Telkom University	**Tugas 3** **Pemrograman Jaringan (CSH4V3)** **----------** **Semester Ganjil 2019 - 2020** **Dosen: Aulia Arif Wardana, S.Kom., M.T. (UIW)**

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

Nama Mahasiswa: I Putu Surya Baratha ……	**NIM:** 1301188566 ……	**Nilai:** ……
Nama Mahasiswa: Muhammad Risdham Nur A.P ……	**NIM:** 1301188603 ……	**Nilai:** ……
Nama Mahasiswa: Sella Tresnasari ……	**NIM:** 1301188565 ……	**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:	NIM:	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:

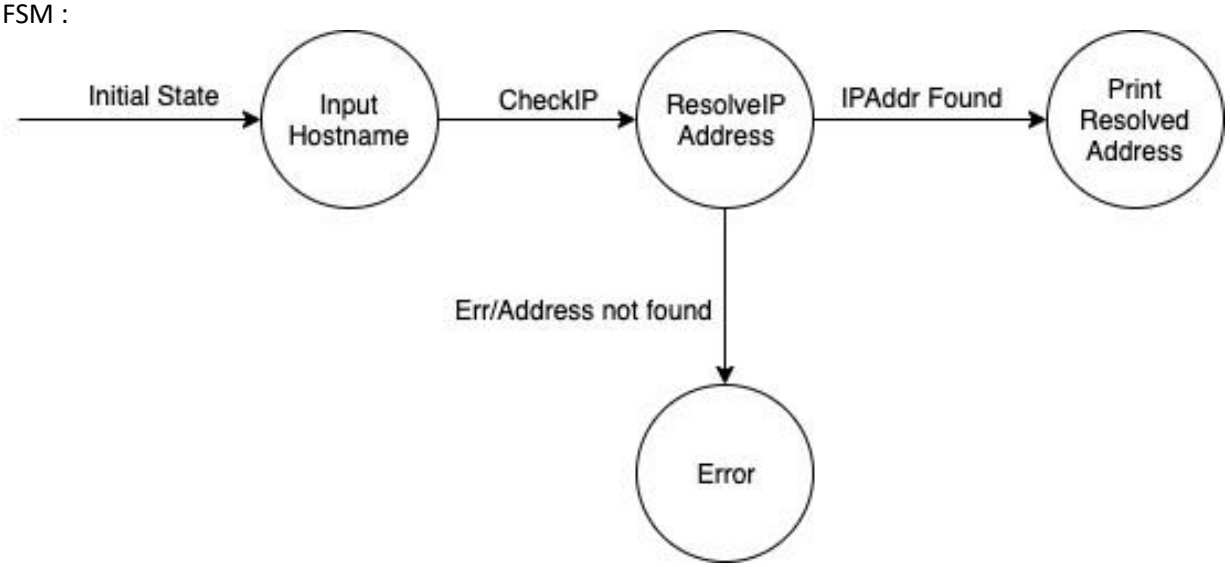
```
localhost:go muhrisdham$ go run /Users/muhrisdham/go/src/ResolveIP.go www.google.com
Reolved address is  216.239.38.120
localhost:go muhrisdham$
```

Pada program nomor 1 mengenai Host Lookup yaitu mencari host yang diberikan menggunakan resolver lokal. Host Lookup dapat mengembalikan sepotong alamat host tersebut. Host lookup juga merupakan program yang memberikan data berupa informasi suatu domain tertentu seperti alamat IP, server, dll.

Cara kerjanya seperti program menjalankan website lalu Resolve memberikan informasi IP nya.

Nama:	NIM:	Nilai:

FSM :



