# LAPORAN PRAKTIKUM
# ALGORITMA DAN PEMROGRAMAN 2

## LATIHAN SOAL 2

## SOAL TIPE A, B dan C


**DISUSUN OLEH:**

**Keishin Naufa Alfaridzhi**

**103112400061**

**S1 IF-12-01**



**S1 TEKNIK INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2025**

# I. DASAR TEORI

## A. Bahasa Yang Digunakan

Pada praktikum ini bahasa pemrograman yang digunakan adalah bahasa pemrograman Go, sesuai dengan modul yang menjadi acuan praktikum. Golang (atau) Go adalah bahasa pemrograman baru, yang mulai dilirik oleh para developer karena kelebihan-kelebihan yang dimilikinya. Sudah banyak Perusahaan besar yang menggunakan bahasa ini untuk produk-produk mereka hingga di level production.

## B. Komentar

Komentar biasa dimanfaatkan untuk menyisipkan catatan pada kode program, menulis penjelasan atau deskripsi mengenai suatu blok kode, atau bisa juga digunakan untuk me-remark kode (men-non-aktifkan kode yang tidak digunakan). Komentar akan diabaikan Ketika kompilasi maupun eksekusi program.

Ada 2 jenis komentar di Golang, yaitu inline dan multiline.

### 1. Komentar Inline

Penulisan komentar jenis ini diawali dengan tanda *double slash* (//) lalu diikuti pesan komentarnya. Komentar inline hanya berlaku untuk satu baris pesan saja. Jika pesan komentar lebih dari satu baris, maka tanda *double slash* harus ditulis lagi di baris selanjutnya.

### 2. Komentar Multiline

Komentar yang cukup panjang akan lebih rapi jika ditulis menggunakan teknik komentar multiline. Ciri dari komentar jenis ini adalah penulisannya diawali dengan tanda (/*) dan diakhiri (*/).

## C. Variabel

Golang mengadopsi 2 jenis penulisan variabel, yang dituliskan tipe data-nya dan yang tidak. Kedua cara tersebut intinya adalah sama, pembedanya hanyalah cara penulisannya saja. Untuk penulisan variabel dengan tipe data, keyword *var* digunakan untuk deklarasi variabel kemudian diakhiri dengan tipe data misalnya *string*. Kemudian untuk penulisan variabel tanpa tipe data, variabel dideklarasikan dengan menggunakan metode type inference. Penandanya tipe data tidak dituliskan pada saat deklarasi. Pada penggunaan metode ini, operand (=) harus diganti dengan (:=) dan keyword *var* dihilangkan.

Golang memiliki aturan unik yang tidak dimiliki bahasa lain, yaitu tidak boleh ada satupun variabel yang menganggur. Artinya, semua variabel yang dideklarasikan harus digunakan. Jika terdapat variabel yang tidak digunakan tapi dideklarasikan, program akan gagal dikompilasi. Untuk mengatasi itu, golang memiliki variabel yaitu underscore. Underscore ( _ ) adalah predefined variabel yang bisa dimanfaatkan untuk menampung nilai yang tidak dipakai.

## D. Tipe Data

Golang mengenal beberapa jenis tipe data, diantaranya adalah tipe data numerik (decimal dan non-desimal), string, dan boolean.

1. Tipe Data Numerik Non-Desimal (uint, int)
2. Tipe Data Numerik Desimal (float64, float32)
3. Tipe Data Bool (true, false)
4. Tipe Data String (string, " ")

## E. Operator Aritmatika

Operator aritmatika merupakan operator yang digunakan untuk operasi yang sifatnya perhitungan. Golang mendukung beberapa operator aritmatika standar, yaitu:

1. Penjumlahan (+)
2. Pengurangan (-)
3. Perkalian (*)
4. Pembagian (/)
5. Modulus atau sisa hasil pembagian (%)

## F. Seleksi Kondisi

Seleksi kondisi pada program berguna untuk mengontrol sebuah blok kode yang akan dieksekusi. Yang dijadikan acuan oleh selksi kondisi adalah nilai bertipe bool, bisa berasal dari variabel, ataupun hasil operasi perbandingan. Nilai tersebut menentukan blok kode mana yang akan dieksekusi. Go memiliki 2 macam keyword untuk selesksi kondisi, yaitu if else dan switch.

