

1. Maksud dari diagram finite state machine dari TCP Connection pada soal no 1 adalah Terhadap server

- Server di mulai pada starting point
- Server mengetahui request dari client
- Server menerima SYN dari server
- Server mengirim SYN ACK dari client
- Server menerima ACK dan melanjutkan ke proses connection established
- Server menerima FIN dan mengirim ACK
- Server mengirim FIN ke ACK terakhir
- Server menerima ACK

Terhadap client

- Client di mulai dari starting point
- Client mengirim ACK
- SYN ACK dari client di terima dan dilanjutkan mengirim ACK untuk masuk ke proses connection established
- Dilakukan data transfer sehingga client mengirim FIN dan menunggu hingga diterima
- Jika FIN pertama tidak diterima, maka client harus menunggu sampai proses FIN yang kedua untuk diterima
- Jika FIN kedua diterima, maka client mengirim ACK
- Jika terjadi timeout maka proses telah selesai.

2. Program 1

```
package main
import "fmt"
func main() {
    i := 1
    for i <= 3 {
        fmt.Println(i)
        i = i + 1
    }

    for j := 7; j <= 9; j++ {
        fmt.Println(j)
    }

    for {
        fmt.Println("loop")
        break
    }

    for n := 0; n <= 5; n++ {
        if n%2 == 0 {
            continue
        }
        fmt.Println(n)
    }
}
```

Maksud dari program diatas adalah program tersebut melakukan perulangan dengan nilai yang sudah ditentukan, berhenti ketika nilai false, dan dapat dihentikan dengan break dan continue.

Program 2

```
package main

import "fmt"

func main() {

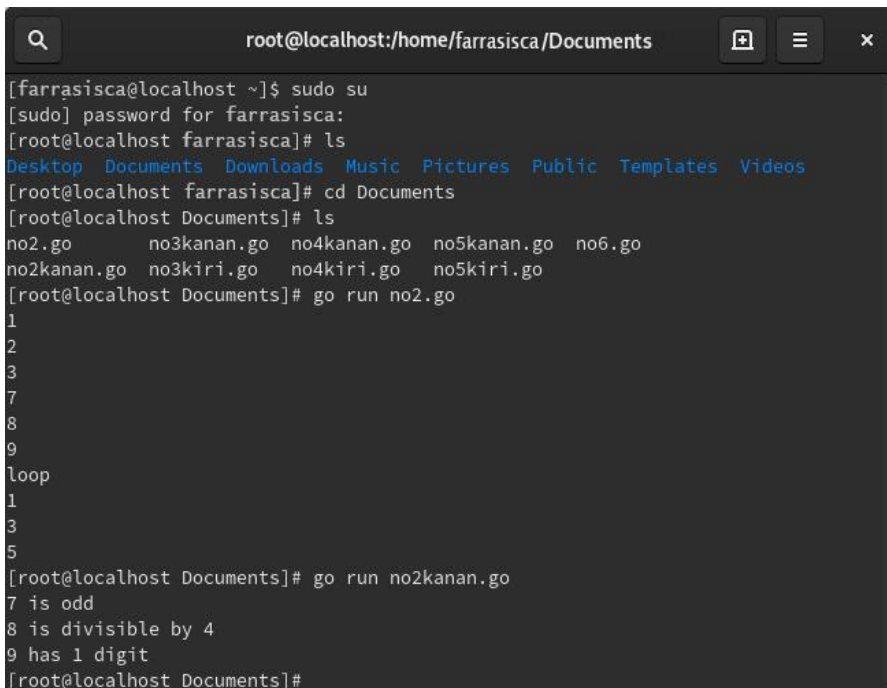
    if 7%2 == 0 {
        fmt.Println("7 is even")
    } else {
        fmt.Println("7 is odd")
    }

    if 8%4 == 0 {
        fmt.Println("8 is divisible by 4")
    }

    if num := 9; num < 0 {
        fmt.Println(num, "is negative")
    } else if num < 10 {
        fmt.Println(num, "has 1 digit")
    } else {
        fmt.Println(num, "has multiple digits")
    }
}
```