Nama:	NIM:	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)
}
```

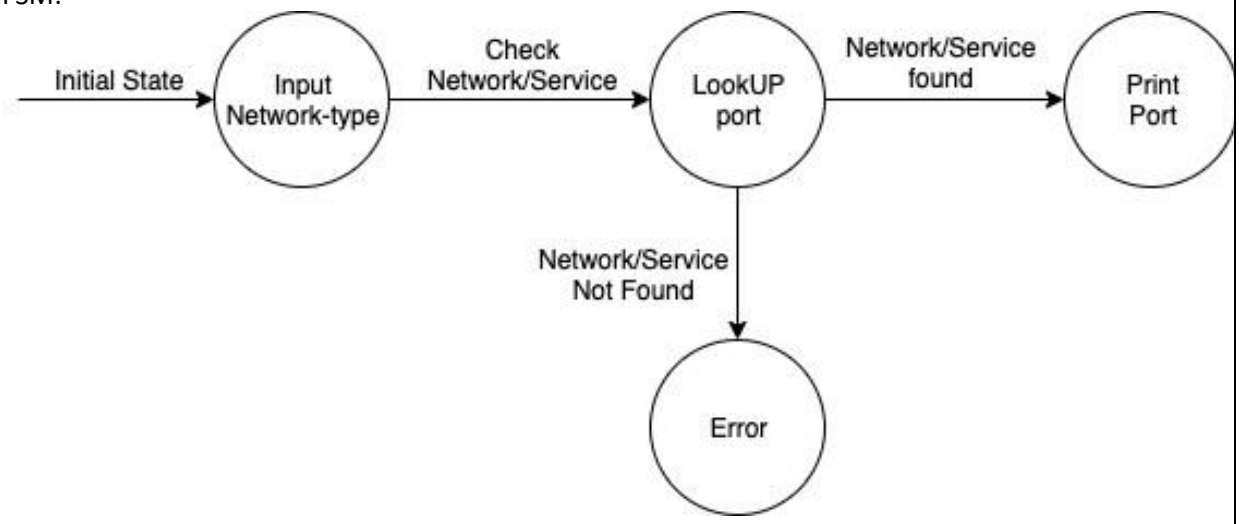
Jalankan program diatas (go run LookupPort.go tcp telnet), apakah outputnya (berikan printscreen) dan jelaskan cara kerjanya menggunakan diagram FSM!

Jawaban:

```
localhost:go muhrisdham$ go run /Users/muhrisdham/go/src/LookupPort.go tcp telnet
Service port  23
localhost:go muhrisdham$
```

Nama:	NIM:	Nilai:

FSM:



Nama:	NIM:	Nilai:

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)

        result, err := ioutil.ReadAll(conn)
        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!

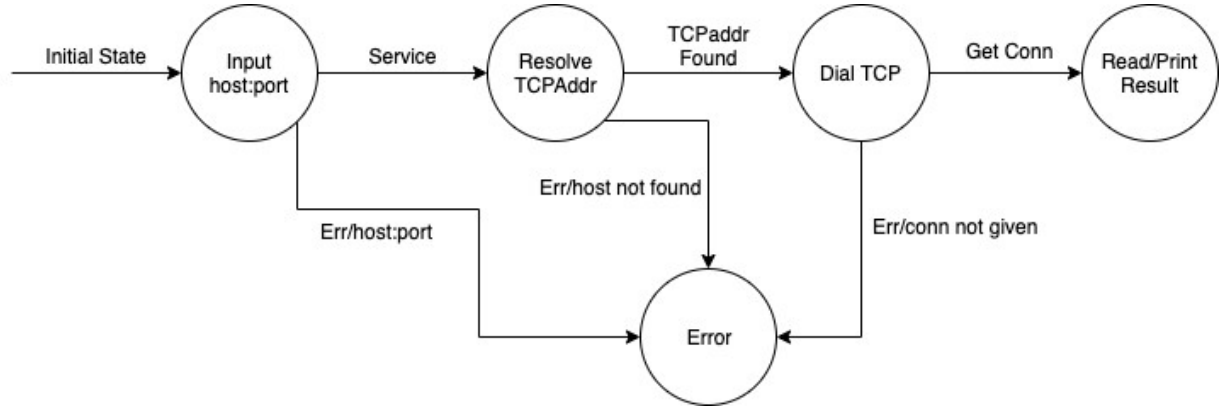
Jawaban:

