LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL 4
PROSEDUR

DISUSUN OLEH:

ANASTASIA ADINDA NARENDRA INDRIANTO

103112400085

S1 IF-12-01

DOSEN:

Dimas Fanny Hebrasianto Permadi, S.ST., M.KOM

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

1. Pengertian Bahasa Pemorgraman Golang

Golang adalah bahasa pemrograman yang memiliki sejumlah kelebihan yang tidak dimiliki bahasa pemrograman yang lainnya. Hadirnya bahasa pemrograman Go Language (Golang) semakin dirasakan oleh para pengembang. Tidak heran jika banyak orang yang mulai belajar bahasa pemrograman yang satu ini.
Golang merupakan bahasa pemrograman yang dibuat Google dan tujuannya untuk menyempurnakan bahasa pemrograman yang ada, seperti C, Phyton dan yang lainnya. Golang bisa jadi pilihan yang tepat saat membuat aplikasi baru.

2. Pengertian Input dan Output

i. *Input* atau masukan adalah data yang diberikan ke dalam program. Masukan ini bisa berasal dari berbagai sumber, seperti pengguna melalui keyboard, mouse, atau suara, dan juga bisa berasal dari sensor atau perangkat lain yang terhubung ke komputer. Dalam pemrograman, input diperlakukan sebagai bahan baku yang akan diproses oleh program.

ii. *Output* atau keluaran adalah hasil yang diproduksi oleh program setelah mengolah input. Output bisa berbentuk tampilan pada layar, cetakan pada printer, suara, ataupun penyimpanan data ke file. Inti dari program yang kita tulis sebenarnya adalah menghasilkan output yang bermakna dari input yang diberikan.

3. Pengertian Tipe Data

Tipe data adalah kategori data yang akan diproses oleh sebuah program komputer. Dengan tipe data, program dapat mengetahui cara menangani data dengan benar, seperti melakukan operasi matematika pada angka atau memanipulasi string untuk tujuan tertentu.Secara sederhana, pengertian tipe data adalah kategori data yang akan diproses oleh sebuah program komputer. Dengan tipe data, program dapat mengetahui cara menangani data dengan benar, seperti melakukan operasi matematika pada angka atau memanipulasi string untuk tujuan tertentu. Contoh paling sederhana dari tipe data adalah tipe data **integer** yang digunakan untuk menyimpan angka bulat atau tipe data **string** yang digunakan untuk menyimpan rangkaian karakter. Penggunaan tipe data yang benar dalam suatu program memastikan bahwa data diolah secara tepat dan mengurangi risiko kesalahan atau bug dalam program tersebut.

4. Fungsi Tipe Data

i. **Menentukan Jenis Nilai**: Tipe data memberi tahu program jenis nilai yang akan disimpan dalam variabel. Misalnya, jika kita ingin menyimpan angka, kita menggunakan tipe data integer atau float, sedangkan untuk menyimpan teks, kita menggunakan tipe data string. Ini membantu program memahami bagaimana cara menangani data tersebut.

ii. **Efisiensi Penggunaan Memori**: Setiap tipe data memerlukan jumlah memori yang berbeda. Jika kita memilih tipe data yang tepat, program bisa menggunakan memori lebih efisien. Misalnya, menggunakan tipe data yang lebih kecil untuk angka yang tidak terlalu besar akan menghemat ruang di memori.

iii. **Menjamin Konsistensi Data**: Dengan menentukan tipe data, kita memastikan bahwa variabel hanya bisa menyimpan jenis nilai yang sesuai. Misalnya, jika kita mendeklarasikan variabel sebagai tipe integer, program tidak akan bisa memasukkan teks atau jenis data lainnya ke dalamnya. Ini membantu menjaga agar data tetap konsisten dan sesuai dengan yang diharapkan.

iv. **Memudahkan Operasi pada Data**: Tipe data juga menentukan operasi apa saja yang bisa dilakukan pada data. Misalnya, kita bisa melakukan perhitungan

matematika pada angka, tetapi kita tidak bisa melakukan hal yang sama pada teks. Jadi, dengan menentukan tipe data yang tepat, kita bisa melakukan operasi yang sesuai dengan jenis data yang kita miliki.

