make([][]int, r1)

    matrix2 := make([][]int, r2)

    result := make([][]int, r1)

    fmt.Println("Enter elements for first matrix:")

    for i := 0; i < r1; i++ {

        matrix1[i] = make([]int, c1)

        for j := 0; j < c1; j++ {

            fmt.Scan(&matrix1[i][j])

        }

    }

    fmt.Println("Enter elements for second matrix:")

    for i := 0; i < r2; i++ {

        matrix2[i] = make([]int, c2)

        for j := 0; j < c2; j++ {

            fmt.Scan(&matrix2[i][j])

        }

    }

    // Initialize result matrix

    for i := 0; i < r1; i++ {

        result[i] = make([]int, c2)

        for j := 0; j < c2; j++ {

            for k := 0; k < c1; k++ {

                result[i][j] += matrix1[i][k] \* matrix2[k][j]

            }

        }

    }

    // Display result matrix

    fmt.Println("Resultant Matrix:")

    for i := 0; i < r1; i++ {

        for j := 0; j < c2; j++ {

            fmt.Print(result[i][j], " ")

        }

        fmt.Println()

    }

}

//B) Write a program in GO language to copy all elements of one array into another using a method.

package main

import (

    "fmt"

)

// Function to copy elements from one array to another

func copyArray(source []int) []int {

    destination := make([]int, len(source))

    copy(destination, source)

    return destination

}

func main() {

    var n int

    fmt.Print("Enter the number of elements: ")

    fmt.Scanln(&n)

    sourceArray := make([]int, n)

    fmt.Println("Enter the elements:")

    for i := 0; i < n; i++ {

        fmt.Scanln(&sourceArray[i])

    }

    destinationArray := copyArray(sourceArray)

    fmt.Println("Copied array:", destinationArray)

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to toggle two LED’s. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

/ Define the LED pins const int led1 = 13; // Built-in LED (usually on pin 13) const int led2 = 2; // External LED connected to pin 2 void setup() { // Initialize both pins as output pinMode(led1, OUTPUT); pinMode(led2, OUTPUT); } void loop() { // Turn on LED1 and turn off LED2 digitalWrite(led1, HIGH); digitalWrite(led2, LOW); delay(1000); // Wait for 1 second // Turn off LED1 and turn on LED2 digitalWrite(led1, LOW); digitalWrite(led2, HIGH); delay(1000); // Wait for 1 second }

Slip 7

//A) Write a program in GO language to accept one matrix and display its transpose.

package main

import (

    "fmt"

)

func main() {

    var rows, cols int

    fmt.Print("Enter the number of rows and columns: ")

    fmt.Scanln(&rows, &cols)

    matrix := make([][]int, rows)

    transpose := make([][]int, cols)

    fmt.Println("Enter the elements of the matrix:")

    for i := 0; i < rows; i++ {

        matrix[i] = make([]int, cols)

        for j := 0; j < cols; j++ {

            fmt.Scan(&matrix[i][j])

        }

    }

    // Compute transpose

    for i := 0; i < cols; i++ {

        transpose[i] = make([]int, rows)

        for j := 0; j < rows; j++ {

            transpose[i][j] = matrix[j][i]

        }

    }

    // Display transpose matrix

    fmt.Println("Transpose of the matrix:")

    for i := 0; i < cols; i++ {

        for j := 0; j < rows; j++ {

            fmt.Print(transpose[i][j], " ")

        }

        fmt.Println()

    }

}

//B) Write a program in GO language to create structure student. Writea

//method show() whose receiver is a pointer of struct student.

package main

import (

    "fmt"

)

type Student struct {

    RollNo int

    Name   string

    Marks  float64

}

// Method to display student details

func (s \*Student) Show() {

    fmt.Println("Student Details:")

    fmt.Println("Roll No:", s.RollNo)

    fmt.Println("Name:", s.Name)

    fmt.Println("Marks:", s.Marks)