Maksud dari program diatas adalah program ke dua, kondisinya yang pertama jika 7 di mod 2 hasilnya 0 maka menghasilkan “7 is even” dan jika tidak “7 is odd”. Pada kondisi berikutnya jika 8 di mod 4 hasilnya 0 maka menghasilkan “8 is divisible by 4”. Pada kondisi berikutnya jika variabel kurang dari 0 maka menghasilkan “is negative”, jika kurang dari 10 menghasilkan “has 1 digit”, jika tidak keduanya maka akan menghasilkan “ has multiple digits”

Hasil



```
root@localhost:/home/farrasisca/Documents

[farrasisca@localhost ~]$ sudo su
[sudo] password for farrasisca:
[root@localhost farrasisca]# ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
[root@localhost farrasisca]# cd Documents
[root@localhost Documents]# ls
no2.go      no3kanan.go  no4kanan.go  no5kanan.go  no6.go
no2kanan.go no3kiri.go   no4kiri.go   no5kiri.go
[root@localhost Documents]# go run no2.go
1
2
3
7
8
9
loop
1
3
5
[root@localhost Documents]# go run no2kanan.go
7 is odd
8 is divisible by 4
9 has 1 digit
[root@localhost Documents]#
```

3. Program 1

```
package main

import "fmt"

func main() {

    var a [5]int
    fmt.Println("emp:", a)

    a[4] = 100
    fmt.Println("set:", a)
    fmt.Println("get:", a[4])

    fmt.Println("len:", len(a))

    b := [5]int{1, 2, 3, 4, 5}
    fmt.Println("dcl:", b)

    var twoD [2][3]int
    for i := 0; i < 2; i++ {
        for j := 0; j < 3; j++ {
            twoD[i][j] = i + j
        }
    }
    fmt.Println("2d: ", twoD)
}
```

- Array adalah sekumpulan variabel yang memiliki tipe data yang sama dan dinyatakan dengan nama yang sama.
- Array dua dimensi atau array multidimensi

Program 2

```
package main

import "fmt"

func plus(a int, b int) int {

    return a + b
}

func plusPlus(a, b, c int) int {
    return a + b + c
}

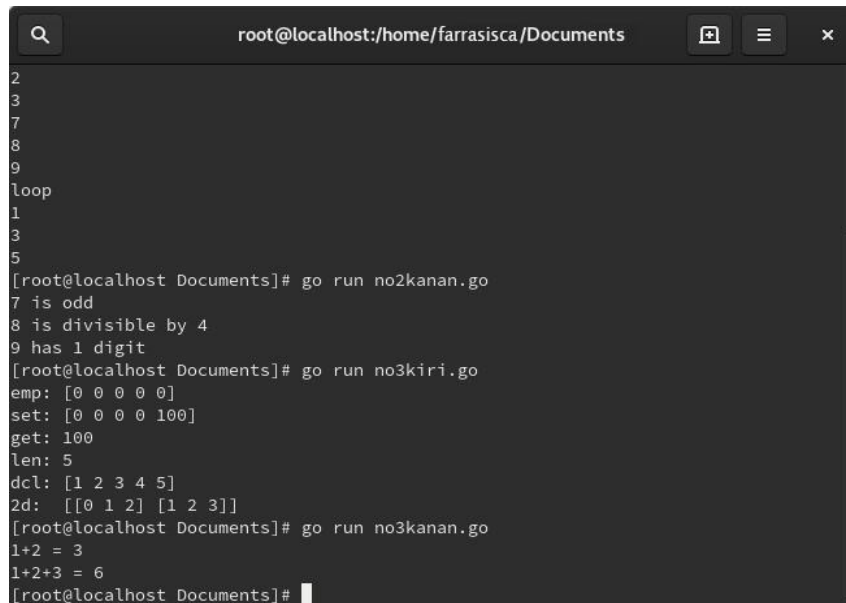
func main() {

    res := plus(1, 2)
    fmt.Println("1+2 =", res)

    res = plusPlus(1, 2, 3)
    fmt.Println("1+2+3 =", res)
}
```

- Function adalah sekelompok pernyataan yang akan dipanggil ketika diinginkan kecuali function utama akan dipanggil secara otomatis.