5. **Pengertian If-Else dan Fungsinya**

   If-else adalah struktur percabangan dalam pemrograman yang digunakan untuk mengeksekusi kode berdasarkan suatu kondisi. Jika kondisi dalam if bernilai true, maka blok kode di dalamnya akan dijalankan. Jika tidak, program akan memeriksa kondisi dalam else if sebagai alternatif. Jika semua kondisi false, maka else akan dijalankan sebagai pilihan terakhir. Dengan menggunakan if-else, program dapat mengambil keputusan dan menjalankan instruksi yang sesuai berdasarkan kondisi yang diberikan.

   If-else berfungsi untuk mengontrol alur program dengan mengeksekusi kode berdasarkan suatu kondisi. If digunakan untuk memeriksa apakah suatu kondisi bernilai true, jika ya, maka blok kode di dalamnya akan dijalankan. Jika tidak, program dapat menggunakan else-if untuk mengevaluasi beberapa kondisi tambahan secara berurutan. Jika semua kondisi false, maka else akan dieksekusi sebagai pilihan terakhir. Selain itu, terdapat if-else bersarang, yang memungkinkan pengecekan kondisi di dalam kondisi lain untuk menangani keputusan yang lebih kompleks.

6. **Pengertian While Loop dan Fungsinya**

   While loop adalah metode perulangan yang mengeksekusi blok kode selama kondisi yang diberikan bernilai true dan akan berhenti ketika kondisi berubah menjadi false. Perulangan ini sangat berguna dalam kasus di mana jumlah iterasi belum diketahui secara pasti, seperti membaca input hingga valid atau menjalankan suatu proses hingga syarat tertentu terpenuhi. Fungsinya adalah;

   i. **Menjalankan Perulangan Berdasarkan Kondisi** – While loop memastikan suatu proses terus berjalan selama kondisi bernilai **true**.
   ii. **Mengatasi Iterasi yang Tidak Diketahui Jumlahnya** – Digunakan ketika jumlah perulangan tidak bisa ditentukan sejak awal, seperti menunggu input yang valid.
   iii. **Mengoptimalkan Kontrol Program** – Membantu mengelola eksekusi kode agar hanya berjalan saat kondisi tertentu terpenuhi, meningkatkan efisiensi program.

7. **Pengertian Fungsi dan Manfaatnya**

   Fungsi merupakan rangkaian instruksi yang menghasilkan suatu nilai dengan memetakkan input ke output tertentu. Dalam pemrograman, suatu subprogram dikategorikan sebagai fungsi apabila memiliki deklarasi tipe nilai yang dikembalikan dan menggunakan kata kunci return dalam tubuhnya. Fungsi digunakan dalam berbagai situasi, seperti menetapkan nilai ke suatu variabel melalui assignment, menjadi bagian dari suatu ekspresi, atau digunakan sebagai argumen dalam subprogram lain. Karena fungsi selalu menghasilkan nilai, penamaannya sebaiknya bersifat deskriptif dan mencerminkan hasil yang diberikan, seperti *median*, *rerata*, *nilaiTerbesar*, atau *selesai*.

   Function memungkinkan programmer membagi kode menjadi segmen-segmen kecil yang lebih terkelola, masing-masing melakukan bagian tertentu dari tugas yang lebih besar. Tidak hanya membantu dalam mengorganisasi kode secara lebih efisien, hal ini juga memudahkan pemeliharaan dan pengujian kode.

