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:	NIM:	Nilai:
Meilyand Evriyan Timor	**1301161769**	**……**
Nama Mahasiswa:	**NIM:**	**Nilai:**
Reyhan Rahmansyah	**1301160805**	**……**
Nama Mahasiswa:	**NIM:**	**Nilai:**
Reno Butar Butar	**1301164724**	**……**

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.

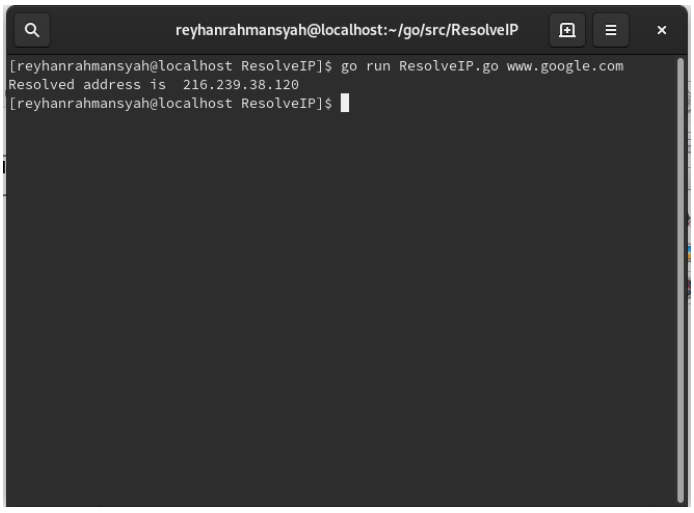
Nama:	NIM:	Nilai:

Soal No 1 (Host Lookup)

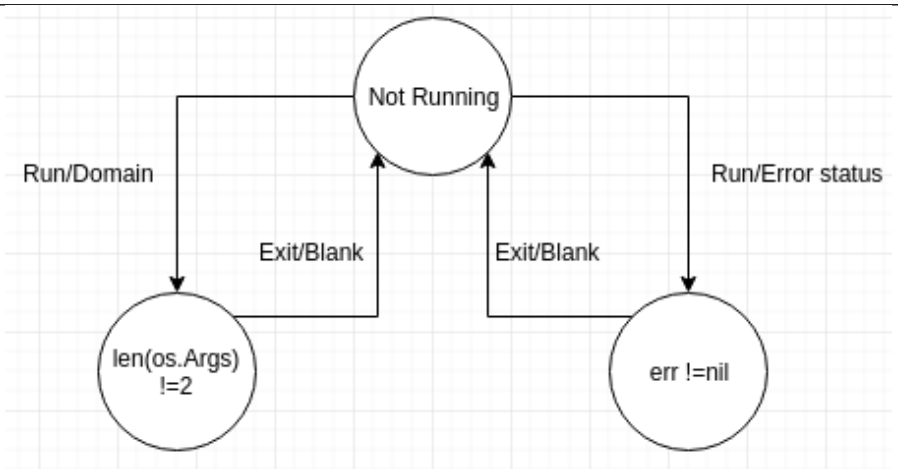
```
/* ResolveIP
 */

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:



Nama:	NIM:	Nilai:



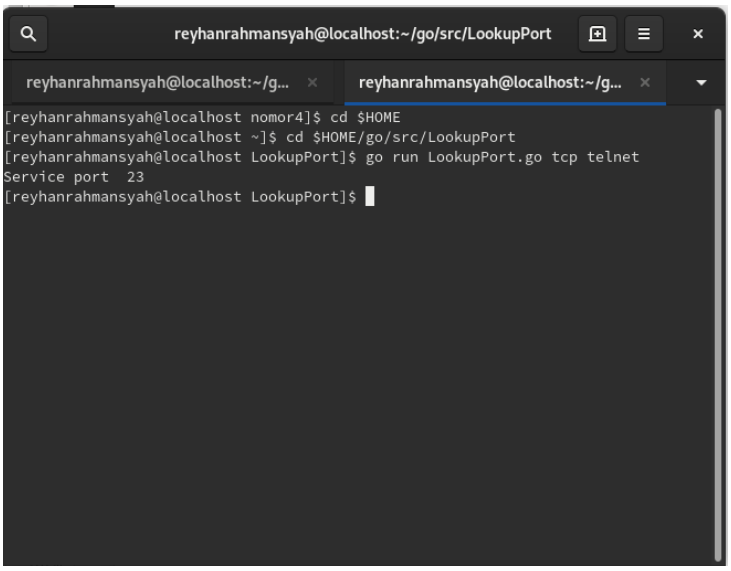
Nama:	NIM:	Nilai:

Soal No 2 (Service Lookup)

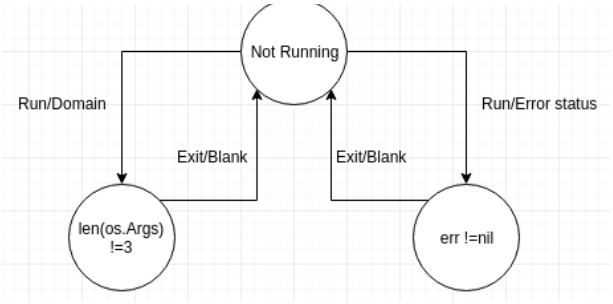
```
/* LookupPort
 */

