

**LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL III DAN IV
FUNGSI DAN PROSEDUR**



Oleh:

RAFI UBAID PUTRA

2311102244

IF-11-07

**S1 TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

2024

I. DASAR TEORI

Dasar Teori

Fungsi adalah blok kode yang dirancang untuk melakukan tugas tertentu dan mengembalikan nilai. Fungsi menerima parameter sebagai input dan mengembalikan hasil sebagai output. Dalam pemrograman, fungsi membantu dalam mengorganisir kode dan memecah masalah kompleks menjadi bagian-bagian yang lebih kecil dan lebih mudah dikelola. Contohnya, fungsi untuk menghitung volume tabung menerima jari-jari dan tinggi sebagai input, menghitung volume berdasarkan rumus matematis, dan mengembalikan nilai volume. Penggunaan fungsi tidak hanya membuat kode lebih terstruktur tetapi juga memudahkan pengujian dan pemeliharaan.

Prosedur, di sisi lain, adalah blok kode yang juga melakukan tugas tertentu tetapi tidak mengembalikan nilai. Prosedur biasanya digunakan untuk melakukan serangkaian perintah yang tidak memerlukan hasil yang perlu dikembalikan. Prosedur membantu mengorganisir kode dan memungkinkan pengulangan tugas tanpa harus menulis ulang kode yang sama. Misalnya, prosedur dapat digunakan untuk mencetak hasil perhitungan atau melakukan inisialisasi pada awal program. Dengan memisahkan logika program ke dalam fungsi dan prosedur, pengembang dapat membuat kode yang lebih bersih, mudah dipahami, dan lebih mudah untuk dimodifikasi di masa depan.

II. GUIDED

1. Source Code

package main

```
package main

import (
    "fmt"
)

func volumeTabung(jariJari, tinggi int) float64 {
    var luasAlas, volume float64

    luasAlas = 3.14 * float64(jariJari) *
float64(jariJari)

    volume = luasAlas * float64(tinggi)

    return volume
}

func main() {
    var r, t int

    var v1, v2 float64

    r = 5
    t = 10

    v1 = volumeTabung(r, t)
```

```

        v2 = volumeTabung(r, t) + volumeTabung(15, t)

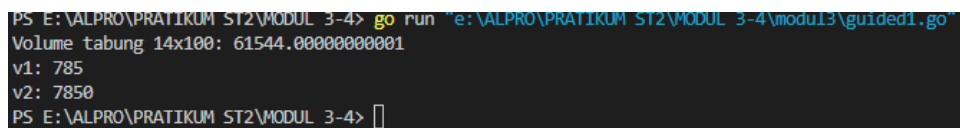
        fmt.Println("Volume tabung 14x100:",
volumeTabung(14, 100))

        fmt.Println("v1:", v1)

        fmt.Println("v2:", v2)
    }

```

Screenshot hasil program



```

PS E:\ALPRO\PRATIUM ST2\MODUL 3-4> go run "e:\ALPRO\PRATIUM ST2\MODUL 3-4\modul3\guided1.go"
Volume tabung 14x100: 61544.00000000001
v1: 785
v2: 7850
PS E:\ALPRO\PRATIUM ST2\MODUL 3-4>

```

Penjelasan program

Import dan Fungsi volumeTabung: Program dimulai dengan mengimpor paket `fmt` dan mendefinisikan fungsi `volumeTabung` untuk menghitung volume tabung berdasarkan jari-jari dan tinggi yang diberikan.

Menghitung Volume: Dalam fungsi `volumeTabung`, variabel `luasAlas` dan `volume` dideklarasikan sebagai `float64`. Luas alas dihitung menggunakan nilai

π

π yang diestimasi sebagai 3.14, dan hasil perhitungan volume dikembalikan.

Fungsi main: Fungsi `main` merupakan titik masuk program, di mana jari-jari (`r`) dan tinggi (`t`) dideklarasikan dan diinisialisasi. Volume tabung dihitung dengan memanggil `volumeTabung`, dan hasilnya disimpan dalam variabel `v1` dan `v2`. Akhirnya, hasil volume ditampilkan di konsol menggunakan `fmt.Println()`.