8. **Pengertian Prosedur dan Manfaatnya**

   Prosedur adalah kumpulan instruksi yang dikemas menjadi satu kesatuan untuk menyederhanakan kode dalam program besar. Prosedur tidak mengembalikan nilai karena tidak memiliki deklarasi tipe nilai kembalian dan tidak menggunakan kata kunci *return*. Ketika dipanggil, prosedur langsung memberikan efek pada program utama, seperti halnya instruksi dasar atau fungsi bawaan (*built-in*). Nama prosedur sebaiknya menggunakan kata kerja atau kata yang menggambarkan proses, seperti *cetak*, *hitungRerata*, atau *cariNilai*. Hal ini bertujuan agar lebih mudah dipahami dan sesuai dengan fungsinya dalam program. Dengan menggunakan prosedur, kode program menjadi lebih terstruktur, mudah dibaca, dan dikelola.

   Manfaat utama prosedur adalah meningkatkan keterbacaan dan keteraturan kode, sehingga lebih mudah dikelola dan diperbaiki. Dengan memecah program menjadi bagian-bagian kecil, prosedur juga membantu dalam mengurangi pengulangan kode (code redundancy), meningkatkan efisiensi, serta mempermudah debugging dan pengembangan program. Hal ini membuat program lebih modular dan fleksibel untuk diperbarui di masa mendatang.

II. GUIDED

1. Guide 1

Source Code:

```
// Anastasia Adinda N.I
// Prosedur Gaji
package main

import "fmt"

func hitungGaji(nama string, gajiPokok float64, jamLembur int) {
   bonusLembur := float64(jamLembur) * 50000
   totalGaji := gajiPokok + bonusLembur

   fmt.Println("\n=== Slip Gaji ===")
   fmt.Println("Nama Karyawan :", nama)
   fmt.Printf("Gaji Pokok : Rp%.2f\n", gajiPokok)
   fmt.Printf("Bonus Lembur : Rp%.2f (%d jam x Rp.50,000)\n", bonusLembur,
jamLembur)
   fmt.Printf("Total Gaji : Rp%.2f\n", totalGaji)

}

func main() {
   var nama string
   var gajiPokok float64
   var jamLembur int

   //Input pengguna
   fmt.Print("Masukan Nama Karyawan: ")
   fmt.Scanln(&nama)

   fmt.Print("Masukan Gaji Pokok: ")
   fmt.Scanln(&gajiPokok)

   fmt.Print("Masukan Jumlah Jam Lembur: ")
   fmt.Scanln(&jamLembur)

   //Memanggil prosedur dengan data dari pengguna
   hitungGaji(nama, gajiPokok, jamLembur)
}
```

Output:

```
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4> go run "d:\Semester 2\ALPRO2 CODING\103112400085_MODUL4\Guided\Guided1.go"
Masukan Nama Karyawan: Dinda
Masukan Gaji Pokok: 7000000
Masukan Jumlah Jam Lembur: 6

=== Slip Gaji ===
Nama Karyawan : Dinda
Gaji Pokok : Rp7000000.00
Bonus Lembur : Rp300000.00 (6 jam x Rp.50,000)
Total Gaji : Rp7300000.00
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4>
```

Deskripsi Program:

Program Guide1.go dengan bahasa Go lang dibuat untuk menghitung gaji karyawan berdasarkan gaji pokok dan jumlah jam lembur menggunakan prosedur hitungGaji. Perhitungan dilakukan dengan rumus Bonus Lembur = Jam Lembur × 50.000 dan Total Gaji = Gaji Pokok + Bonus Lembur. Pengguna diminta memasukkan nama, gaji pokok, dan jumlah jam lembur, lalu program menampilkan slip gaji berisi rincian gaji pokok, bonus lembur, dan total gaji yang diterima.