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:



Nama:	NIM:	Nilai:



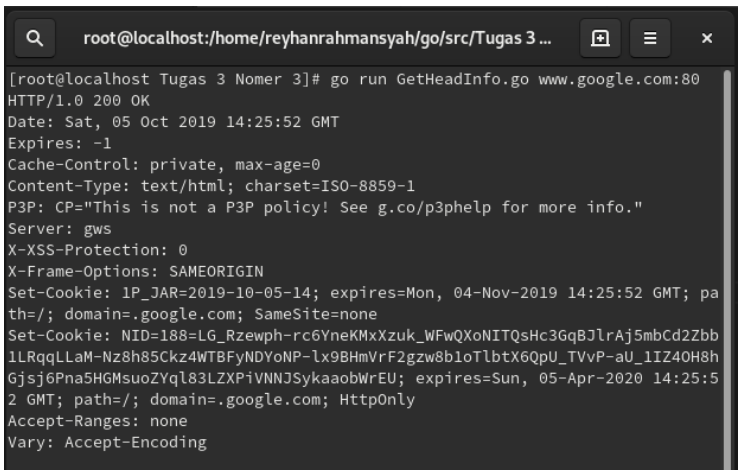
Soal No 3 (TCP Client)

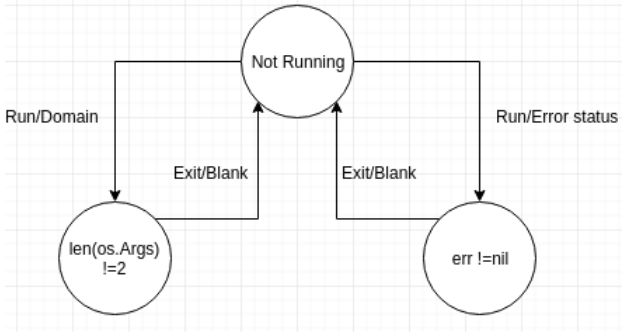
```
/* GetHeadInfo
 */
package main

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!

Nama:	NIM:	Nilai:

Jawaban:





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)
```

Nama:	NIM:	Nilai:

```
        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])           n+1])
        }
        fmt.Println()
        os.Exit(0)
}
        return answer
}

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

Nama:	NIM:	Nilai:

```
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!