}

func main() {

    var s Student

    fmt.Print("Enter Roll No: ")

    fmt.Scanln(&s.RollNo)

    fmt.Print("Enter Name: ")

    fmt.Scanln(&s.Name)

    fmt.Print("Enter Marks: ")

    fmt.Scanln(&s.Marks)

    s.Show()

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to turn ON/OFF buzzer. c. Write down the observations on Input an

=>

// Define the buzzer pin const int buzzerPin = 9; // Connect the piezo buzzer to pin 9 // Define the frequency for the "tick" sound (in Hz) const int tickFrequency = 1000; // Frequency of the tick sound (1000 Hz) // Define the duration for the "tick" sound const int tickDuration =200; // Duration of each tick sound (in milliseconds) void setup() { // Set the buzzer pin as an output pinMode(buzzerPin, OUTPUT); }

OR

const int buzzerPin = 9; // Connect the positive pin of the buzzer to pin 9 void setup() { pinMode(buzzerPin, OUTPUT); } void loop() { tone(buzzerPin, 1000); // Play a 1 kHz tone delay(100); // Duration of the tick noTone(buzzerPin); // Stop the tone delay(900); // Wait before the next tick }

Slip 8

//A) Write a program 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 (

    "fmt"

)

type Book struct {

    BookID int

    Title  string

    Author string

    Price  float64

}

func main() {

    var n int

    fmt.Print("Enter the number of books: ")

    fmt.Scanln(&n)

    books := make([]Book, n)

    for i := 0; i < n; i++ {

        fmt.Printf("\nEnter details for book %d:\n", i+1)

        fmt.Print("Book ID: ")

        fmt.Scanln(&books[i].BookID)

        fmt.Print("Title: ")

        fmt.Scanln(&books[i].Title)

        fmt.Print("Author: ")

        fmt.Scanln(&books[i].Author)

        fmt.Print("Price: ")

        fmt.Scanln(&books[i].Price)

    }

    fmt.Println("\nBook Details:")

    for \_, book := range books {

        fmt.Printf("Book ID: %d, Title: %s, Author: %s, Price: %.2f\n", book.BookID, book.Title, book.Author, book.Price)

    }

}

//B) 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 (

    "fmt"

    "math"

)

type Shape interface {

    Area() float64

    Perimeter() float64

}

type Circle struct {

    Radius float64

}

type Rectangle struct {

    Length, Width float64

}

func (c Circle) Area() float64 {

    return math.Pi \* c.Radius \* c.Radius

}

func (c Circle) Perimeter() float64 {

    return 2 \* math.Pi \* c.Radius

}

func (r Rectangle) Area() float64 {

    return r.Length \* r.Width

}

func (r Rectangle) Perimeter() float64 {

    return 2 \* (r.Length + r.Width)

}

func main() {

    var c Circle

    var r Rectangle

    fmt.Print("Enter radius of circle: ")

    fmt.Scanln(&c.Radius)

    fmt.Println("Circle Area:", c.Area())

    fmt.Println("Circle Perimeter:", c.Perimeter())

    fmt.Print("\nEnter length and width of rectangle: ")

    fmt.Scanln(&r.Length, &r.Width)

    fmt.Println("Rectangle Area:", r.Area())

    fmt.Println("Rectangle Perimeter:", r.Perimeter())

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to blink LED. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

void setup() { // initialize digital pin LED\_BUILTIN as an output. pinMode(LED\_BUILTIN, OUTPUT); } // the loop function runs over and over again forever void loop() { digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level) delay(1000); // wait for a second digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW delay(1000); // wait for a second }

Slip 9

//A) Write a program in GO language using a function to check

//whether the accepted number is palindrome or not.

package main

import (

    "fmt"

)

// Function to check if a number is palindrome

func isPalindrome(n int) bool {

    reverse, temp := 0, n

    for temp > 0 {

        digit := temp % 10

        reverse = reverse\*10 + digit

        temp /= 10

    }

    return n == reverse

}

func main() {

    var num int

    fmt.Print("Enter a number: ")

    fmt.Scanln(&num)

    if isPalindrome(num) {

        fmt.Println(num, "is a palindrome.")

