

***Berdo'alah sebelum mengerjakan. Dilarang berbuat curang.
Tugas ini untuk mengukur kemampuan anda, jadi kerjakan dengan sepenuh hati.
Selamat belajar, semoga sukses !***

Nama Mahasiswa: Akbar Agus Wijaya	NIM: 1301188572	Nilai:
Nama Mahasiswa: Stana Edro Swargara	NIM: 1301188539	Nilai:
Nama Mahasiswa: Dzulfikar Nur Ahmad Faisal	NIM: 1301188596	Nilai:

Siapkan tools berikut sebelum mengerjakan:

1. Go Programming Language (<https://golang.org/dl/>).
2. Visual Studio Code (<https://code.visualstudio.com/>) atau LiteIDE (<https://github.com/visualfc/liteide>).
3. Harus menggunakan linux dengan distro fedora (<https://getfedora.org/id/workstation/>).
4. Buatlah git repository pada <https://github.com/> kemudian push semua kode dan hasil laporan anda ke dalam repository github yang sudah anda buat.
5. Kumpulkan link repository github tersebut sebagai tanda bahwa anda mengerjakan tugas modul ini.
6. Link repository harus berbeda untuk setiap tugasnya. Buatlah markdown yang rapi disetiap repository tugas yang anda kumpulkan.
7. Printscreen program harus dari desktop kelompok anda sendiri, dan harus dari linux yang sudah diinstall. Jika tidak, maka harus mengulang pengerjaan tugasnya.
8. Jangan lupa untuk menuliskan NAMA dan NIM pada laporan.
9. Laporan berbentuk PDF dan dikumpulkan pada link repository github beserta kodenya.
10. Walaupun tugas berkelompok tapi pengumpulan link github harus individu, jika tidak mengumpulkan maka dianggap tidak mengerjakan.

Nama: Akbar Agus Wijaya	NIM: 1301188572	Nilai:
-------------------------	-----------------	--------

Soal No 1 (Host Lookup)

```

/* ResolveIP
*/

package main

import (
    "fmt"
    "net"
    "os"
)

func main() {
    if len(os.Args) != 2 {
        fmt.Fprintf(os.Stderr, "Usage: %s hostname\n", os.Args[0])
        fmt.Println("Usage: ", os.Args[0], "hostname")
        os.Exit(1)
    }
    name := os.Args[1]

    addr, err := net.ResolveIPAddr("ip", name)
    if err != nil {
        fmt.Println("Resolution error", err.Error())
        os.Exit(1)
    }

    fmt.Println("Resolved address is ", addr.String())
    os.Exit(0)
}

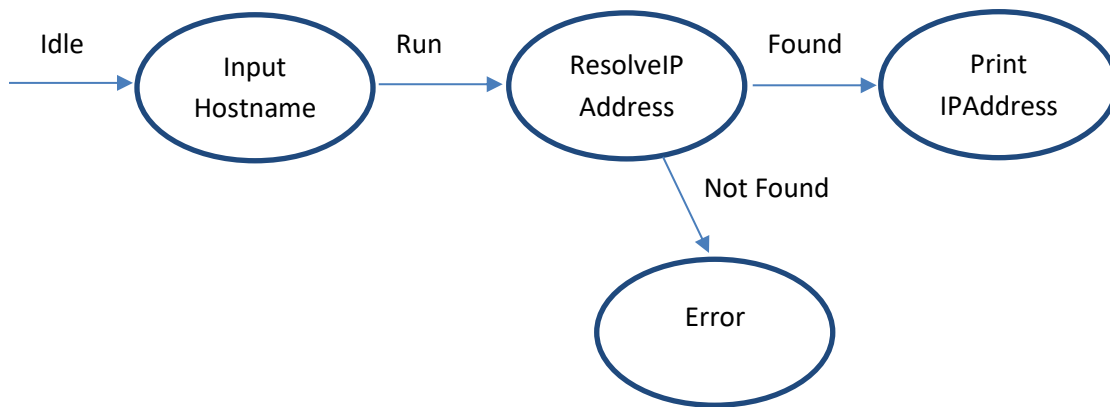
```

Jalankan program diatas (`go run ResolveIP.go www.google.com`), apakah outputnya (berikan printscreen) dan jelaskan cara kerjanya menggunakan diagram FSM!