Nama:	NIM:	Nilai:

Jawaban:

```
[reyhanrahmansyah@localhost src]$ sudo go run Tugas3Nomer4.go www.google.com
Message sent :  8 0 247 205 0 13 0 37
Message received :  0 0 255 205 0 13 0 37
```

Soal No 5 (Multi-Threaded Server)

Nama:	NIM:	Nilai:

```go
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!

Nama:	NIM:	Nilai:

Jawaban:

```
[root@localhost src]# go run Tugas3Nomer5.go
# runtime
/usr/local/go/src/runtime/cpuflags_amd64.go:14:2: offsetX86HasAVX2 redeclared in this block
        previous declaration at /usr/local/go/src/runtime/cpuflags.go:14:37
/usr/local/go/src/runtime/mem_linux.go:20:6: sysAlloc redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:14:43
/usr/local/go/src/runtime/mem_linux.go:37:6: sysUnused redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:32:36
/usr/local/go/src/runtime/mem_linux.go:108:6: sysUsed redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:36:34
/usr/local/go/src/runtime/mem_linux.go:134:6: sysFree redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:42:52
/usr/local/go/src/runtime/mem_linux.go:139:6: sysFault redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:48:35
/usr/local/go/src/runtime/mem_linux.go:143:6: sysReserve redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:52:46
/usr/local/go/src/runtime/mem_linux.go:151:6: sysMap redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:60:51
/usr/local/go/src/runtime/netpoll_epoll.go:25:6: netpollinit redeclared in this block
        previous declaration at /usr/local/go/src/runtime/netpoll_aix.go:55:6
/usr/local/go/src/runtime/netpoll_epoll.go:39:6: netpolldescriptor redeclared in this block
        previous declaration at /usr/local/go/src/runtime/netpoll_aix.go:90:26
/usr/local/go/src/runtime/netpoll_epoll.go:39:6: too many errors
# math
/usr/local/go/src/math/trig_reduce.go:42:13: undefined: bits.Mul64
/usr/local/go/src/math/trig_reduce.go:43:16: undefined: bits.Mul64
/usr/local/go/src/math/trig_reduce.go:45:11: undefined: bits.Add64
/usr/local/go/src/math/trig_reduce.go:46:11: undefined: bits.Add64
[root@localhost src]# _
```

```
[reyhanrahmansyah@localhost ~]$ telnet 192.168.1.3 8080
Trying 192.168.1.3...
telnet: connect to address 192.168.1.3: No route to host
```

Nama:	NIM:	Nilai:

Soal No 6 (Multi-Threaded Server)

```go
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:

```
[root@localhost src]# go run Tugas3Nomer6.go
# runtime
/usr/local/go/src/runtime/cpuflags_amd64.go:14:2: offsetX86HasAVX2 redeclared in this block
        previous declaration at /usr/local/go/src/runtime/cpuflags.go:14:37
/usr/local/go/src/runtime/mem_linux.go:20:6: sysAlloc redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:14:43
/usr/local/go/src/runtime/mem_linux.go:37:6: sysUnused redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:32:36
/usr/local/go/src/runtime/mem_linux.go:108:6: sysUsed redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:36:34
/usr/local/go/src/runtime/mem_linux.go:134:6: sysFree redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:42:52
/usr/local/go/src/runtime/mem_linux.go:139:6: sysFault redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:48:35
/usr/local/go/src/runtime/mem_linux.go:143:6: sysReserve redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:52:46
/usr/local/go/src/runtime/mem_linux.go:151:6: sysMap redeclared in this block
        previous declaration at /usr/local/go/src/runtime/mem_aix.go:60:51
/usr/local/go/src/runtime/netpoll_epoll.go:25:6: netpollinit redeclared in this block
        previous declaration at /usr/local/go/src/runtime/netpoll_aix.go:55:6
/usr/local/go/src/runtime/netpoll_epoll.go:39:6: netpolldescriptor redeclared in this block
        previous declaration at /usr/local/go/src/runtime/netpoll_aix.go:90:26
/usr/local/go/src/runtime/netpoll_epoll.go:39:6: too many errors
# math
/usr/local/go/src/math/trig_reduce.go:42:13: undefined: bits.Mul64
/usr/local/go/src/math/trig_reduce.go:43:16: undefined: bits.Mul64
/usr/local/go/src/math/trig_reduce.go:45:11: undefined: bits.Add64
/usr/local/go/src/math/trig_reduce.go:46:11: undefined: bits.Add64
[root@localhost src]#
```

Nama:	NIM:	Nilai: