LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 4

MATERI

Oleh:

MUHAMMAD ZAKY MUBAROK

103112400073

KELAS

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

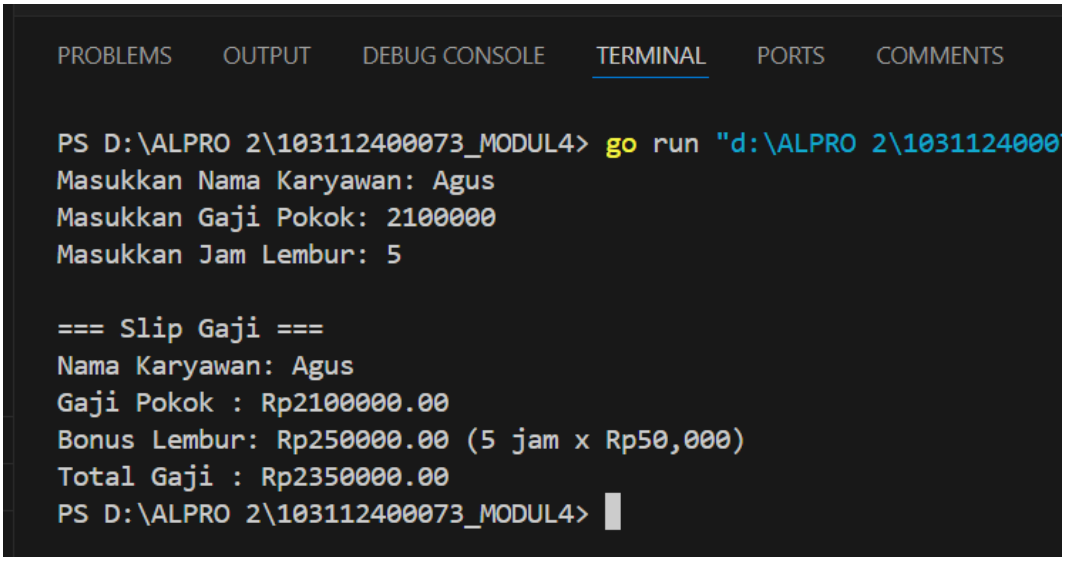
2025

I. DASAR TEORI

MODUL 4. PROSEDUR

II. GUIDED

```go
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 Rp50,000)\n",
bonusLembur, jamLembur)
    fmt.Printf("Total Gaji : Rp%.2f\n", totalGaji)
}

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

    fmt.Print("Masukkan Nama Karyawan: ")
    fmt.Scanln(&nama)
    fmt.Print("Masukkan Gaji Pokok: ")
    fmt.Scanln(&gajiPokok)
    fmt.Print("Masukkan Jam Lembur: ")
    fmt.Scanln(&jamLembur)

    hitungGaji(nama, gajiPokok, jamLembur)
}
```



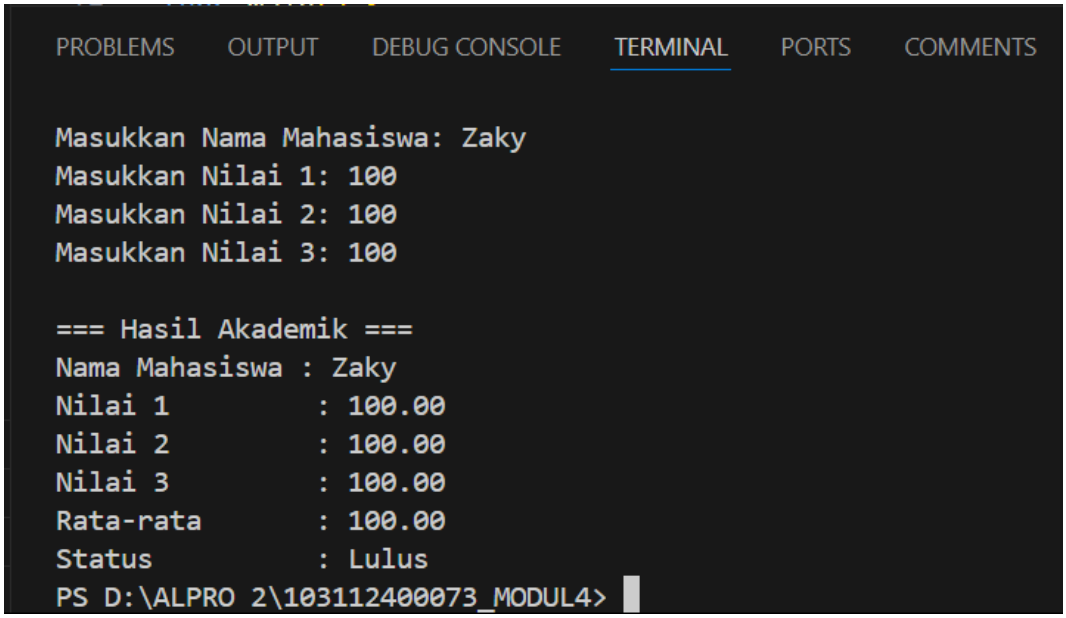
Penjelasan :