Jawaban:

```
home > wijaya > Dokumen > Tugas2Akbar > nomor1.go
1 package main
2
3 import (
4     "fmt"
5     "net"
6     "os"
7 )
8
9 func main() {
10     if len(os.Args) != 2 {
11         fmt.Fprintf(os.Stderr, "Usage: %s hostname\n", os.Args[0])
12         fmt.Println("Usage : ", os.Args[0], "hostname")
13         os.Exit(1)
14     }
15     hostname := os.Args[1]
16     ip := net.LookupIP(hostname)
17     if len(ip) > 0 {
18         fmt.Println("Resolved address is", ip[0].String())
19     } else {
20         fmt.Println("Not found")
21     }
22 }
```

```
[wijaya@192 ~]$ go run /home/wijaya/Dokumen/Tugas2Akbar/nomor1.go www.google.com
Resolved address is 216.239.38.120
[wijaya@192 ~]$
```



Nama: Akbar Agus Wijaya	NIM: 1301188572	Nilai:
-------------------------	-----------------	--------

Soal No 2 (Service Lookup)

```

/* LookupPort
*/

package main

import (
    "fmt"
    "net"
    "os"
)

func main() {
    if len(os.Args) != 3 {
        fmt.Fprintf(os.Stderr,
            "Usage: %s network-type service\n",
            os.Args[0])
        os.Exit(1)
    }
    networkType := os.Args[1]
    service := os.Args[2]

    port, err := net.LookupPort(networkType, service)
    if err != nil {
        fmt.Println("Error: ", err.Error())
        os.Exit(2)
    }

    fmt.Println("Service port ", port)
    os.Exit(0)
}

```

Jalankan program diatas (go run LookupPort.go tcp telnet), apakah outputnya (berikan printscreen) dan jelaskan cara kerjanya menggunakan diagram FSM!

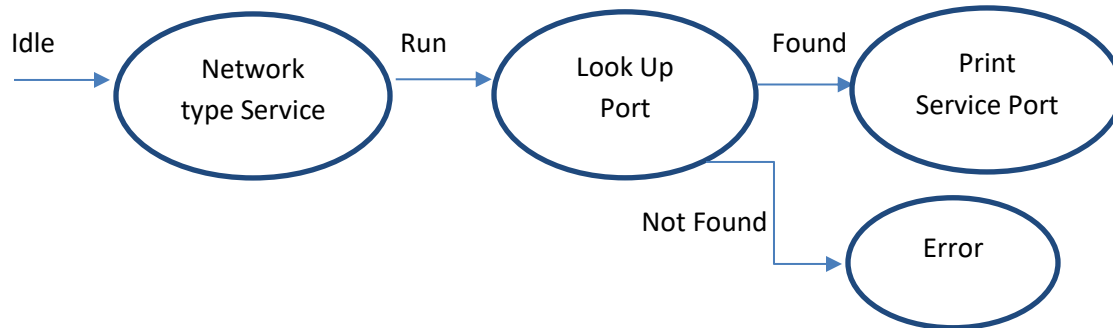
Jawaban:

Nama: Akbar Agus Wijaya

NIM: 1301188572

Nilai:

The screenshot shows the Visual Studio Code interface with a Go file named `nomor2.go`. The code defines a `main` function that checks the number of command-line arguments. If there are 3 arguments, it prints the usage. If there are 2 arguments, it prints the service port. The terminal output shows the program being run with `network-type service` and `tcp telnet` as arguments, resulting in the output `Service port 23`.