1. **If Expression**

   If adalah salah satu kata kunci yang digunakan dalam percabangan. Percabangan artinya kitabisa mengeksekusi kode program tertentu ketika suatu kondisi terpenuhi. Hampir semua bahasa pemrograman mendukung if expression.

2. **Else if expression**

   Terkadang kita butuh membuat beberapa kondisi. Kasus seperti ini dapat menggunakan else if expression. If mendukung short statement sebelum kondisi.

Hal ini sangat cocok untuk membuat statement yang sederhana sebelum melakukan pengecekan terhadap kondisi.

3. **Switch-Case**

   Switch merupakan seleksi kondisi yang sifatnya fokus pada satu variabel, lalu kemudian di-cek nilainya. Contoh sederhananya seperti penentuan apakah nilai variabel x adalah: 1, 2, 3, atau lainnya. Perlu diketahui, switch pada pemrograman Go memiliki perbedaan dibanding bahasa lain. Di Go, ketika sebuah case terpenuhi, tidak akan dilanjutkan ke pengecekan case selanjutnya, meskipun tidak ada keyword "break" di situ. Konsep ini berkebalikan dengan switch pada umumnya pemrograman lain (yang ketika sebuah case terpenuhi, maka akan tetap dilanjut mengecek case selanjutnya kecuali ada keyword "break").

## G. Perulangan

Perulangan merupakan proses mengulang dan mengeksekusi blok kode tanpa henti sesuai dengan kondisi yang dijadikan acuan. Biasanya disiapkan variabel untuk iterasi atau penanda kapan perulangan akan dihentikan.

### a. For Loop

For loop merupakan statement perulangan dasar dan cukup sering ditemui. Format for loop yaitu sebagai berikut.

- *Init Statement*: bagian ini akan dieksekusi sebelum perulangan dimulai. Biasanya diisi dengan mendeklarasi variabel iterasi.
- *Condition Expression*: bagian ini akan dicek dan dieksekusi setiap perulangan yang dilakukan, jika true maka perulangan akan terus berjalan hingga kondisi bernilai false.
- *Post Statement*: statement ini akan dieksekusi pada akhir iterasi. Jika terdapat range, maka perulangan akan dieksekusi untuk setiap item pada range.

### b. While Loop

While loop merupakan perulangan yang akan terus berjalan hingga suatu kondisi terpenuhi. Penulisan while loop adalah dengan menuliskan kondisi setelah keyword for (hanya kondisi). Deklarasi dan iterasi variabel counter tidak dituliskan setelah keyword, hanya kondisi perulangan saja. Konsepnya mirip seperti while milik bahasa pemrograman lain.

**c. Repeat Until**

Untuk Repeat Until ini mirip seperti for loop biasa namun hanya menggunakan inisiasi dan kondisi saja.

## H. Fungsi

Dalam konteks pemrograman, fungsi adalah sekumpulan blok kode yang dibungkus dengan nama tertentu. Penerapan fungsi yang tepat akan menjadikan kode lebih modular dan juga *dry* (singkatan dari *don't repeat yourself*) yang artinya kita tidak perlu menuliskan banyak kode untuk kegunaan yang sama berulang kali. Cukup deklarasikan sekali saja blok kode sebagai suatu fungsi, lalu panggil sesuai kebutuhan.

### 1. Penerapan Fungsi

Sebenarnya kita sudah mengimplementasikan fungsi pada banyak praktek sebelumnya, yaitu fungsi `main()`. Fungsi `main()` sendiri merupakan fungsi utama pada program Go, yang akan dieksekusi ketika program dijalankan.

Selain fungsi `main()`, kita juga bisa membuat fungsi lainnya. Dan caranya cukup mudah, yaitu dengan menuliskan keyword `func` kemudian diikuti nama fungsi, lalu kurung `()` (yang bisa diisi parameter), dan diakhiri dengan kurung kurawal untuk membungkus blok kode.