2. Guide 2

Source Code:

```
// Anastasia Adinda N.I
// Prosedur Kelulusan

package main

import "fmt"

func hitungRataRata(nama string, nilai1, nilai2, nilai3 float64) {

    ratarata := (nilai1 + nilai2 + nilai3) / 3
    status := "Tidak Lulus"

    if ratarata >= 60 {
        status = "Lulus"
    }

    fmt.Println("\n=== Hasil Akademik ===")
    fmt.Println("Nama Mahasiswa :", nama)
    fmt.Printf("Nilai 1     : %.2f\n", nilai1)
    fmt.Printf("Nilai 2     : %.2f\n", nilai2)
    fmt.Printf("Nilai 3     : %.2f\n", nilai3)
    fmt.Printf("Rata-rata   : %.2f\n", ratarata)
    fmt.Println("Status     :", status)
}

func main() {
```

```
    var nama string
    var nilai1, nilai2, nilai3 float64

    fmt.Print("Masukkan Nama Mahasiswa: ")
    fmt.Scanln(&nama)

    fmt.Print("Masukkan Nilai 1: ")
    fmt.Scanln(&nilai1)

    fmt.Print("Masukkan Nilai 2: ")
    fmt.Scanln(&nilai2)

    fmt.Print("Masukkan Nilai 3: ")
    fmt.Scanln(&nilai3)

    hitungRataRata(nama, nilai1, nilai2, nilai3)
}
```

Output:

```
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4> go run "d:\Semester 2\ALPRO2 CODING\103112400085_MODUL4\Guided\Guided2.go"
Masukkan Nama Mahasiswa: Dinda
Masukkan Nilai 1: 95
Masukkan Nilai 2: 90
Masukkan Nilai 3: 90

=== Hasil Akademik ===
Nama Mahasiswa : Dinda
Nilai 1        : 95.00
Nilai 2        : 90.00
Nilai 3        : 90.00
Rata-rata      : 91.67
Status         : Lulus
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4>
```

Deskripsi Program:

Program Guide2.go dengan Bahasa Go dibuat dengan tujuan untuk menghitung rata-rata nilai mahasiswa dan menentukan status kelulusannya menggunakan prosedur hitungRataRata. Perhitungan rata-rata dilakukan dengan rumus Rata-rata = (Nilai 1 + Nilai 2 + Nilai 3) / 3. Jika nilai rata-rata mencapai 60 atau lebih, mahasiswa dinyatakan "Lulus", sedangkan jika kurang dari 60, statusnya "Tidak Lulus". Pengguna diminta memasukkan nama dan tiga nilai, kemudian program menampilkan hasil akademik berupa nilai-nilai yang dimasukkan, rata-rata, serta status kelulusan mahasiswa.

3. **Guide 3.1**

Source Code:

```
//Anastasia Adinda
//Iterasi

package main

import "fmt"

func pangkatIteratif(base, exp int) int {
   hasil := 1

   for i := 0; i < exp; i++ {
      hasil *= base

   }

   return hasil
}

func faktorialIteratif(n int) int {
   hasil := 1
   for i := 2; i <= n; i++ {
      hasil *= i
   }
   return hasil
}

func main() {
   var base, exp, n int

   fmt.Print("Masukan Bilangan: ")
   fmt.Scanln(&base)
   fmt.Print("Masukan Pangkat: ")
   fmt.Scanln(&exp)

   fmt.Printf("%d^%d = %d\n", base, exp, pangkatIteratif(base, exp))

   fmt.Print("Masukan angka untuk faktorial: ")
   fmt.Scanln(&n)

   fmt.Printf("%d! = %d\n", n, faktorialIteratif(n))
}
```

Output:

```
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4> go run "d:\Semester 2\ALPRO2 CODING\103112400085_MODUL4\Guided\Guided3-1.go"
Masukan Bilangan: 5
Masukan Pangkat: 2
5^2 = 25
Masukan angka untuk faktorial: 7
7! = 5040
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4>
```

Deskripsi Program:

Program Guide3-1.go dengan bahasa Go lang digunakan untuk menghitung perpangkatan dan faktorial suatu bilangan menggunakan metode iteratif. Fungsi pangkatIteratif menghitung hasil perpangkatan dengan rumus hasil = base^exp, di mana base dikalikan dengan dirinya sendiri sebanyak exp kali menggunakan perulangan. Sedangkan fungsi faktorialIteratif menghitung faktorial dengan rumus n! = n × (n-1) × (n-2) × ... × 1 melalui perulangan dari 2 hingga n. Pengguna diminta memasukkan bilangan dan pangkatnya untuk perhitungan perpangkatan, serta bilangan untuk perhitungan faktorial, kemudian program menampilkan hasilnya.