Soal No 3 (TCP Client)

```
/* GetHeadInfo
*/
package main

import (
    "fmt"
    "io/ioutil"
    "net"
    "os"
)

func main() {
    if len(os.Args) != 2 {
        fmt.Fprintf(os.Stderr, "Usage: %s host:port ", os.Args[0])
        os.Exit(1)
    }
    service := os.Args[1]

    tcpAddr, err := net.ResolveTCPAddr("tcp4", service)
    checkError(err)

    conn, err := net.DialTCP("tcp", nil, tcpAddr)
    checkError(err)

    _, err = conn.Write([]byte("HEAD / HTTP/1.0\r\n\r\n"))
    checkError(err)
}
```

```

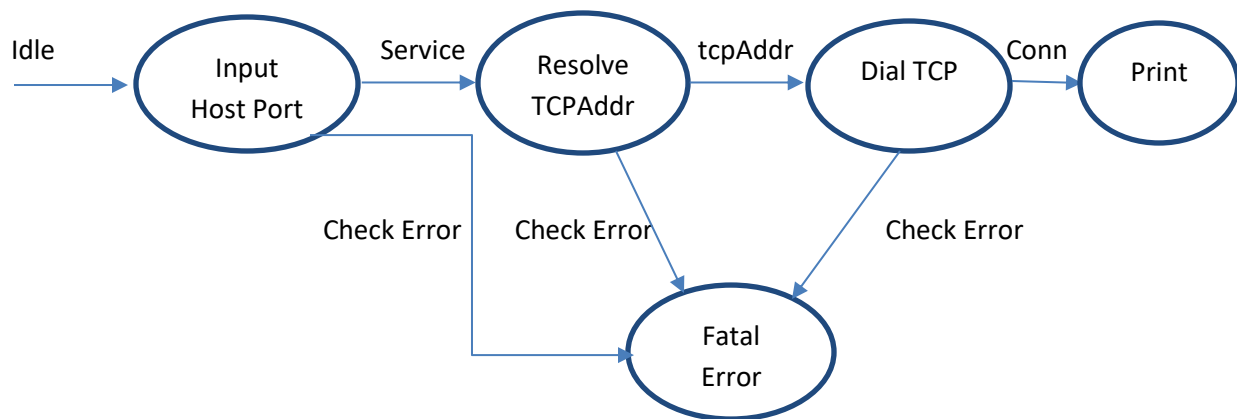
        fmt.Println(string(result))
    }
    os.Exit(0)

    func checkError(err error) {
        if err != nil {
            fmt.Fprintf(os.Stderr, "Fatal error: %s", err.Error())
            os.Exit(1)
        }
    }

```

Jalankan program diatas (`go run GetHeadInfo.go http://www.google.com:80`), apakah outputnya (berikan printscreen) dan jelaskan cara kerjanya menggunakan diagram FSM!

Jawaban:



Nama: Akbar Agus Wijaya	NIM: 1301188572	Nilai:
-------------------------	-----------------	--------