Kode program di atas adalah program Go yang menghitung gaji total seorang karyawan berdasarkan gaji pokok dan jam lembur. Berikut adalah penjelasan singkat:

1. **Fungsi** hitungGaji:
    - Menghitung bonus lembur berdasarkan jumlah jam lembur dan tarif Rp50.000 per jam.
    - Menambahkan bonus lembur ke gaji pokok untuk mendapatkan total gaji.
    - Menampilkan slip gaji yang berisi nama karyawan, gaji pokok, bonus lembur, dan total gaji.
2. **Fungsi** main:
    - Meminta input dari pengguna berupa nama karyawan, gaji pokok, dan jumlah jam lembur.
    - Memanggil fungsi hitungGaji dengan data input tersebut untuk menghitung dan menampilkan gaji.

III. GUIDED

```go
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)
}
```



Penjelasan :

Kode ini adalah program Go sederhana untuk menghitung rata-rata nilai tiga mata pelajaran dan menentukan status kelulusan seorang mahasiswa. Berikut penjelasannya:

1. **Fungsi hitungRataRata:**
    - **Mengambil nama mahasiswa dan tiga nilai sebagai parameter.**
    - **Menghitung rata-rata dari tiga nilai.**
    - **Menentukan status "Lulus" jika rata-rata ≥ 60, atau "Tidak Lulus" jika kurang dari 60.**
    - **Menampilkan hasil berupa nama mahasiswa, nilai-nilai yang dimasukkan, rata-rata, dan status kelulusan.**
2. **Fungsi main:**
    - **Meminta input dari pengguna (nama mahasiswa dan nilai-nilai).**
    - **Memanggil fungsi hitungRataRata untuk melakukan perhitungan dan menampilkan hasil.**

IV. UNGUIDED I

```go
//Muhammad Zaky Mubarok
package main

import "fmt"


func hitungFaktorial(n int, hasil *int) {
    if n == 0 || n == 1 {
        *hasil = 1
        return
    }

    var temp int
    hitungFaktorial(n-1, &temp)
    *hasil = n * temp
}


func hitungPermutasi(n, r int, hasil *int) {
    var faktorialN, faktorialNR int
    hitungFaktorial(n, &faktorialN)
    hitungFaktorial(n-r, &faktorialNR)
    *hasil = faktorialN / faktorialNR
}


func hitungKombinasi(n, r int, hasil *int) {
    var faktorialN, faktorialR, faktorialNR int
    hitungFaktorial(n, &faktorialN)
    hitungFaktorial(r, &faktorialR)
    hitungFaktorial(n-r, &faktorialNR)
    *hasil = faktorialN / (faktorialR * faktorialNR)
}

func main() {
    var a, b, c, d int
    fmt.Scanln(&a, &b, &c, &d)

    if a >= c && b >= d {

        var p1, c1, p2, c2 int
        hitungPermutasi(a, c, &p1)
```

```
        hitungKombinasi(a, c, &c1)
        hitungPermutasi(b, d, &p2)
        hitungKombinasi(b, d, &c2)


        fmt.Printf("%d %d\n", p1, c1)
        fmt.Printf("%d %d\n", p2, c2)
    } else {
        fmt.Println("Input tidak valid: a harus lebih besar atau
sama dengan c, dan b harus lebih besar atau sama dengan d.")
    }
}
```

```
PS D:\ALPRO 2\103112400073_MODUL4> go ru
5 10 3 10
60 10
3628800 1
PS D:\ALPRO 2\103112400073_MODUL4> go ru
8 0 2 0
56 28
1 1
PS D:\ALPRO 2\103112400073_MODUL4>
```

Penjelasan :

1. **Fungsi** hitungFaktorial**:**
    - Fungsi ini menghitung hasil perkalian semua bilangan bulat dari 1 hingga angka tertentu (n) secara rekursif.
    - Bila n adalah 0 atau 1, hasilnya adalah 1. Jika tidak, fungsi akan memanggil dirinya sendiri dengan nilai n-1 hingga selesai.
2. **Fungsi** hitungPermutasi**:**
    - Fungsi ini menggunakan hasil faktorial untuk menghitung jumlah kemungkinan urutan dari sejumlah elemen yang diambil sebagian.