Parameter merupakan variabel yang menempel di fungsi yang nilainya ditentukan saat pemanggilan fungsi tersebut. Parameter sifatnya opsional, suatu fungsi bisa tidak memiliki parameter, atau bisa saja memeliki satu atau banyak parameter (tergantung kebutuhan).

### 2. Fungsi dengan Nilai Balik / Return Value

Selain parameter, fungsi bisa memiliki attribute **return value** atau nilai balik. Fungsi yang memiliki return value, saat deklarasinya harus ditentukan terlebih dahulu tipe data dari nilai baliknya.

### 3. Fungsi tanpa Nilai Balik / Return Value

Fungsi juga dapat tidak memiliki nilai balik yang dapat disebut juga sebagai Prosedur. Prosedur dapat dianggap sebagai potongan beberapa instruksi program menjadi suatu instruksi baru yang dibuat untuk mengurangi kerumitan dari kode program yang kompleks pada suatu program yang besar.

## II. LATIHAN SOAL 2

### A. Tipe A

1. Latihan no. 1

Source Code:

```
package main
import "fmt"

func main() {
    var (
        jam, menit float32
        member_103112400061 bool
        voucher int
    )
    fmt.Print("Masukkan durasi (jam): ")
    fmt.Scan(&jam)
    fmt.Print("Masukkan durasi (menit): ")
    fmt.Scan(&menit)
    fmt.Print("Apakah member? (true/false): ")
    fmt.Scan(&member_103112400061)
    fmt.Print("Masukkan nomor voucher (jika ada): ")
    fmt.Scan(&voucher) // NIM: 103112400061
    biaya := sewaSepeda(jam, menit, member_103112400061, voucher)
    fmt.Printf("Biaya sewa setelah diskon (jika memenuhi syarat): Rp %.2f\n", biaya)
}

func sewaSepeda(jam, menit float32, member_103112400061 bool, voucher int)
float32 {
    var biaya float32
    if menit > 10 {
        jam = jam + 1
    } // AUTHOR: KEISHIN NAUFA ALFARIDZHI
    if member_103112400061 {
        biaya = jam * 3500
    } else {
        biaya = jam * 5000
    }
    if voucher > 9999 && voucher < 1000000 {
        biaya = biaya - (biaya * 0.1)
    }

    return biaya
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeA\103112400061
Masukkan durasi (jam): 2
Masukkan durasi (menit): 30
Apakah member? (true/false): true
Masukkan nomor voucher (jika ada): 123456
Biaya sewa setelah diskon (jika memenuhi syarat): Rp 9450.00
```

Deskripsi Program:

Mencari total biaya sewa. Memiliki suatu function bernama **sewaSepeda** dengan parameter jam, menit, member dan voucher. Di dalamnya dilakukan cek jika menit >10 maka +1 jam; cek apakah member atau bukan; cek apakah voucher valid atau tidak; kemudian return total biaya.

2. Latihan no. 2

Source Code:

```
package main
import "fmt"

func main() {
    var a, b int
    fmt.Print("Masukkan nilai a: ")
    fmt.Scan(&a)
    fmt.Print("Masukkan nilai b: ")
    fmt.Scan(&b)
    perfectNumber(a, b)
}

func perfectNumber(a, b int) {
    var perfectNumber int
    fmt.Printf("Perfect numbers antara %v dan %v: ", a, b)
    for i := b; i > a; i-- {
        var j int
        var factors_103112400061 int = 0
        // AUTHOR: KEISHIN NAUFA ALFARIDZHI
        for j = 1; j < i; j++ {
            if i%j == 0 {
                // fmt.Print(j, " ")
                factors_103112400061 += j
            }
        }
        if factors_103112400061 == i { // NIM: 103112400061
            perfectNumber = factors_103112400061
            fmt.Printf("%v ", perfectNumber)
        }
    }
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeA\1031124000
AL2.go"
Masukkan nilai a: 3
Masukkan nilai b: 13
Perfect numbers antara 3 dan 13: 6
```

Deskripsi Program:

Program mencari perfect number antara dua variabel. Memiliki suatu prosedur bernama **perfectNumber** dengan 2 parameter bertipe data integer. Kondisi b harus lebih besar daripada a. Kemudian cek faktor menggunakan iterasi jika setiap bilangan antara a dan b apakah setiap faktornya jika dijumlahkan merupakan bilangan itu sendiri. Jika memenuhi kondisi tersebut maka perfect number ditemukan dan dicetak.