Soal No 4 (Raw Sockets and the IPConn Type)
<pre> /* Ping */ package main import ("bytes" "fmt" "io" "net" "os") // change this to my own IP address or set to 0.0.0.0 const myIPAddress = "192.168.1.2" const ipv4HeaderSize = 20 func main() { if len(os.Args) != 2 { fmt.Println("Usage: ", os.Args[0], "host") os.Exit(1) } localAddr, err := net.ResolveIPAddr("ip4", myIPAddress) </pre>

```
        if err != nil {
            fmt.Println("Resolution error", err.Error())
            os.Exit(1)
        }

        remoteAddr, err := net.ResolveIPAddr("ip4", os.Args[1])
        if err != nil {
            fmt.Println("Resolution error", err.Error())
            os.Exit(1)
        }

        conn, err := net.DialIP("ip4:icmp", localAddr, remoteAddr)
        checkError(err)

        var msg [512]byte
        msg[0] = 8 // echo
        msg[1] = 0 // code 0
        msg[2] = 0 // checksum, fix later
        msg[3] = 0 // checksum, fix later
        msg[4] = 0 // identifier[0]
        msg[5] = 13 // identifier[1] (arbitrary)
        msg[6] = 0 // sequence[0]
        msg[7] = 37 // sequence[1] (arbitrary)
        len := 8

        // now fix checksum bytes
        check := checksum(msg[0:len])
        msg[2] = byte(check >> 8)
        msg[3] = byte(check & 255)

        // send the message
        _, err = conn.Write(msg[0:len])
        checkError(err)

        fmt.Print("Message sent:  ")
        for n := 0; n < 8; n++ {
            fmt.Print(" ", msg[n])
        }
        fmt.Println()

        // receive a reply
        size, err2 := conn.Read(msg[0:])
        checkError(err2)

        fmt.Print("Message received:")
        for n := ipv4HeaderSize; n < size; n++ {
            fmt.Print(" ", msg[n])
        }
        fmt.Println()
        os.Exit(0)
    }
}
```



```
func checksum(msg []byte) uint16 {
    sum := 0

    // assume even for now
    for n := 0; n < len(msg); n += 2 {
        sum += int(msg[n])*256 + int(msg[n+1])
    }
    sum = (sum >> 16) + (sum & 0xffff)
    sum += (sum >> 16)
    var answer uint16 = uint16(^sum)
    return answer
}

func checkError(err error) {
    if err != nil {
        fmt.Fprintf(os.Stderr, "Fatal error: %s", err.Error())
        os.Exit(1)
    }
}

func readFully(conn net.Conn) ([]byte, error) {
    defer conn.Close()

    result := bytes.NewBuffer(nil)
    var buf [512]byte
    for {
        n, err := conn.Read(buf[0:])
        result.Write(buf[0:n])
        if err != nil {
            if err == io.EOF {
                break
            }
            return nil, err
        }
    }
    return result.Bytes(), nil
}
```

Jalankan program diatas, apakah outputnya (berikan printscreen) dan jelaskan cara kerjanya!

Jawaban:

```

Kam 19 Sep 23:19
nomor4.go - Visual Studio Code

File Edit Selection View Go Debug Terminal Help

DEVELOPER
No Configurations
nomor4.go x
home > wijaya > Dokumen > Tugas2Akbar > nomor4.go
7  "net"
8  "os"
9  }
10
11 const myIPAddress = "192.168.1.16"
12 const ipv4HeaderSize = 20
13
14 func main() {
15     if len(os.Args) != 2 {
16         fmt.Println("Usage : ", os.Args[0], "host")
17         os.Exit(1)
18     }
19 }

VARIABLES
WATCH
CALL STACK
BREAKPOINTS

TERMINAL
1: bash
Fatal error : dial ip4:icmp 216.239.38.120->216.239.38.120: socket: operation not permitted exit status 1
[wijaya@192 ~]$ sudo go run /home/wijaya/Dokumen/Tugas2Akbar/nomor4.go www.google.com
[sudo] password for wijaya:
Fatal error : dial ip4:icmp 216.239.38.120->216.239.38.120: bind: cannot assign requested address exit status 1
[wijaya@192 ~]$ go run /home/wijaya/Dokumen/Tugas2Akbar/nomor4.go www.google.com
Fatal error : dial ip4:icmp 192.168.1.16->216.239.38.120: socket: operation not permitted exit status 1
[wijaya@192 ~]$ go run /home/wijaya/Dokumen/Tugas2Akbar/nomor4.go www.google.com
Fatal error : dial ip4:icmp 36.79.165.105->216.239.38.120: socket: operation not permitted exit status 1
[wijaya@192 ~]$ sudo go run /home/wijaya/Dokumen/Tugas2Akbar/nomor4.go www.google.com
Fatal error : dial ip4:icmp 36.79.165.105->216.239.38.120: bind: cannot assign requested address exit status 1
[wijaya@192 ~]$ sudo go run /home/wijaya/Dokumen/Tugas2Akbar/nomor4.go www.google.com
# command-line-arguments
Dokumen/Tugas2Akbar/nomor4.go:62:11: undefined: ipv4HeaderSize
[wijaya@192 ~]$ sudo go run /home/wijaya/Dokumen/Tugas2Akbar/nomor4.go www.google.com
Message sent :  8 0 247 205 0 13 0 37
Message received :  0 0 255 205 0 13 0 37
[wijaya@192 ~]$

```

Cara kerjanya adalah :