4. Guide 3.2

Source Code:

```
//Anastasia Adinda
//Rekursif

package main

import "fmt"

func pangkatRekursif(base, exp int) int {
    if exp == 0 {
        return 1
    }
    return base * pangkatRekursif(base, exp-1)
}

func faktorialRekrusif(n int) int {
    if n == 0 || n == 1 {
        return 1
    }
    return n * faktorialRekrusif(n-1)
}

func main() {
    var base, exp, n int

    fmt.Print("Masukan Bilangan: ")
    fmt.Scanln(&base)
    fmt.Print("Masukan Pangkat: ")
```

```
    fmt.Scanln(&exp)

    fmt.Printf("%d^%d = %d\n", base, exp, pangkatRekursif(base, exp))

    fmt.Print("Masukan angka untuk faktorial: ")
    fmt.Scanln(&n)

    fmt.Printf("%d! = %d\n", n, faktorialRekrusif(n))
}
```

Output:

```
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4> go run "d:\Semester 2\ALPRO2 CODING\103112400085_MODUL4\Guided\Guided3-1.go"
Masukan Bilangan: 5
Masukan Pangkat: 2
5^2 = 25
Masukan angka untuk faktorial: 7
7! = 5040
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4>
```

Deskripsi Program:

Program Guide3-2.go dengan bahasa Go lang digunakan untuk menghitung perpangkatan dan faktorial suatu bilangan menggunakan metode rekursif. Fungsi pangkatRekursif menghitung hasil perpangkatan dengan rumus base^exp = base × base^(exp-1), di mana proses pemanggilan fungsi terus berlanjut hingga exp mencapai 0. Fungsi faktorialRekursif menghitung faktorial dengan rumus n! = n × (n-1)!, yang akan terus memanggil dirinya sendiri hingga nilai n mencapai 1 atau 0. Pengguna diminta memasukkan bilangan dan pangkatnya untuk perhitungan perpangkatan, serta bilangan untuk perhitungan faktorial, kemudian program menampilkan hasilnya.