Nilai ini dihitung dengan menghitung faktorial pada beberapa langkah tertentu.

3. **Fungsi** hitungKombinasi**:**
    - Fungsi ini juga bergantung pada hasil faktorial untuk menghitung berapa banyak cara untuk memilih sejumlah elemen tanpa memperhatikan urutannya. Prosesnya melibatkan beberapa panggilan ke fungsi faktorial.
4. **Fungsi** main**:**
    - Pengguna memasukkan nilai a, b, c, dan d.
    - Program mengecek apakah a lebih besar atau sama dengan c dan b lebih besar atau sama dengan d. Jika valid, program menghitung hasil permutasi dan kombinasi untuk dua set input (a, c dan b, d).
    - Hasil perhitungan kemudian ditampilkan. Jika input tidak valid, program memberikan pesan kesalahan.

Program ini memanfaatkan rekursi untuk menghitung faktorial dan pointer (*) untuk menyimpan hasil kalkulasi di dalam fungsi, sehingga nilai akhir dapat digunakan di tempat lain.

V. UNGUIDED II

```go
//Muhammad Zaky Mubarok
package main

import "fmt"


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


func cariPemenang(peserta map[string][]int, pemenang *string,
maxSoal *int, minSkor *int) {
    *pemenang = ""
    *maxSoal = 0
    *minSkor = 9999999

    for nama, waktu := range peserta {
        var soal, skor int
        hitungSkor(waktu, &soal, &skor)
        if soal > *maxSoal || (soal == *maxSoal && skor < *minSkor)
{
            *pemenang = nama
            *maxSoal = soal
            *minSkor = skor
        }
    }
}

func main() {

    peserta := make(map[string][]int)


    fmt.Println("Format: Nama Waktu")
```

```
    fmt.Println("Ketik 'Selesai' untuk mengakhiri input. (S nya
kapital ya)")


    for {
        var nama string
        var waktu [8]int
        fmt.Print("Input: ")

        _, err := fmt.Scan(&nama)
        if err != nil {
            fmt.Println("Input error.")
            break
        }

        if nama == "Selesai" {
            break
        }

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

        peserta[nama] = waktu[:]
    }


    var pemenang string
    var jumlahSoal, totalSkor int
    cariPemenang(peserta, &pemenang, &jumlahSoal, &totalSkor)


    fmt.Printf("\n%s %d %d\n", pemenang, jumlahSoal, totalSkor)
}
```

```
PS D:\ALPRO 2\103112400073_MODUL4> go run "d:\ALPRO 2\103112
Format: Nama Waktu
Ketik 'Selesai' untuk mengakhiri input. (S nya kapital ya)
Input: Astuti 20 50 301 301 61 71 75 10
Input: Bertha 25 47 301 26 50 60 65 21
Input: Selesai

Bertha 7 294
PS D:\ALPRO 2\103112400073_MODUL4>
```

Penjelasan :

- **Fungsi** `hitungSkor`:
    - Menghitung jumlah soal yang diselesaikan dalam waktu ≤ 300 detik dan total skor waktu dari array waktu yang diberikan.
- **Fungsi** `cariPemenang`:
    - Menentukan pemenang dengan membandingkan peserta berdasarkan:
        - Jumlah soal terbanyak yang diselesaikan.
        - Jika jumlah soal sama, peserta dengan total skor waktu terendah dipilih.
    - Hasil pemenang, jumlah soal, dan skor waktu disimpan melalui pointer.
- **Fungsi** `main`:
    - Meminta input dari pengguna untuk memasukkan nama peserta dan waktu penyelesaian untuk 8 soal.
    - Input dihentikan ketika pengguna mengetik "Selesai".
    - Setelah data selesai dimasukkan, fungsi `cariPemenang` dipanggil untuk menghitung hasil, dan nama pemenang beserta statistiknya ditampilkan.