3. Latihan no. 3

Source Code:

```
package main
import "fmt"

func main() {
    var x, y int
    fmt.Print("Masukkan nilai x: ")
    fmt.Scan(&x)
    fmt.Print("Masukkan nilai y: ")
    fmt.Scan(&y)
    fmt.Println("Jumlah pertemuan dalam setahun:", rendezvous(x, y))
}

func rendezvous(x, y int) int {
    var total_103112400061 int
    for i := 1; i <= 365; i++ {
        if i % x == 0 && i % y != 0 {
            total_103112400061++
        }
    }
    return total_103112400061
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeA\1031124000
AL3.go"
Masukkan nilai x: 2
Masukkan nilai y: 5
Jumlah pertemuan dalam setahun: 146
```

Deskripsi Program:

Program pertemuan mata-mata dalam setahun. Memiliki funciton bernama **rendezvous** dengan 2 parameter integer. Kemudian dalam iterasi dengan batas 365 (hari dalam satu tahun), cek kondisi hari kelipatan x tapi tidak dengan hari kelipatan y. Setiap kali kondisi terpenuhi maka hari peretemuan mata-mata akan bertambah.

## B. Tipe B

4. Latihan no. 1

Source Code:

```
package main
import "fmt"

func main() {
    var a, b int
    fmt.Print("Masukkan nilai a: ")
    fmt.Scan(&a)
    fmt.Print("Masukkan nilai b: ")
    fmt.Scan(&b)
    // NIM: 103112400061
    fmt.Println("Banyaknya angka ganjil:", ganjil(a,b))
}

func ganjil(a, b int) int {
    var ganjil_103112400061 int
    for i := a; i <= b; i++ {
        if i%2 != 0 { // AUTHOR: KEISHIN NAUFA ALFARDZHI
            ganjil_103112400061++
        }
    }
    return ganjil_103112400061
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeB\103112400
AL1.go"
Masukkan nilai a: 1
Masukkan nilai b: 1000
Banyaknya angka ganjil: 500
```

Deskripsi Program:

Mencari total banyaknya angka ganjil antara dua variabel a dan b. Memiliki function bernama **ganjil** dengan 2 parameter a dan b. Menggunakan iterasi sebagai algoritmanya dengan inisiasi i = a dan kondisi i <= b. Didalamnya terdapat seleksi kondisi berupa

apakah i merupakan bilangan ganjil, jika iya maka variabel ganjil bertambah 1 tiap perulangannya.

5. Latihan no. 2

Source Code:

```
package main
import "fmt"

func main() {
    var (
        menu, orang, rombongan_103112400061, sisa int
    )
    fmt.Print("Masukkan jumlah rombongan_103112400061: ")
    fmt.Scan(&rombongan_103112400061)
    for i := 1; i <= rombongan_103112400061; i++ {
        fmt.Print("Masukkan jumlah menu, jumlah orang, dan status sisa makanan (0 untuk tidak, 1 untuk iya): ")
        fmt.Scan(&menu, &orang, &sisa)
        fmt.Printf("Total biaya untuk rombongan %v: Rp %v\n", i, restoran(menu, orang, sisa))
    }
}

func restoran(menu, orang, sisa int) int {
    var biaya int
    if menu <= 3 && sisa == 0 {
        biaya = 10000
    } else if menu > 3 && sisa == 0 {
        biaya = 10000
        for i := menu; i > 3; i-- {
            biaya += 2500
        }
    }

    if menu <= 3 && sisa == 1 {
        biaya = 10000 * orang
    } else if menu > 3 && sisa == 1 {
        biaya = 10000
        for i := menu; i > 3; i-- {
            biaya += 2500
        }
        biaya *= orang
    }

    return biaya
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeB\103112400061_
AL2.go"
Masukkan jumlah rombongan_103112400061: 2
Masukkan jumlah menu, jumlah orang, dan status sisa makanan (0 untuk tidak, 1 untuk iya): 3 12 0
Total biaya untuk rombongan 1: Rp 10000
Masukkan jumlah menu, jumlah orang, dan status sisa makanan (0 untuk tidak, 1 untuk iya): 2 15 1
Total biaya untuk rombongan 2: Rp 150000
```

