

# Ejercicios de Programación en Go

Sistemas Distribuidos

Andree Trujillo

17 de octubre de 2025

## Índice

1. Ejercicio 1: Cliente TCP	2
2. Ejercicio 2: Servidor TCP	2
3. Ejercicio 3: Importación con Alias	3
4. Ejercicio 4: Variables Básicas	3
5. Ejercicio 5: Tipos de Declaración de Variables	4
6. Ejercicio 6: Bucles For	4
7. Ejercicio 7: Bucle For como While	5
8. Ejercicio 8: Sentencia If	5
9. Ejercicio 9: Funciones	5
10.Ejercicio 10: Entrada de Usuario con Bufio	6

## 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.

```
1 package main
2
3 import (
4     "bufio"
5     "fmt"
6     "net"
7     "os"
8     "strings"
9 )
10
11 func main() {
12     fmt.Println("Este es el cliente - Computadora B")
13
14     // PEDIR NOMBRE AL USUARIO
15     reader := bufio.NewReader(os.Stdin)
16     fmt.Print("Ingrese su nombre: ")
17     nombre, _ := reader.ReadString('\n')
18     nombre = strings.TrimSpace(nombre)
19
20     // CONECTARSE AL SERVIDOR
21     fmt.Println("Conectando al servidor...")
22     conn, err := net.Dial("tcp", "192.168.124.140:9000")
23     if err != nil {
24         fmt.Println("Error al conectar:", err)
25         return
26     }
27
28     // ENVIAR NOMBRE AL SERVIDOR
29     fmt.Println("Enviando nombre al servidor...")
30     conn.Write([]byte(nombre))
31
32     // RECIBIR SALUDO DEL SERVIDOR
33     buffer := make([]byte, 1024)
34     n, _ := conn.Read(buffer)
35     respuesta := string(buffer[:n])
36     fmt.Println("Respuesta del servidor:", respuesta)
37
38     conn.Close()
39 }
```

## 2. Ejercicio 2: Servidor TCP

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

```
1 package main
2
3 import (
4     "fmt"
5     "net"
6 )
```

```
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 G000000D")
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")
13 }
```

13 }

## 5. Ejercicio 5: Tipos de Declaración de Variables

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

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     // Definicion de variable corta
7     nombre := "Andree"
8
9     // Definicion de variable larga (con var)
10    var apellido string = "Trujillo"
11
12    // Definicion de variable multiple
13    var (
14        peso    = 75
15        talla   = 1.74
16        estado = true
17    )
18
19    // Tipos de variables
20    var edad int = 21
21    var temperatura float64 = 8.67
22    var aprobado bool = true
23
24    fmt.Println("Hola mi nombre es", nombre, apellido, "peso", peso,
25               "kg.", "Mido", talla, "m.", estado)
26    fmt.Println("Tengo", edad, "a os.", "La temperatura de Puno es",
27               temperatura, "grados.", aprobado)
28 }
```

## 6. Ejercicio 6: Bucles For

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

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     for i := 1; i <= 10; i++ {
7         fmt.Println("Numero:", i)
8     }
9
10    numero := 2
11    for i := 1; i <= 10; i++ {
12        fmt.Printf("%d x %d = %d\n", numero, i, numero*i)
13    }
14 }
```

## 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`.

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     contador := 0
7     for contador < 10 {
8         fmt.Println("Contador:", contador)
9         contador++
10    }
11
12    contador1 := 0
13
14    // FOR infinito con BREAK
15    for {
16        if contador1 >= 10 {
17            break
18        }
19        fmt.Println("Contador:", contador1)
20        contador1++
21    }
22 }
```

## 8. Ejercicio 8: Sentencia If

Uso de la sentencia condicional `if-else`.

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     edad := 16
7     if edad >= 18 {
8         fmt.Println("Eres legal. :)")
9     } else {
10        fmt.Println("Eres carcel. :(")
11    }
12 }
```

## 9. Ejercicio 9: Funciones

Ejemplo de definición y uso de funciones en Go.

```
1 package main
2
3 import "fmt"
4
5 func main() {
```

```
6     nombre := "Andree"
7     saludar := saludo(nombre)
8     fmt.Println(saludar)
9 }
10
11 func saludo(nombre string) string {
12     return "Bienvenido a Sistemas Distribuidos " + nombre
13 }
```

## 10. Ejercicio 10: Entrada de Usuario con Bufio

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

```
1 package main
2
3 import (
4     "bufio"
5     "fmt"
6     "os"
7     "strings"
8 )
9
10 func main() {
11     reader := bufio.NewReader(os.Stdin)
12     fmt.Print("Ingrese sus nombres: ")
13     nombre, _ := reader.ReadString('\n')
14     nombre = strings.TrimSpace(nombre)
15     saludar := saludo(nombre)
16     fmt.Println(saludar)
17 }
18
19 func saludo(nombre string) string {
20     return "Bienvenido a Sistemas Distribuidos " + nombre
21 }
```