Hasil



```
2
3
7
8
9
loop
1
3
5
[root@localhost Documents]# go run no2kanan.go
7 is odd
8 is divisible by 4
9 has 1 digit
[root@localhost Documents]# go run no3kiri.go
emp: [0 0 0 0 0]
set: [0 0 0 0 100]
get: 100
len: 5
dcl: [1 2 3 4 5]
2d: [[0 1 2] [1 2 3]]
[root@localhost Documents]# go run no3kanan.go
1+2 = 3
1+2+3 = 6
[root@localhost Documents]#
```

4. Program 1

```
package main

import "fmt"

type person struct {
    name string
    age  int
}

func main() {

    fmt.Println(person{"Bob", 20})

    fmt.Println(person{name: "Alice", age: 30})

    fmt.Println(person{name: "Fred"})

    fmt.Println(&person{name: "Ann", age: 40})

    s := person{name: "Sean", age: 50}
    fmt.Println(s.name)

    sp := &s
    fmt.Println(sp.age)

    sp.age = 51
    fmt.Println(sp.age)
}
```

- Maksud dari program diatas adalah tipe data di assign name dan age untuk memberikan nilai, dan nilai atribut di panggil satu persatu sesuai dengan urutan tipe data.

Program 2

```

package main

import "fmt"

type rect struct {
    width, height int
}

func (r *rect) area() int {
    return r.width * r.height
}

func (r rect) perim() int {
    return 2*r.width + 2*r.height
}

func main() {
    r := rect{width: 10, height: 5}

    fmt.Println("area: ", r.area())
    fmt.Println("perim:", r.perim())

    rp := &r
    fmt.Println("area: ", rp.area())
    fmt.Println("perim:", rp.perim())
}

```

- Maksud dari program diatas adalah metode yang digunakan saat struct sudah di inisialisasi pada variabel dan dipanggil dengan atribut dari tipe data

Hasil

```

[root@localhost Documents]# go run no4kanan.go
area: 50
perim: 30
area: 50
perim: 30

```

```

3
5
[root@localhost Documents]# go run no2kanan.go
7 is odd
8 is divisible by 4
9 has 1 digit
[root@localhost Documents]# go run no3kiri.go
emp: [0 0 0 0 0]
set: [0 0 0 0 100]
get: 100
len: 5
dcl: [1 2 3 4 5]
2d: [[0 1 2] [1 2 3]]
[root@localhost Documents]# go run no3kanan.go
1+2 = 3
1+2+3 = 6
[root@localhost Documents]# go run no4kiri.go
{Bob 20}
{Alice 30}
{Fred 0}
Sean
50
51
[root@localhost Documents]#

```

5. Program 1

```

package main

import "fmt"

func vals() (int, int) {
    return 3, 7
}

func main() {
    a, b := vals()
    fmt.Println(a)
    fmt.Println(b)

    _, c := vals()
    fmt.Println(c)
}

```

- Multiple return adalah suatu kebutuhan dimana data yang dikembalikan harus banyak, biasanya memiliki beberapa tipe data sebagai nilai balik.

Program 2

```

package main

import "flag"
import "fmt"

func main() {
    wordPtr := flag.String("word", "foo", "a string")

    numPtr := flag.Int("numb", 42, "an int")
    boolPtr := flag.Bool("fork", false, "a bool")

    var svar string
    flag.StringVar(&svar, "svar", "bar", "a string var")

    flag.Parse()

    fmt.Println("word:", *wordPtr)
    fmt.Println("numb:", *numPtr)
    fmt.Println("fork:", *boolPtr)
    fmt.Println("svar:", svar)
    fmt.Println("tail:", flag.Args())
}

```

- Command line adalah interaksi dengan sistem operasi atau perangkat lunak komputer dengan mengetikkan perintah untuk menjalankan tugas tertentu.

Hasil

```
root@localhost:/home/farrasisca/Documents
{Fred 0}
Sean
50
51
[root@localhost Documents]# go run no4kanan.go
# command-line-arguments
./no4kanan.go:6:15:

[root@localhost Documents]# go run no4kanan.go
area: 50
perim: 30
area: 50
perim: 30
[root@localhost Documents]# go run no5kiri.go
3
7
7
[root@localhost Documents]# go run no5kanan.go
word: foo
numb: 42
fork: false
svar: bar
tail: []
[root@localhost Documents]#
```