    } else {

        fmt.Println(num, "is not a palindrome.")

    }

}

//B) Write a program in GO language to create an interface shape that

//includes area and volume. Implements these methods in square and rectangle type.

package main

import (

    "fmt"

)

type Shape interface {

    Area() float64

    Volume() float64

}

type Square struct {

    Side float64

}

type Rectangle struct {

    Length, Width, Height float64

}

func (s Square) Area() float64 {

    return s.Side \* s.Side

}

func (s Square) Volume() float64 {

    return s.Side \* s.Side \* s.Side

}

func (r Rectangle) Area() float64 {

    return r.Length \* r.Width

}

func (r Rectangle) Volume() float64 {

    return r.Length \* r.Width \* r.Height

}

func main() {

    var s Square

    var r Rectangle

    fmt.Print("Enter side of square: ")

    fmt.Scanln(&s.Side)

    fmt.Println("Square Area:", s.Area())

    fmt.Println("Square Volume:", s.Volume())

    fmt.Print("\nEnter length, width, and height of rectangle: ")

    fmt.Scanln(&r.Length, &r.Width, &r.Height)

    fmt.Println("Rectangle Area:", r.Area())

    fmt.Println("Rectangle Volume:", r.Volume())

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to blink LED. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

void setup() { // initialize digital pin LED\_BUILTIN as an output. pinMode(LED\_BUILTIN, OUTPUT); } // the loop function runs over and over again forever void loop() { digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level) delay(1000); // wait for a second digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW delay(1000); // wait for a second }

Slip 10

//A) Write a program in GO language to create  an interface and displayits values with the help of type assertion.

package main

import (

    "fmt"

)

type Data interface{}

func displayValue(d Data) {

    switch v := d.(type) {

    case int:

        fmt.Println("Integer value:", v)

    case float64:

        fmt.Println("Float value:", v)

    case string:

        fmt.Println("String value:", v)

    default:

        fmt.Println("Unknown type")

    }

}

func main() {

    var d Data

    d = 42

    displayValue(d)

    d = 3.14

    displayValue(d)

    d = "Hello, Go!"

    displayValue(d)

}

//B) Write a program in GO language to read and write Fibonacci series to the using channel.

package main

import (

    "fmt"

)

// Function to generate Fibonacci series and send values to channel

func fibonacci(n int, ch chan int) {

    a, b := 0, 1

    for i := 0; i < n; i++ {

        ch <- a

        a, b = b, a+b

    }

    close(ch)

}

func main() {

    var n int

    fmt.Print("Enter the number of Fibonacci terms: ")

    fmt.Scanln(&n)

    ch := make(chan int, n)

    go fibonacci(n, ch)

    fmt.Println("Fibonacci Series:")

    for num := range ch {

        fmt.Print(num, " ")

    }

    fmt.Println()

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to turn ON/OFF buzzer. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

// Define the buzzer pin const int buzzerPin = 9; // Connect the piezo buzzer to pin 9 // Define the frequency for the "tick" sound (in Hz) const int tickFrequency = 1000; // Frequency of the tick sound (1000 Hz) // Define the duration for the "tick" sound const int tickDuration =200; // Duration of each tick sound (in milliseconds) void setup() { // Set the buzzer pin as an output pinMode(buzzerPin, OUTPUT); }

OR

const int buzzerPin = 9; // Connect the positive pin of the buzzer to pin 9 void setup() { pinMode(buzzerPin, OUTPUT); } void loop() { tone(buzzerPin, 1000); // Play a 1 kHz tone delay(100); // Duration of the tick noTone(buzzerPin); // Stop the tone delay(900); // Wait before the next tick }

**Slip 11**

Q1. A) Write a program in GO language to check whether the accepted number is two digit or not.