III. UNGUIDED

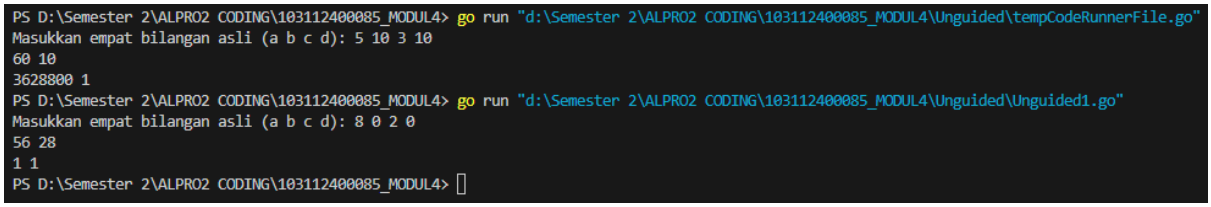
1. Unguided 1

Source Code:

```
//Anastasia Adinda
package main

import (
    "fmt"
)

func faktorial(n int, hasil *int) {
    *hasil = 1
    for i := 2; i <= n; i++ {
        *hasil *= i
    }
}

func permutasi(n, r int, hasil *int) {
    var fn, fnr int
    faktorial(n, &fn)
    faktorial(n-r, &fnr)
    *hasil = fn / fnr
}

func kombinasi(n, r int, hasil *int) {
    var fn, fr, fnr int
    faktorial(n, &fn)
    faktorial(r, &fr)
    faktorial(n-r, &fnr)
    *hasil = fn / (fr * fnr)
}

func main() {
    var a, b, c, d int
    var hasilP1, hasilC1, hasilP2, hasilC2 int

    fmt.Print("Masukkan empat bilangan asli (a b c d): ")
    fmt.Scan(&a, &b, &c, &d)

    if a >= c && b >= d {
        permutasi(a, c, &hasilP1)
        kombinasi(a, c, &hasilC1)
        permutasi(b, d, &hasilP2)
        kombinasi(b, d, &hasilC2)
        fmt.Printf("%d %d\n", hasilP1, hasilC1)
        fmt.Printf("%d %d\n", hasilP2, hasilC2)
    } else {
```

```
        fmt.Println("Input tidak memenuhi syarat a >= c dan b >= d")
    }
}
```

Output:



Deskripsi Program:

Program Unguided1.go dengan bahasa Go lang digunakan untuk menghitung nilai permutasi dan kombinasi dari dua pasang bilangan menggunakan prosedur. Fungsi faktorial digunakan untuk menghitung faktorial suatu bilangan, yang menjadi dasar perhitungan permutasi dan kombinasi. Rumus yang digunakan adalah P(n, r) = n! / (n-r)! untuk permutasi dan C(n, r) = n! / (r!(n-r)!) untuk kombinasi. Pengguna memasukkan empat bilangan asli (a, b, c, d) dengan syarat a ≥ c dan b ≥ d. Program kemudian menghitung dan mencetak hasil permutasi serta kombinasi untuk kedua pasangan bilangan yang diberikan. Jika syarat tidak terpenuhi, program akan menampilkan pesan kesalahan.

2. Unguided 2

Source Code:

```
// Anastasia Adinda
package main

import (
    "fmt"
)

func hitungSkor(waktu []int, jumlahSoal *int, totalWaktu *int) {
    batasWaktu := 301
    *jumlahSoal, *totalWaktu = 0, 0
    for _, t := range waktu {
        if t < batasWaktu {
            *jumlahSoal++
            *totalWaktu += t
        }
    }
}

func main() {
    var pemenang string
    var maxSoal, minWaktu int = 0, 1000000
```

```
    for {
        var nama string
        var waktu [8]int
        _, err := fmt.Scan(&nama)
        if err != nil || nama == "Selesai" {
            break
        }

        for i := 0; i < 8; i++ {
            _, err = fmt.Scan(&waktu[i])
            if err != nil {
                break
            }
        }

        var jumlahSoal, totalWaktu int
        hitungSkor(waktu[:], &jumlahSoal, &totalWaktu)

        if jumlahSoal > maxSoal || (jumlahSoal == maxSoal && totalWaktu <
minWaktu) {
            maxSoal = jumlahSoal
            minWaktu = totalWaktu
            pemenang = nama
        }
    }

    fmt.Println(pemenang, maxSoal, minWaktu)
}
```

Output:

```
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4> go run "d:\Semester 2\ALPRO2 CODING\103112400085_MODUL4\Unguided\Unguided2.go"
Astuti 20 50 301 301 61 71 75 10
Bertha 25 47 301 20 50 60 65 21
Selesai
Bertha 7 288
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4>
```

Deskripsi Program:

Program Unguided2.go dengan bahasa Go lang dibuat menentukan pemenang kompetisi berdasarkan jumlah soal yang diselesaikan dan total waktu pengerjaan. Soal yang tidak selesai dianggap memiliki waktu 301 menit. Pemenang adalah peserta dengan soal terbanyak, atau jika sama, dengan waktu paling sedikit. Rumus yang digunakan adalah S = jumlah soal dengan t < 301 dan T = sigma t untuk soal dengan t < 301.