Deskripsi Program:

Program untuk menghitung total biaya menu sebuah restoran. Memiliki function bernama **restoran** dengan 3 parameter yaitu orang, menu dan sisa. Menggunakan seleksi kondisi untuk cek apakah menu <= 3 atau sebaliknya, dan apakah sisa = 0 atau sisa = 1. Dengan kriteria sebagai berikut: Jika menu > 3 maka akan dikenakan biaya tambahan 2,500 tiap menu yang ditambahkan, dan jika sisa = 1 maka tiap orang akan dikenakan biaya sebesar 10,000.

Kemudian pada function main digunakan iterasi untuk menghitung tiap rombongan. Dalam iterasi ini juga function **restoran** akan dipanggil.

6. Latihan no. 3

Source Code:

```
package main
import "fmt"

func main() {
   var jumlah_103112400061 int
   fmt.Println("Masukkan bilangan (negatif untuk berhenti):")
   jumlah_103112400061 = kelipatanEmpat(0)
   fmt.Println("jumlah bilangan kelipatan 4:", jumlah_103112400061) // NIM:
103112400061
}

func kelipatanEmpat(jumlah_103112400061 int) int {
   var x int
   fmt.Scan(&x)
   // AUTHOR: KEISHIN NAUFA ALFARIDZHI
   if x < 0{
      return jumlah_103112400061
   }

   if x%4 == 0 {
      jumlah_103112400061 += x
   }
```

```
    return kelipatanEmpat(jumlah_103112400061)
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeB\1
AL3.go"
Masukkan bilangan (negatif untuk berhenti):
2 3 4 5 6 -1
jumlah bilangan kelipatan 4: 4
```

Deskripsi Program:

Program penjumlahan bilangan kelipatan 4 dengan input terus menerus hingga terdapat suatu bilangan negatif. Memiliki function bernama **kelipatanEmpat** dengan 1 parameter bernama jumlah. Kita perlu 1 variabel lagi untuk menampung nilai inputan, kita namakan saja x. Dalam function, akan digunakan package fmt.Scan() untuk inputannya. Kemudian terdapat base case untuk cek jika bilangan < 0 maka akan return langsung jumlahnya. Lalu cek apakah x merupakan bilangan kelipatan 4, jika iya maka akan ditambahkan ke dalam jumlah. Terakhir terdapat rekursif untuk memanggil kembali function jika kondisi bilangan < 0 sebelumnya tidak terpenuhi (program akan jalan terus menerus hingga base case tercapai).

## C. Tipe C

7. Latihan no. 1

Source Code:

```
package main
import "fmt"

func main() {
    var x_103112400061 int
    fmt.Print("Masukkan bilangan bulat positif (>10): ")
    fmt.Scan(&x_103112400061)
    splitter(x_103112400061)
}

func countDigits(x int) int {
    var digits int = 0

    if x <= 0 {
        return 1
    }

    for x > 0 {
```

```
        digits++
        x /= 10
    }
    return digits
}

func splitter(x int) {
    if x <= 10 {
        fmt.Println("Bilangan harus >10 !")
    } else {
        var digits int = countDigits(x)
        var parts int = digits / 2
        var divisor int = 1

        for i := 1; i <= parts; i++ {
            divisor *= 10
        }

        bil1 := x / divisor
        bil2 := x % divisor
        jumlah := bil1 + bil2

        fmt.Println("Bilangan 1:", bil1)
        fmt.Println("Bilangan 2:", bil2)
        fmt.Println("Hasil penjumlahan:", jumlah)
    }
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeC\103112400061_
AL1.go"
Masukkan bilangan bulat positif (>10): 15
Bilangan 1: 1
Bilangan 2: 5
Hasil penjumlahan: 6

D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeC\103112400061_
AL1.go"
Masukkan bilangan bulat positif (>10): 12345
Bilangan 1: 123
Bilangan 2: 45
Hasil penjumlahan: 168
```