Ans- package main

import (

"fmt"

)

func main() {

var num int

// Accepting user input

fmt.Print("Enter a number: ")

fmt.Scan(&num)

// Checking if the number is a two-digit number

if num >= 10 && num <= 99 || num <= -10 && num >= -99 {

fmt.Println("The number is a two-digit number.")

} else {

fmt.Println("The number is NOT a two-digit number.")

}

}

OR

B) 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.

Ans- package main

import (

"fmt"

)

func main() {

ch := make(chan int, 5)

ch <- 10

ch <- 20

ch <- 30

fmt.Println("Channel Capacity:", cap(ch))

fmt.Println("Channel Length before reading:", len(ch))

fmt.Println("Reading values from channel:")

fmt.Println(<-ch) // Reads 10

fmt.Println(<-ch) // Reads 20

fmt.Println("Channel Length after reading:", len(ch)) }

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to turn ON/OFF buzzer. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

// Define the buzzer pin const int buzzerPin = 9; // Connect the piezo buzzer to pin 9 // Define the frequency for the "tick" sound (in Hz) const int tickFrequency = 1000; // Frequency of the tick sound (1000 Hz) // Define the duration for the "tick" sound const int tickDuration =200; // Duration of each tick sound (in milliseconds) void setup() { // Set the buzzer pin as an output pinMode(buzzerPin, OUTPUT); }

OR

const int buzzerPin = 9; // Connect the positive pin of the buzzer to pin 9 void setup() { pinMode(buzzerPin, OUTPUT); } void loop() { tone(buzzerPin, 1000); // Play a 1 kHz tone delay(100); // Duration of the tick noTone(buzzerPin); // Stop the tone delay(900); // Wait before the next tick }

**Slip 12**

Q1. A) Write a program in GO language to swap two numbers using call by reference concept.

Ans- package main

import (

"fmt")

func swap(a, b \*int) {

\*a, \*b = \*b, \*a

}

func main() {

var num1, num2 int

// Accepting input

fmt.Print("Enter first number: ")

fmt.Scan(&num1)

fmt.Print("Enter second number: ")

fmt.Scan(&num2)

fmt.Println("Before Swapping:", num1, num2)

// Passing addresses of variables

swap(&num1, &num2)

fmt.Println("After Swapping:", num1, num2)

}

OR

B) Write a program in GO language 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.

Ans-

package main

import (

"fmt"

)

func evenNumbers(ch <-chan int) {

for num := range ch {

fmt.Println("Even number received:", num)

}

}

func oddNumbers(ch <-chan int) {

for num := range ch {

fmt.Println("Odd number received:", num)

}

}

func main() {

numbers := []int{10, 23, 45, 12, 8, 33, 50}

evenCh := make(chan int)

oddCh := make(chan int)

go evenNumbers(evenCh)

go oddNumbers(oddCh)

for \_, num := range numbers {

if num%2 == 0 {

evenCh <- num

} else {

oddCh <- num

}

}

close(evenCh)

close(oddCh)

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to toggle two LED’s. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

/ Define the LED pins const int led1 = 13; // Built-in LED (usually on pin 13) const int led2 = 2; // External LED connected to pin 2 void setup() { // Initialize both pins as output pinMode(led1, OUTPUT); pinMode(led2, OUTPUT); } void loop() { // Turn on LED1 and turn off LED2 digitalWrite(led1, HIGH); digitalWrite(led2, LOW); delay(1000); // Wait for 1 second // Turn off LED1 and turn on LED2 digitalWrite(led1, LOW); digitalWrite(led2, HIGH); delay(1000); // Wait for 1 second }

**Slip 13**

Q1. A) Write a program in GO language to print sum of all even and odd numbers separately between 1 to 100.