VI. UNGUIDED III

```go
//Muhammad Zaky Mubarok
package main

import "fmt"


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

func main() {
    var n int

    fmt.Scanln(&n)

    cetakDeret(n)
}
```

```
PS D:\ALPRO 2\103112400073_MODUL4> go run "d:
22
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
PS D:\ALPRO 2\103112400073_MODUL4>
```

Penjelasan :

Fungsi cetakDeret(n int)
Fungsi ini menerima sebuah bilangan bulat positif n dan mencetak deret angka yang dihasilkan dari aturan Collatz.

- **Proses utama:**
  - **Selama n tidak sama dengan 1:**
    - **Cetak nilai n saat ini diikuti oleh spasi.**
    - **Jika n adalah bilangan genap (n % 2 == 0):**
      - **Bagi nilai n dengan 2 (n = n / 2).**
    - **Jika n adalah bilangan ganjil:**
      - **Kalikan nilai n dengan 3, lalu tambahkan 1 (n = 3*n + 1).**
  - **Ketika mencapai nilai 1, perulangan berhenti, dan angka 1 dicetak sebagai elemen terakhir dari deret.**
- **Contoh:**
  - **Jika n = 6, maka deret yang dihasilkan adalah: 6 3 10 5 16 8 4 2 1.**

2. Fungsi main()
Fungsi utama dalam program yang:

- **Menerima input pengguna berupa bilangan bulat positif n menggunakan fmt.Scanln(&n).**
- **Memanggil fungsi cetakDeret(n) untuk mencetak deret angka sesuai aturan.**

3. Alur Program

- **Program meminta pengguna memasukkan sebuah angka bulat positif.**
- **Jika pengguna memasukkan angka valid, fungsi cetakDeret akan menghasilkan dan mencetak deret angka berdasarkan aturan Collatz Conjecture:**
  - **Jika angka genap, bagi dua.**
  - **Jika angka ganjil, kalikan dengan 3 dan tambahkan 1.**
- **Program terus berulang hingga angka 1 tercapai.**

4. Aturan Collatz Conjecture

Collatz Conjecture menyatakan bahwa untuk bilangan bulat positif apa pun, jika aturan di atas diterapkan berulang kali, nilai akhirnya akan selalu mencapai 1. Hal ini belum terbukti untuk semua bilangan, tetapi sangat menarik secara matematis.

5. Contoh Eksekusi

Input: n = 7

Output: 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

Program ini memberikan pandangan menarik tentang algoritma sederhana dengan penggunaan loop dan kondisi

VII. KESIMPULAN

Kesimpulan dari Modul 4 - Praktikum Algoritma dan Pemrograman 2

Modul ini membahas konsep **prosedur** dalam pemrograman, khususnya dalam bahasa **Go** dan notasi **pseudocode**. Berikut adalah beberapa poin utama yang dapat disimpulkan dari modul ini:

1. **Definisi Prosedur**
   - Prosedur adalah subprogram yang terdiri dari sekumpulan instruksi yang membantu mengurangi kompleksitas program.
   - Tidak memiliki nilai yang dikembalikan seperti fungsi, dan tidak menggunakan kata kunci `return`.
2. **Deklarasi Prosedur**
   - Ditulis di luar `func main()` pada bahasa Go.
   - Contoh deklarasi dalam pseudocode dan Go diberikan untuk prosedur seperti **cetakNFibo** yang mencetak deret Fibonacci.
3. **Pemanggilan Prosedur**
   - Dilakukan dengan menyebutkan nama prosedur dan memberikan argumen yang diperlukan.
   - Bisa dipanggil langsung di dalam `main()` atau dari subprogram lain.
4. **Parameter dalam Prosedur**
   - **Parameter Formal**: Ditentukan saat deklarasi prosedur, berfungsi sebagai placeholder untuk argumen.
   - **Parameter Aktual**: Nilai nyata yang dikirimkan saat pemanggilan prosedur.
   - Dapat menggunakan metode **Pass by Value** (nilai dikopi) atau **Pass by Reference** (menggunakan pointer untuk referensi langsung ke memori).
5. **Contoh Implementasi dalam Program**
   - Membuat prosedur untuk mencetak pesan berdasarkan flag.
   - Menghitung **volume tabung** menggunakan parameter formal dan aktual.
   - Penggunaan **pointer (`*`)** pada bahasa Go untuk **Pass by Reference**.
6. **Latihan Pemrograman**
   - Implementasi prosedur untuk menghitung **faktorial, permutasi, dan kombinasi**.
   - Membuat program untuk menentukan **pemenang kompetisi** berdasarkan skor dan waktu.
   - Menampilkan **deret bilangan Skiena dan Revilla** menggunakan prosedur.

REFERENSI

MODUL 4. PROSEDUR