Deskripsi Program:

Memecah suatu bilangan bulat menjadi dua bagian lalu mencari hasil penjumlahan dari kedua bagian bilangan bulat tersebut. Terdapat 1 function dan 1 prosedur. Function bernama **countDigits** untuk menghitung berapa panjang bilangan yang diinputkan. Cara kerjanya adalah dengan menggunakan for loop dengan kondisi x > 0. Selama kondisi true maka akan membagi inputan dengan 10 tiap perulangan dan melakukan increment untuk variabel digits. Setelah kondisi false maka akan langsung return digits.

Untuk prosedur bernama **splitter** merupakan program utamanya, dimulai dengan syarat kondisi x harus lebih dari 10. Memiliki 3 variabel yaitu digits (ini berdasarkan function **countDigits**), parts (digits / 2) dan divisor. Kemudian terdapat iterasi dengan kondisi i < parts guna mencari titik tengah dari bilangan yang diinputkan. Di dalam iterasi terdapat divisor * 10 tiap perulangannya, untuk memecah bilangan menjadi dua. Setelah itu deklarasi bil1 dan bil2, yang mana bil1 akan menjadi bilangan bagian sebelah kiri dengan cara bilangan yang diinputkan akan dibagi dengan divisor, lalu bil2 akan menjadi bilangan bagian sebelah kanan dengan cara bilangan yang diinputkan akan dimodulo dengan divisor. Kemudian tinggal menjumlahkan saja bil1 dan bil2 untuk mendapatkan hasil akhir penjumlahan.

8. Latihan no. 2

Source Code:

```
package main
import "fmt"

func main() {
    var (
        x, n_103112400061 int
        jumlahHadiahA, jumlahHadiahB, jumlahHadiahC int
    )
    fmt.Print("Masukkan jumlah peserta: ")
    fmt.Scan(&n_103112400061)
    for i := 1; i <= n_103112400061; i++ {
        fmt.Printf("Masukkan nomor kartu peserta ke-%v: ", i)
        fmt.Scan(&x)

        result := cekHadiah(x)

        if result == "Hadiah A" {
            jumlahHadiahA++
        } else if result == "Hadiah B" {
            jumlahHadiahB++
        } else if result == "Hadiah C" {
            jumlahHadiahC++
        }

        fmt.Println(result)
    }

    fmt.Println()
    fmt.Println("Jumlah yang memperoleh Hadiah A:", jumlahHadiahA)
    fmt.Println("Jumlah yang memperoleh Hadiah B:", jumlahHadiahB)
```

```
    fmt.Println("Jumlah yang memperoleh Hadiah C:", jumlahHadiahC)
}

func cekHadiah(x int) string {
    var digits []int
    var temp int = x
    var allSama bool = true
    var allDiff bool = true

    for temp > 0 {
        digits = append([]int{temp%10}, digits...)
        temp /= 10
    }

    for i := 1; i < len(digits); i++ {
        if digits[i] != digits[0] {
            allSama = false
            break
        }
    }

    if allSama {
        return "Hadiah A"
    }
        for i := 0; i < len(digits); i++ {
            for j := i + 1; j < len(digits); j++ {
                if digits[i] == digits[j] {
                    allDiff = false
                    break
                }
            }
            if !allDiff {
                break
            }
        }

    if allDiff {
        return "Hadiah B"
    } else {
        return "Hadiah C"
    }
}

}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeC\103112400061_
AL2.go"
Masukkan jumlah peserta: 3
Masukkan nomor kartu peserta ke-1: 333
Hadiah A
Masukkan nomor kartu peserta ke-2: 123
Hadiah B
Masukkan nomor kartu peserta ke-3: 898
Hadiah C

Jumlah yang memperoleh Hadiah A: 1
Jumlah yang memperoleh Hadiah B: 1
Jumlah yang memperoleh Hadiah C: 1
```