Ans- package main

import "fmt"

func main() {

    // Variables to store sums

    evenSum, oddSum := 0, 0

    // Loop from 1 to 100

    for i := 1; i <= 100; i++ {

        if i%2 == 0 {

            evenSum += i // Add even numbers

        } else {

            oddSum += i // Add odd numbers

        }

    }

    fmt.Println("Sum of Even Numbers (1-100):", evenSum)

    fmt.Println("Sum of Odd Numbers (1-100):", oddSum)

}

OR

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

Ans- package main

import (

    "fmt"

    "testing"

)

func square(n int) int {

    return n \* n

}

func main() {

    var num int

    fmt.Print("Enter a number: ")

    fmt.Scan(&num)

    fmt.Println("Square of", num, "is:", square(num))

}

func BenchmarkSquare(b \*testing.B) {

    for i := 0; i < b.N; i++ {

        square(10)

    }

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to toggle two LED’s. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

/ Define the LED pins const int led1 = 13; // Built-in LED (usually on pin 13) const int led2 = 2; // External LED connected to pin 2 void setup() { // Initialize both pins as output pinMode(led1, OUTPUT); pinMode(led2, OUTPUT); } void loop() { // Turn on LED1 and turn off LED2 digitalWrite(led1, HIGH); digitalWrite(led2, LOW); delay(1000); // Wait for 1 second // Turn off LED1 and turn on LED2 digitalWrite(led1, LOW); digitalWrite(led2, HIGH); delay(1000); // Wait for 1 second }

**Slip 14**

Q1. A) Write a program in GO language to demonstrate working of slices (like append, remove, copy etc.)

Ans- package main

import (

"fmt"

func main() {

numbers := []int{10, 20, 30, 40, 50}

fmt.Println("Initial Slice:", numbers)

// Append elements

numbers = append(numbers, 60, 70)

fmt.Println("After Append:", numbers)

// Remove an element (removing element at index 2)

indexToRemove := 2

numbers = append(numbers[:indexToRemove], numbers[indexToRemove+1:]...)

fmt.Println("After Removing Element at Index 2:", numbers)

newSlice := make([]int, len(numbers))

copy(newSlice, numbers)

fmt.Println("Copied Slice:", newSlice)

subSlice := numbers[1:4]

fmt.Println("Sub-Slice [1:4]:", subSlice)

}

B) Write a program in GO language 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).

Ans- package main

import (

    "fmt"

    "strconv"

)

func sumOfSquares(num int, ch chan int) {

    sum := 0

    for \_, digit := range strconv.Itoa(num) {

        d := int(digit - '0')

        sum += d \* d

    }

    ch <- sum // Send result to channel

}

// Function to calculate sum of cubes

func sumOfCubes(num int, ch chan int) {

    sum := 0

    for \_, digit := range strconv.Itoa(num) {

        d := int(digit - '0')

        sum += d \* d \* d

    }

    ch <- sum // Send result to channel

}

func main() {

    var num int

    fmt.Print("Enter a number: ")

    fmt.Scan(&num)

    squareCh := make(chan int)

    cubeCh := make(chan int)

    go sumOfSquares(num, squareCh)

    go sumOfCubes(num, cubeCh)

    squares := <-squareCh

    cubes := <-cubeCh

    fmt.Println("Sum of Squares:", squares)

    fmt.Println("Sum of Cubes:", cubes)

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to turn ON/OFF buzzer. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

// Define the buzzer pin const int buzzerPin = 9; // Connect the piezo buzzer to pin 9 // Define the frequency for the "tick" sound (in Hz) const int tickFrequency = 1000; // Frequency of the tick sound (1000 Hz) // Define the duration for the "tick" sound const int tickDuration =200; // Duration of each tick sound (in milliseconds) void setup() { // Set the buzzer pin as an output pinMode(buzzerPin, OUTPUT); }

OR

const int buzzerPin = 9; // Connect the positive pin of the buzzer to pin 9 void setup() { pinMode(buzzerPin, OUTPUT); } void loop() { tone(buzzerPin, 1000); // Play a 1 kHz tone delay(100); // Duration of the tick noTone(buzzerPin); // Stop the tone delay(900); // Wait before the next tick }

**Slip 15**

Q1. A) Write a program in GO language to demonstrate function return multiple values.

