Ejercicios de Programación en Go

Sistemas Distribuidos

Jean Carlos William Huancoillo Rojas

17 de octubre de 2025

Índice

1. Ejercicio 1: Cliente TCP

Este programa implementa un cliente TCP que se conecta a un servidor, envía el nombre del usuario y recibe un saludo.

```go
package main

import (
    "bufio"
    "fmt"
    "net"
    "os"
    "strings"
)

func main() {
    fmt.Println("Este es el cliente - Computadora B")

    // PEDIR NOMBRE AL USUARIO
    reader := bufio.NewReader(os.Stdin)
    fmt.Print("Ingrese su nombre: ")
    nombre, _ := reader.ReadString('\n')
    nombre = strings.TrimSpace(nombre)

    // CONECTARSE AL SERVIDOR
    fmt.Println("Conectando al servidor...")
    conn, err := net.Dial("tcp", "192.168.124.140:9000")
    if err != nil {
        fmt.Println("Error al conectar:", err)
        return
    }

    // ENVIAR NOMBRE AL SERVIDOR
    fmt.Println("Enviando nombre al servidor...")
    conn.Write([]byte(nombre))

    // RECIBIR SALUDO DEL SERVIDOR
    buffer := make([]byte, 1024)
    n, _ := conn.Read(buffer)
    respuesta := string(buffer[:n])
    fmt.Println("Respuesta del servidor:", respuesta)

    conn.Close()
}
```

2. Ejercicio 2: Servidor TCP

Este programa implementa un servidor TCP que escucha conexiones, recibe nombres de clientes y envía saludos personalizados.

```go
package main

import (
    "fmt"
    "net"
)
```

```
7
8  func main() {
9      fmt.Println("Este es el servidor de saludos - Computadora HP")
10     fmt.Println("Esperando conexiones...")
11
12     listener, _ := net.Listen("tcp", ":9000")
13     for {
14         conn, _ := listener.Accept()
15         fmt.Println("Cliente conectado.")
16
17         // Leer el nombre del cliente
18         buffer := make([]byte, 1024)
19         n, _ := conn.Read(buffer)
20         nombre := string(buffer[:n])
21         fmt.Println("Nombre recibido:", nombre)
22
23         // Enviar saludo al cliente
24         respuesta := "Hola :) " + nombre + " Bienvenido a Sistemas
       Distribuidos "
25         conn.Write([]byte(respuesta))
26         fmt.Println("Envie saludo")
27         conn.Close()
28     }
29 }
```

3. Ejercicio 3: Importación con Alias

Ejemplo básico de uso de alias en la importación de paquetes.

```
1 package main
2
3 import fat "fmt"
4
5 func main() {
6     fat.Println("Sistemas Distribuidos GOOOOOOD")
7 }
```

4. Ejercicio 4: Variables Básicas

Declaración y uso de variables simples con diferentes tipos de datos.

```
1  package main
2
3  import (
4      "fmt"
5  )
6
7  func main() {
8      nombre := "Andree"
9      apellidos := "Trujillo"
10     peso := 75
11     fmt.Println("Hola, me llamo", nombre, apellidos, "y peso", peso, "Kg
       .")
12     fmt.Println("Adios")
```

```
}
```

5. Ejercicio 5: Tipos de Declaración de Variables

Diferentes formas de declarar variables en Go: declaración corta, larga y múltiple.

```go
package main

import "fmt"

func main() {
    // Definicion de variable corta
    nombre := "Andree"

    // Definicion de variable larga (con var)
    var apellido string = "Trujillo"

    // Definicion de variable multiple
    var (
        peso   = 75
        talla  = 1.74
        estado = true
    )

    // Tipos de variables
    var edad int = 21
    var temperatura float64 = 8.67
    var aprobado bool = true

    fmt.Println("Hola mi nombre es", nombre, apellido, "peso", peso,
        "kg.", "Mido", talla, "m.", estado)
    fmt.Println("Tengo", edad, "a os.", "La temperatura de Puno es",
        temperatura, "grados.", aprobado)
}
```

6. Ejercicio 6: Bucles For

Ejemplos de uso del bucle `for` en Go, incluyendo iteración simple y tabla de multiplicar.

```go
package main

import "fmt"

func main() {
    for i := 1; i <= 10; i++ {
        fmt.Println("Numero:", i)
    }

    numero := 2
    for i := 1; i <= 10; i++ {
        fmt.Printf("%d x %d = %d\n", numero, i, numero*i)
    }
}
```

7. Ejercicio 7: Bucle For como While

En Go no existe `while`, pero se puede simular con `for`. También se muestra el bucle infinito con `break`.

```go
package main

import "fmt"

func main() {
    contador := 0
    for contador < 10 {
        fmt.Println("Contador:", contador)
        contador++
    }

    contador1 := 0

    // FOR infinito con BREAK
    for {
        if contador1 >= 10 {
            break
        }
        fmt.Println("Contador:", contador1)
        contador1++
    }
}
```

8. Ejercicio 8: Sentencia If

Uso de la sentencia condicional `if-else`.

```go
package main

import "fmt"

func main() {
    edad := 16
    if edad >= 18 {
        fmt.Println("Eres legal. :)")
    } else {
        fmt.Println("Eres carcel. :(")
    }
}
```

9. Ejercicio 9: Funciones

Ejemplo de definición y uso de funciones en Go.

```go
package main

import "fmt"

func main() {
```

```
    nombre := "Andree"
    saludar := saludo(nombre)
    fmt.Println(saludar)
}

func saludo(nombre string) string {
    return "Bienvenido a Sistemas Distribuidos " + nombre
}
```

10. Ejercicio 10: Entrada de Usuario con Bufio

Lectura de entrada del usuario utilizando el paquete `bufio` para manejar espacios en los nombres.

```
package main

import (
    "bufio"
    "fmt"
    "os"
    "strings"
)

func main() {
    reader := bufio.NewReader(os.Stdin)
    fmt.Print("Ingrese sus nombres: ")
    nombre, _ := reader.ReadString('\n')
    nombre = strings.TrimSpace(nombre)
    saludar := saludo(nombre)
    fmt.Println(saludar)
}

func saludo(nombre string) string {
    return "Bienvenido a Sistemas Distribuidos " + nombre
}
```