```
localhost:go muhrisdham$ go run /Users/muhrisdham/go/src/GetHeadInfo.go google.com:80
HTTP/1.0 200 OK
Date: Tue, 10 Sep 2019 09:29:22 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
P3P: CP="This is not a P3P policy! See g.co/p3phelp for more info."
Server: gws
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN
Set-Cookie: 1P_JAR=2019-09-10-09; expires=Thu, 10-Oct-2019 09:29:22 GMT; path=/; domain=.g
oogle.com; SameSite=none
Set-Cookie: NID=188=qSwuwqP6QSvroHI4Ql14pvmFp7-NB7KvgPi7JCWOs20HlQvpIYc-Wttv5ywmPmo2EhkA8t
JQCaPaBWqxNGBK5Ty6tmNcQ1G0B8Y1xfxWjpXs6aUZwi87CLDaposevUxrOlfsphpxv8v0gcmOtTqd9yNAnm6XDTMC
upWPwBLEtxg; expires=Wed, 11-Mar-2020 09:29:22 GMT; path=/; domain=.google.com; HttpOnly
Accept-Ranges: none
Vary: Accept-Encoding
```

Nama:	NIM:	Nilai:

Transmission Control Protocol (TCP) adalah salah satu protokol dari dua protokol inti dari Internet Protokol (IP), dan sangat umum sehingga paket tersebut sering disebut TCP/IP yang menyediakan kehandalan, memerintahkan, pemeriksaan kesalahan pengiriman aliran paket antara program yang berjalan pada komputer yang terhubungke intranet maupun internet.
Cara kerja TCP lebih mementingkan tata-cara dan keandalan dalam pengiriman data antara dua komputer dalam jaringan.

FSM:



Nama:	NIM:	Nilai:

Soal No 4 (Raw Sockets and the IPConn Type)

```
/* 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)

        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)
```

Nama:	NIM:	Nilai:

```
        // 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
}
```

Nama:	NIM:	Nilai:

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

Jawaban:

```
localhost:go muhrisdham$ sudo go run /Users/muhrisdham/go/src/Ping.go www.google.com
Message sent:     8 0 247 205 0 13 0 37
Message received: 0 0 255 205 0 13 0 37
localhost:go muhrisdham$
```

Cara kerja

Ping dengan menggunakan perintah echo dari protocol ICMP. Ini merupakan protocol dengan berorientasi byte yang dimana klien mengirim aliran byte ke host lain lalu host akan membalas. Formatnya adalah:

- Byte yang pertama adalah 8, untuk pesan echo
- Byte yang kedua adalah nol
- Byte yang ketiga dan keempat merupakan checksum untuk seluruh pesan
- Byte yang kelima dan keenam merupakan pengidentifikasi arbitrer
- Byte yang ketujuh dan kedelapan merupakan nomor urut yang sembarang
- Sisa paket merupakan data pengguna

Program tersebut akan menyiapkan koneksi IP dan mengirim permintaan untuk ping ke host serta mendapatkan balasan.

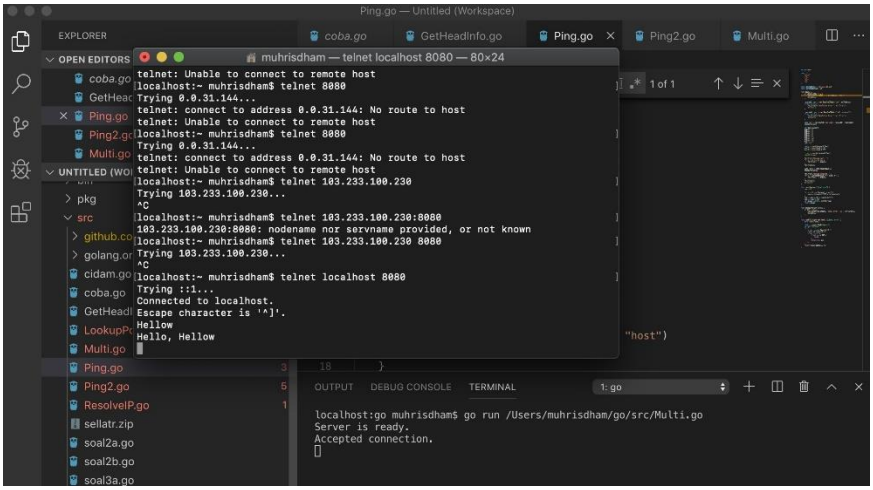
Nama:	NIM:	Nilai:

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))
                        }
                }()
        }
}
```

Jalankan 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:



Nama:	NIM:	Nilai:

Multi-threading merupakan salah satu teknik dalam pemrograman yang bertujuan untuk membuat proses berjalan secara bersamaan dalam satu waktu. Pada sebuah thread bagian program yang dapat berjalan mandiri, sehingga dua atau lebih thread dapat
berjalan bersamaan, tanpa yang satu harus menunggu selesainya yang lain.