Ans-

package main

import "fmt"

func calculate(a, b int) (int, int, int, float64) {

sum := a + b

diff := a - b

product := a \* b

quotient := float64(a) / float64(b) // Type conversion for division

return sum, diff, product, quotient

}

func main() {

sum, diff, product, quotient := calculate(10, 2)

fmt.Println("Sum:", sum)

fmt.Println("Difference:", diff)

fmt.Println("Product:", product)

fmt.Println("Quotient:", quotient)

}

OR

B) Write a program in GO language to read XML file into structure and display structure.

Ans-

package main

import (

    "encoding/xml"

    "fmt"

    "io/ioutil"

    "os"

)

// Structs for XML Parsing

type Employee struct {

    ID       int    `xml:"ID"`

    Name     string `xml:"Name"`

    Position string `xml:"Position"`

}

type Employees struct {

    EmployeeList []Employee `xml:"Employee"`

}

func main() {

    // Open XML file

    file, err := os.Open("D:/GO Program/Assignment 4/Method/practicalSlips/data.xml")

    if err != nil {

   fmt.Println("Error opening file:", err)

        return

    }

    defer file.Close()

    // Read file content

    data, err := ioutil.ReadAll(file)

    if err != nil {

        fmt.Println("Error reading file:", err)

        return

    }

    // Parse XML

    var employees Employees

    err = xml.Unmarshal(data, &employees)

    if err != nil {

        fmt.Println("Error unmarshaling XML:", err)

        return

    }

    // Display parsed data

    fmt.Println("Employee Details:")

    for \_, emp := range employees.EmployeeList {

        fmt.Printf("ID: %d, Name: %s, Position: %s\n", emp.ID, emp.Name, emp.Position)

    }

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to toggle two LED’s. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

/ Define the LED pins const int led1 = 13; // Built-in LED (usually on pin 13) const int led2 = 2; // External LED connected to pin 2 void setup() { // Initialize both pins as output pinMode(led1, OUTPUT); pinMode(led2, OUTPUT); } void loop() { // Turn on LED1 and turn off LED2 digitalWrite(led1, HIGH); digitalWrite(led2, LOW); delay(1000); // Wait for 1 second // Turn off LED1 and turn on LED2 digitalWrite(led1, LOW); digitalWrite(led2, HIGH); delay(1000); // Wait for 1 second }

**Slip 16**

B) Write a program in GO language that prints out the numbers from0 to 10, waiting between 0 and 250 ms after each one using the delay function.

Ans- package main

import (

    "fmt"

    "math/rand"

    "time"

)

func main() {

    rand.Seed(time.Now().UnixNano()) // Seed the random number generator

    for i := 0; i <= 10; i++ {

        fmt.Println(i)

        // Generate a random delay between 0 and 250 ms

        delay := time.Duration(rand.Intn(251)) \* time.Millisecond

        time.Sleep(delay) // Pause execution

    }

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to blink LED. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

void setup() { // initialize digital pin LED\_BUILTIN as an output. pinMode(LED\_BUILTIN, OUTPUT); } // the loop function runs over and over again forever void loop() { digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level) delay(1000); // wait for a second digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW delay(1000); // wait for a second }

**Slip 17**

Q1. A) Write a program in GO language to illustrate the concept of returning multiple values from a function. ( Add, Subtract, Multiply, Divide)

Ans-package main

import (

    "fmt"

)