3. Unguided 3

Source Code:

```
//Anastasia Adinda
package main

import (
    "fmt"
)

func cetakDeret(n int) {
    for n != 1 {
        fmt.Print(n, " ")
        if n%2 == 0 {
            n /= 2
        } else {
            n = 3*n + 1
        }
    }
    fmt.Println(n)
}

func main() {
    var n int
    fmt.Print("Masukkan bilangan positif (<1000000): ")
    fmt.Scan(&n)
    if n > 0 && n < 1000000 {
        cetakDeret(n)
    } else {
        fmt.Println("Bilangan tidak valid.")
    }
}
```

Output:

```
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4> go run "d:\Semester 2\ALPRO2 CODING\103112400085_MODUL4\Unguided\Unguided3.go"
Masukkan bilangan positif (<1000000): 22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS D:\Semester 2\ALPRO2 CODING\103112400085_MODUL4>
```

Deskripsi Program:

Program Unguided3.go dengan bahasa Go dibuat untuk mencetak deret Skiena berdasarkan aturan: jika bilangan nnn genap, maka suku berikutnya adalah n/2 sedangkan jika ganjil, suku berikutnya adalah 3n+1. Deret akan terus berlanjut hingga mencapai nilai 1. Program menerima satu bilangan positif kurang dari 1.000.000 sebagai input dan mencetak deret dalam satu baris dengan spasi sebagai pemisah.

IV. KESIMPULAN

- **Guided**

1. **Perhitungan Gaji**
    - Menghitung total gaji berdasarkan gaji pokok dan bonus lembur dengan rumus Bonus Lembur = Jam Lembur × 50.000 dan Total Gaji = Gaji Pokok + Bonus
    - Program meminta input nama, gaji pokok, dan jumlah jam lembur, lalu menampilkan slip gaji dengan rincian perhitungan.
2. **Perhitungan Rata-rata nilai**
    - Menghitung rata-rata nilai mahasiswa menggunakan rumus Rata-rata = (Nilai 1 + Nilai 2 + Nilai 3) / 3.
    - Jika nilai rata-rata ≥ 60, mahasiswa dinyatakan Lulus, sedangkan jika < 60, dinyatakan Tidak Lulus.
3. **Perpangkatan dan Fakorial**
    - Menghitung perpangkatan dengan rumus base^exp dan faktorial dengan rumus n! = n × (n-1) × (n-2) × ... × 1.
    - Program dibuat dalam dua metode, yaitu iteratif dan rekursif, untuk memahami konsep perulangan dan rekursi.

- **Unguided**

1. **Perhitungan Permutasi dan Kombinasi**
    - Menghitung permutasi dengan rumus P(n, r) = n! / (n-r)! dan kombinasi dengan C(n, r) = n! / (r!(n-r)!).
    - Perhitungan hanya dilakukan jika input memenuhi syarat n ≥ r.
2. **Kompetisi Pemrograman**
    - Menentukan pemenang kompetisi berdasarkan jumlah soal yang diselesaikan dan total waktu pengerjaan.
    - Soal yang tidak selesai dianggap memiliki waktu 301 menit, dan pemenang adalah peserta dengan soal terbanyak atau waktu pengerjaan tersingkat.
3. **Deret Skiena**
    - Mencetak deret bilangan dengan aturan: jika bilangan n genap, maka suku berikutnya adalah n/2, sedangkan jika ganjil, suku berikutnya adalah 3n + 1.
    - Program berjalan hingga bilangan mencapai 1 dan mencetak deret dalam satu baris dengan spasi sebagai pemisah.

V. REFRENSI

https://sko.dev/wiki/input-dan-output

https://codingstudio.id/blog/golang-adalah/

https://dif.telkomuniversity.ac.id/tipe-data-pemrograman/

https://rpubs.com/maulidyarahmah/829044

https://repository.unikom.ac.id/62967/1/Materi%20Pertemuan%204_Labview%201%2BWhile%20Loop%20%2B%20Shift%20Register.pdf

https://www.revou.co/kosakata/function

https://sko.dev/wiki/fungsi

https://drive.google.com/file/d/1tMnxOLAYoMqLB9cKyeJHwyFjmyVtKqMV/view?usp=sharing