Pesan dikirim melalui ping ke host yang merupakan protocol berorientasi byte. Formatnya :

- byte pertama adalah 8, untuk pesan echo
- byte kedua adalah 0
- byte ketiga dan keempat merupakan checksum untuk seluruh pesan
- byte kelima dan keenam merupakan identifikasi arbitrer
- byte ketujuh dan kedelapan merupakan nomor urut sembarang
- sisa paket merupakan data pengguna

Soal No 5 (Multi-Threaded Server)

```
package main

import (
    "bufio"
    "fmt"
    "net"
)

func check(err error, message string) {
    if err != nil {
        panic(err)
    }
    fmt.Printf("%s\n", message)
}

func main() {
    ln, err := net.Listen("tcp", ":8080")
    check(err, "Server is ready.")

    for {
        conn, err := ln.Accept()
        check(err, "Accepted connection.")

        go func() {
            buf := bufio.NewReader(conn)

            for {
                name, err := buf.ReadString('\n')

                if err != nil {
                    fmt.Printf("Client disconnected.\n")
                    break
                }

                conn.Write([]byte("Hello, " + name))
            }
        }()
    }
}
```

Salin program diatas di dalam virtual box yang sudah anda buat, kemudian lakukan telnet ke port 8080 dalam jumlah yang banyak secara bersamaan, apakah outputnya (berikan printscreen) dan jelaskan cara kerjanya!

Jawaban:
Server

The screenshot shows the Visual Studio Code interface with a Go file named `nomor5.go` open. The code defines a simple HTTP server using `net/http` and `bufio`. The terminal output shows the server running on `localhost:8080` and accepting a connection from `wijaya@192`. The terminal also displays system information for the `wijaya@localhost` session.

```
1 package main
2
3 import (
4     "bufio"
5     "fmt"
6     "net"
7 )
8
9 func check(err error, message string) {
10     if err != nil {
11         panic(err)
12     }
13     fmt.Printf("%s\n", message)
14 }
15
16 func main() {
17     listener, err := net.Listen("tcp", ":8080")
18     if err != nil {
19         check(err, "Failed to listen on port 8080")
20     }
21     for {
22         conn, err := listener.Accept()
23         if err != nil {
24             check(err, "Failed to accept connection")
25             continue
26         }
27         go handleConnection(conn)
28     }
29 }
30
31 func handleConnection(conn net.Conn) {
32     reader := bufio.NewReader(conn)
33     request, err := reader.ReadString('\n')
34     if err != nil {
35         check(err, "Failed to read request")
36         return
37     }
38     fmt.Println(request)
39     response := "HTTP/1.1 200 OK\nContent-Type: text/plain\n\nHello, World!"
40     conn.Write([]byte(response))
41     conn.Close()
42 }
```

Terminal Output:

```
[wijaya@192 ~]$ go run /home/wijaya/nomor5.go
Server is ready.
Accepted connection.
[wijaya@localhost ~]$
[wijaya@localhost ~]$
```

Client

The screenshot shows the Visual Studio Code interface with a Go file named `nomor6.go` open. The code defines a simple HTTP client using `net/http` and `bufio`. The terminal output shows the client running on `localhost:8080` and connecting to the server. The terminal also displays system information for the `wijaya@localhost` session.

```
1 package main
2
3 import (
4     "bufio"
5     "fmt"
6     "net"
7     "time"
8 )
9
10 func check(err error, message string) {
11     if err != nil {
12         panic(err)
13     }
14 }
15
16 func main() {
17     client := &net.Dialer{
18         Timeout: 5 * time.Second,
19     }
20     conn, err := client.Dial("tcp", "localhost:8080")
21     if err != nil {
22         check(err, "Failed to connect to localhost:8080")
23         return
24     }
25     writer := bufio.NewWriter(conn)
26     request := "GET / HTTP/1.1"
27     writer.WriteString(request)
28     writer.Flush()
29     response, err := bufio.NewReader(conn).ReadString('\n')
30     if err != nil {
31         check(err, "Failed to read response")
32         return
33     }
34     fmt.Println(response)
35 }
```