1. Program dijalankan dengan melakukan koneksi ke telnet port 8080 dengan jumlah yang banyak dengan waktu yang bersamaan.
2. Pertama panggil “telnet 8080” lalu program akan melakukan koneksi ke IP yang ditemukan.
3. Program mengeluarkan statement bahwa tidak bias koneksi ke Remote Host.
4. Yang kedua juga sama melakukan konksi ke “telnet 8080)
5. Lalu dicoba kembali dengan koseksi telnet dengan menambahkan alamat IP lalu diikuti port 8080
6. Program akan menjalankan looping sampai terkoneksi ke localhost dengan port 8080 dan melakukan input berupa string

Nama:	NIM:	Nilai:

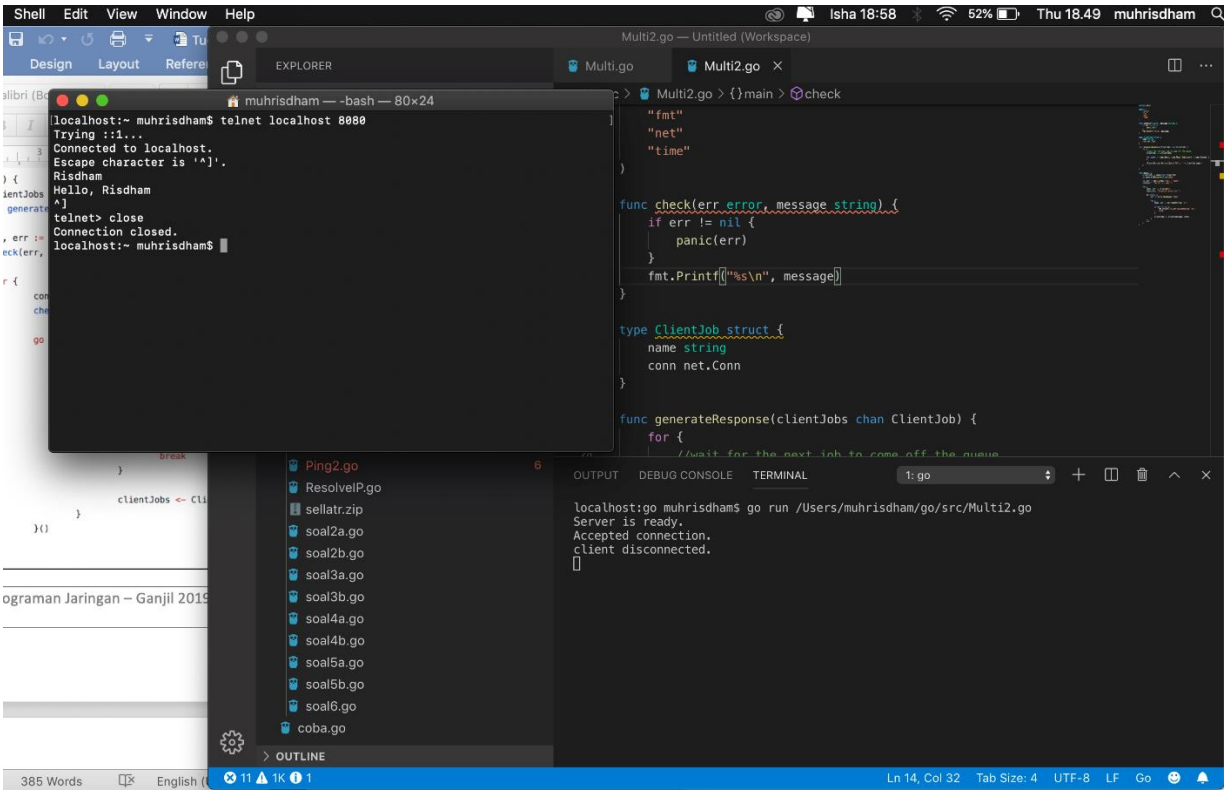
Soal No 6 (Multi-Threaded Server)

```
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 buys 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:	NIM:	Nilai:

Jalankan 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:



Multi-threading merupakan salah satu teknik dalam pemrograman yang bertujuan untuk membuat proses berjalan secara bersamaan dalam satu waktu. Pada sebuah thread bagian program yang dapat berjalan mandiri, sehingga dua atau lebih thread dapat berjalan bersamaan, tanpa yang satu harus menunggu selesainya yang lain. Namun pada kasus kali ini program akan melakukan proses menurut antrian.

Cara kerja:

1. Program terkoneksi ke localhost dengan melakukan telnet ke port 8080
2. Program memproses sesuai antrian
3. CPU melakukan proses selama 1 detik
4. Input data berupa stirng
5. Menampilkan kembali output
6. Melakukan telnet> cost agar keluar dari program