// Function to perform calculations and return multiple values

func calculate(a, b float64) (float64, float64, float64, float64) {

    add := a + b

    subtract := a - b

    multiply := a \* b

    divide := a / b

    return add, subtract, multiply, divide

}

func main() {

    var num1, num2 float64

    // Taking user input

    fmt.Print("Enter first number: ")

    fmt.Scan(&num1)

    fmt.Print("Enter second number: ")

    fmt.Scan(&num2)

    // Calling function and receiving multiple values

    add, subtract, multiply, divide := calculate(num1, num2)

    // Displaying results

    fmt.Println("Addition:", add)

    fmt.Println("Subtraction:", subtract)

    fmt.Println("Multiplication:", multiply)

    fmt.Println("Division:", divide)

}

OR

B) Write a program in GO language to add or append content at the end of a text file

Ans- package main

import (

    "fmt"

    "os"

)

func main() {

    // File name

    filename := "example.txt"

    // Content to append

    content := "\nThis is new appended content."

    // Open file in append mode, create it if it doesn't exist

    file, err := os.OpenFile(filename, os.O\_APPEND|os.O\_CREATE|os.O\_WRONLY, 0644)

    if err != nil {

        fmt.Println("Error opening file:", err)

        return

    }

    defer file.Close()

    // Writing to file

    \_, err = file.WriteString(content)

    if err != nil {

        fmt.Println("Error writing to file:", err)

        return

    }

    fmt.Println("Content appended successfully!")

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to toggle two LED’s. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

/ Define the LED pins const int led1 = 13; // Built-in LED (usually on pin 13) const int led2 = 2; // External LED connected to pin 2 void setup() { // Initialize both pins as output pinMode(led1, OUTPUT); pinMode(led2, OUTPUT); } void loop() { // Turn on LED1 and turn off LED2 digitalWrite(led1, HIGH); digitalWrite(led2, LOW); delay(1000); // Wait for 1 second // Turn off LED1 and turn on LED2 digitalWrite(led1, LOW); digitalWrite(led2, HIGH); delay(1000); // Wait for 1 second }

**Slip 18**

Q1. A) Write a program in GO language to print a multiplication table of number using function.

Ans- package main

import (

"fmt"

)

// Function to print multiplication table

func printTable(num int) {

fmt.Println("Multiplication Table of", num)

for i := 1; i <= 10; i++ {

fmt.Printf("%d x %d = %d\n", num, i, num\*i)

}

}

func main() {

var num int

// Taking user input

fmt.Print("Enter a number: ")

fmt.Scan(&num)

// Calling the function

printTable(num)

}

OR

B) Write a program in GO language using a user defined package calculator that performs one calculator operation as per the user's choice.

Ans-

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to turn ON/OFF buzzer. c. Write down the observations on Input and Output d. Write down the Result and Conclusion

=>

// Define the buzzer pin const int buzzerPin = 9; // Connect the piezo buzzer to pin 9 // Define the frequency for the "tick" sound (in Hz) const int tickFrequency = 1000; // Frequency of the tick sound (1000 Hz) // Define the duration for the "tick" sound const int tickDuration =200; // Duration of each tick sound (in milliseconds) void setup() { // Set the buzzer pin as an output pinMode(buzzerPin, OUTPUT); }

OR

const int buzzerPin = 9; // Connect the positive pin of the buzzer to pin 9 void setup() { pinMode(buzzerPin, OUTPUT); } void loop() { tone(buzzerPin, 1000); // Play a 1 kHz tone delay(100); // Duration of the tick noTone(buzzerPin); // Stop the tone delay(900); // Wait before the next tick }

**Slip 19**

Q1. A) Write a program in GO language to illustrate the function returning multiple values(add, subtract).