Terminal Output:

```
[wijaya@192 ~]$ go run /home/wijaya/nomor6.go
Server is ready.
Accepted connection.
[wijaya@localhost ~]$
[wijaya@localhost ~]$
```

Nama: Akbar Agus Wijaya	NIM: 1301188572	Nilai:
-------------------------	-----------------	--------

Program dijalankan dengan melakukan koneksi ke telnet port 8080, panggil telnet 8080 lalu program akan melakukan koneksi ke IP yang ditemukan, program tidak bisa melakukan koneksi ke host dan koneksi ke telnet 8080, dicoba kembali dengan menambahkan alamat IP dan diikuti dengan port 8080, program akan melakukan perulangan sampai terkoneksi ke localhost dengan port 8080

Soal No 6 (Multi-Threaded Server)

```
package main

import (
    "bufio"
    "fmt"
    "net"
    "time"
)

func check(err error, message string) {
    if err != nil {
        panic(err)
    }
    fmt.Printf("%s\n", message)
}

type ClientJob struct {
    name string
    conn net.Conn
}

func generateResponses(clientJobs chan ClientJob) {
    for {
        // Wait for the next job to come off the queue.
        clientJob := <-clientJobs

        // Do something thats keeps the CPU busy for a whole second.
        for start := time.Now(); time.Now().Sub(start) < time.Second; {
        }

        // Send back the response.
        clientJob.conn.Write([]byte("Hello, " + clientJob.name))
    }
}

func main() {
    clientJobs := make(chan ClientJob)
    go generateResponses(clientJobs)

    ln, err := net.Listen("tcp", ":8080")
    check(err, "Server is ready.")

    for {
        conn, err := ln.Accept()
        check(err, "Accepted connection.")

        go func() {
            buf := bufio.NewReader(conn)

            for {
                name, err := buf.ReadString('\n')

                if err != nil {
                    fmt.Printf("Client disconnected.\n")
                    break
                }

                clientJobs <- ClientJob{name, conn}
            }
        }()
    }
}
```

Nama: Akbar Agus Wijaya	NIM: 1301188572	Nilai:
-------------------------	-----------------	--------

alankan program diatas di dalam virtual box yang sudah anda buat, kemudian lakukan telnet ke port 8080 dalam jumlah yang banyak secara bersamaan, apakah outputnya (berikan printscreen) dan jelaskan cara kerjanya!

Jawaban:

```
1 package main
2
3 import (
4     "bufio"
5     "fmt"
6     "net"
7     "time"
8 )
9
10 func check(err error, message string) {
11     if err != nil {
12         panic(err)
13     }
14 }
15
16 func main() {
17     listener, err := net.Listen("tcp", ":8080")
18     check(err, "Server is ready.")
19     for {
20         conn, err := listener.Accept()
21         check(err, "Accepted connection.")
22         go handleConn(conn)
23     }
24 }
25
26 func handleConn(conn net.Conn) {
27     reader := bufio.NewReader(conn)
28     for {
29         message, err := reader.ReadString('\n')
30         check(err, "client disconnected.")
31         fmt.Println("Hello, " + message)
32         time.Sleep(1 * time.Second)
33     }
34 }
```

```
[wijaya@192 ~]$ go run /home/wijaya/Dokumen/Tugas2Akbar/nomor6.go
Server is ready.
Accepted connection.
client disconnected.
[wijaya@192 ~]$
```

```
wijaya@localhost:~$ telnet localhost 8080
Trying ::1...
Connected to localhost.
Escape character is '^]'.
Hello, cek
Connection closed by foreign host.
[wijaya@192 ~]$ telnet localhost 8080
Trying ::1...
Connected to localhost.
Escape character is '^]'.
Hello
Hello, hello
fikir
Hello, fikir
akbar
Hello, akbar
Hello, *)
telnet> close
Hello, telnet> close
*)
Hello, ^]
^]
Hello, '^]'
^]
telnet> close
Connection closed.
[wijaya@192 ~]$
```

Server akan menunggu client untuk mereply dengan accepted connection.