Deskripsi Program:

Program untuk cek hadiah yang didapat berdasarkan nomor unik yang didapat kemudian cetak jumlah tiap hadiah yang diperoleh. Memiliki function bernama **cekHadiah** dengan 1 parameter x. Pertama deklarasi sebuah array bernama digits, kemudian variabel bernama temp untuk menampung nilai sementara dari parameter x, lalu 2 variabel boolean yaitu allSama dan allDiff. Dimulai dengan for loop dengan kondisi temp > 0, didalamnya terdapat array digits yang akan ditambahkan value berupa temp dimodulo dengan 10 yang mana akan mendapatkan nilai paling kanan dari temp (cth. 123 maka didapatkan 3). Kemudian terdapat temp dibagi dengan 10 tiap perulangannya yang digunakan untuk mengeliminasi value yang telah didapatkan sehingga dapat berlanjut untuk menambahkan value selanjutnya ke dalam array digits.

Selanjutnya terdapat for loop untuk cek tiap value dalam array digits, didalamnya terdapat kondisi jika value selanjutnya tidak sama dengan value pertama maka allSama akan set jadi false dan for loop akan break. Setelah itu akan dicek jika allSama apakah true atau tidak. Jika true akan return “Hadiah A”, jika tidak maka akan lanjut untuk cek apakah tiap value dalam array digits berbeda untuk mendapatkan return “Hadiah B”, jika ternyata didapatkan allDiff = false yang mana tidak seluruh value berbeda maka akan return “Hadiah C”.

Pada function main akan dipanggil function **cekHadiah** dalam sebuah iterasi. Pada function main jugalah akan dicek berapa jumlah tiap hadiah yang didapatkan.

9. Latihan no. 3

Source Code:

```
package main
import "fmt"

func main() {
   var n_103112400061, m int
   fmt.Print("Masukkan bilangan n: ")
   fmt.Scan(&n_103112400061)
   fmt.Print("Masukkan bilangan m: ")
   fmt.Scan(&m)
   fmt.Printf("Hasil dari %v x %v = %v\n", n_103112400061, m,
perkalianJumlah(n_103112400061, m))
}

func perkalianJumlah(n, m int) int {
   if (m == 0) {
      return 0
   }
   return (n + perkalianJumlah(n, m-1))
}
```

Output:

```
D:\Tugas\SEM2\Alpro\Praktikum\latso>go run "d:\Tugas\SEM2\Alpro\Praktikum\latso\tipeC\103112400061
AL3.go"
Masukkan bilangan n: 5
Masukkan bilangan m: 6
Hasil dari 5 x 6 = 30
```

Deskripsi Program:

Program mencari nilai suatu perkalian dengan penjumlah berulang menggunakan rekursif. Terdapat function **perkalianJumlah** dengan 2 parameter n dan m. Pada function ini terdapat base case untuk rekursif yaitu jika m = 0 maka return 0. Selanjutnya pada rekursif akan dilakukan n + **perkalianJumlah(**n, m-1**)** secara terus menerus hingga mencapai basecase yaitu m = 0.

# III. DAFTAR PUSTAKA


Noval Agung Prayogo. *Dasar Pemrograman Golang.* Diakses pada 01 Oktober 2024. https://dasarpemrogramangolang.novalagung.com

Annisa Nur Isnaeni. *Golang — Seleksi Kondisi.* Diakses pada 01 Oktober 2024.

https://medium.com/@annisaisna/golang-seleksi-kondisi-f988ead004b4

Parvez Alam, *Golang for loop example | Golang Loops Tutorial – Phpflow.com* https://medium.com/@parvez1487/golang-for-loop-example-golang-loops-tutorial-phpflow-com-f4b2b0e57944