Ans- package main

import (

    "fmt"

)

// Function that returns both addition and subtraction results

func calculate(a, b int) (int, int) {

    add := a + b

    subtract := a - b

    return add, subtract

}

func main() {

    var num1, num2 int

    // Taking user input

    fmt.Print("Enter first number: ")

    fmt.Scan(&num1)

    fmt.Print("Enter second number: ")

    fmt.Scan(&num2)

    // Calling function and receiving multiple values

    addition, subtraction := calculate(num1, num2)

    // Displaying results

    fmt.Println("Addition:", addition)

    fmt.Println("Subtraction:", subtraction)

}

OR

B) Write a program in the GO language program to open a file in READ only mode.

Ans –

package main

import (

    "fmt"

    "os"

)

func main() {

    // File name

    filename := "example.txt"

    // Open file in READ-ONLY mode

    file, err := os.Open(filename)

    if err != nil {

        fmt.Println("Error opening file:", err)

        return

    }

    defer file.Close()

    fmt.Println("File opened successfully in READ-ONLY mode.")

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to turn ON/OFF buzzer. c. Write down the observations on Input and Output. d. Write down the Result and Conclusion.

=>

// Define the buzzer pin const int buzzerPin = 9; // Connect the piezo buzzer to pin 9 // Define the frequency for the "tick" sound (in Hz) const int tickFrequency = 1000; // Frequency of the tick sound (1000 Hz) // Define the duration for the "tick" sound const int tickDuration =200; // Duration of each tick sound (in milliseconds) void setup() { // Set the buzzer pin as an output pinMode(buzzerPin, OUTPUT); }

OR

const int buzzerPin = 9; // Connect the positive pin of the buzzer to pin 9 void setup() { pinMode(buzzerPin, OUTPUT); } void loop() { tone(buzzerPin, 1000); // Play a 1 kHz tone delay(100); // Duration of the tick noTone(buzzerPin); // Stop the tone delay(900); // Wait before the next tick }

**Slip 20**

Q1. A) Write a program in Go language to add or append content at the end of a text file.

Ans-package main

import (

    "fmt"

    "os"

)

func main() {

    // File name

    filename := "example.txt"

    // Content to append

    content := "\nThis is newly appended content."

    // Open file in append mode, create it if it doesn't exist

    file, err := os.OpenFile(filename, os.O\_APPEND|os.O\_CREATE|os.O\_WRONLY, 0644)

    if err != nil {

        fmt.Println("Error opening file:", err)

        return

    }

    defer file.Close()

    // Writing to file

    \_, err = file.WriteString(content)

    if err != nil {

        fmt.Println("Error writing to file:", err)

        return

    }

    fmt.Println("Content appended successfully!"

}

OR

B) 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.

Ans- package main

import (

"fmt"

)

func main() {

// Creating a channel of integers

numbers := make(chan int)

// Goroutine to send values to the channel

go func() {

for i := 1; i <= 5; i++ {

numbers <- i

}

close(numbers) // Closing the channel after sending all values

}()

// Receiving values using for range loop

fmt.Println("Receiving values from channel:")

for num := range numbers {

fmt.Println(num)

}

fmt.Println("Channel closed, exiting program.")

}

Q2. a. Draw block diagram /pin diagram of Raspberry-Pi/ Beagle board /Arduino Uno board interfacing with IR Sensor/Temperature Sensor/Camera. (Internal Examiner assign any one option for board and interface device and respective interface programming option) b. WAP in python/C++ language to toggle two LED’s. c. Write down the observations on Input and Output. d. Write down the Result and Conclusion.

/ Define the LED pins const int led1 = 13; // Built-in LED (usually on pin 13) const int led2 = 2; // External LED connected to pin 2 void setup() { // Initialize both pins as output pinMode(led1, OUTPUT); pinMode(led2, OUTPUT); } void loop() { // Turn on LED1 and turn off LED2 digitalWrite(led1, HIGH); digitalWrite(led2, LOW); delay(1000); // Wait for 1 second // Turn off LED1 and turn on LED2 digitalWrite(led1, LOW); digitalWrite(led2, HIGH); delay(1000); // Wait for 1 second }