III. UNGUIDED

1. Source Code

package main

```
package main

import (
    "fmt"
)

func factorial(n int) int {
    if n == 0 {
        return 1
    }
    result := 1
    for i := 1; i <= n; i++ {
        result *= i
    }
    return result
}

func permutation(n, r int) int {
    if n < r {
        return 0
    }
}
```

```
        return factorial(n) / factorial(n-r)
    }

func combination(n, r int) int {
    if n < r {
        return 0
    }

    return factorial(n) / (factorial(r) *
factorial(n-r))
}

func main() {
    var a, b, c, d int

    fmt.Scan(&a, &b, &c, &d)

    p1 := permutation(a, c)
    c1 := combination(a, c)
    fmt.Printf("%d %d\n", p1, c1)

    p2 := permutation(b, d)
    c2 := combination(b, d)
    fmt.Printf("%d %d\n", p2, c2)
}
```

Screenshot hasil program

```
PS E:\ALPRO\PRATIUM ST2\MODUL 3-4\modul3> go run "e:\ALPRO\PRATIUM ST2\MODUL 3-4\modul3\unguided1.go"
5 16 3 18
60 10
0 0
PS E:\ALPRO\PRATIUM ST2\MODUL 3-4\modul3> █
```

Penjelasan program

Input Data: Program meminta 4 angka, misalnya a, b, c, d.

Fungsi Faktorial: Digunakan untuk menghitung hasil perkalian beruntun dari sebuah angka, contohnya $5! = 5 * 4 * 3 * 2 * 1 = 120$.

Permutasi: Menghitung cara mengatur sejumlah objek, menggunakan angka a dan c, serta b dan d.

Kombinasi: Menghitung cara memilih objek tanpa memperhatikan urutannya.

Hasil: Program menampilkan dua baris output, satu untuk pasangan a dan c, satu lagi untuk pasangan b dan d.

2. Source Code

```
IV. package main
V.
VI. import (
VII.     "fmt"
VIII. )
IX.
X. func f(x int) int {
XI.     return x * x
XII. }
XIII.
XIV. func g(x int) int {
```



```

XV.         return x - 2
XVI.     }
XVII.
XVIII.     func h(x int) int {
XIX.         return x + 1
XX.     }
XXI.
XXII.     func fogoh(x int) int {
XXIII.         return f(g(h(x)))
XXIV.     }
XXV.
XXVI.     func gohof(x int) int {
XXVII.         return g(h(f(x)))
XXVIII.     }
XXIX.
XXX.     func hofog(x int) int {
XXXI.         return h(f(g(x)))
XXXII.     }
XXXIII.
XXXIV.     func main() {
XXXV.         var a, b, c int
XXXVI.
XXXVII.         fmt.Scan(&a, &b, &c)
XXXVIII.
XXXIX.         fmt.Println(fogoh(a))
XL.
XLI.         fmt.Println(gohof(b))
XLII.
XLIII.        fmt.Println(hofog(c))
XLIV.    }
XLV.

```

Hasil Program

```

PS E:\ALPRO\PRATIUM ST2\MODUL 3-4\modul3> go run "e:\ALPRO\PRATIUM ST2\MODUL 3-4\modul3\unguided2.go"
7 2 10
36
3
65
PS E:\ALPRO\PRATIUM ST2\MODUL 3-4\modul3> 

```

Penjelasan Program

- Fungsi f(x): Mengembalikan hasil x kuadrat ($x * x$).
- Fungsi g(x): Mengembalikan hasil $x - 2$.
- Fungsi h(x): Mengembalikan hasil $x + 1$.
- Fungsi komposisi:
- fogoh(x): Menghitung $f(g(h(x)))$.

- gohof(x): Menghitung $g(h(f(x)))$.
- hofog(x): Menghitung $h(f(g(x)))$.
- Hasil:
- Baris pertama mencetak hasil komposisi $f(g(h(a)))$.
- Baris kedua mencetak hasil komposisi $g(h(f(b)))$.
- Baris ketiga mencetak hasil komposisi $h(f(g(c)))$.

3.Source Code

```
package main

import (
    "fmt"
    "math"
)

func jarak(a, b, c, d float64) float64 {
    return math.Sqrt((a-c)*(a-c) + (b-d)*(b-d))
}

func didalam(cx, cy, r, x, y float64) bool {
    return jarak(cx, cy, x, y) <= r
}

func main() {
    var cx1, cy1, r1, cx2, cy2, r2, x, y float64

    fmt.Scan(&cx1, &cy1, &r1)

    fmt.Scan(&cx2, &cy2, &r2)

    fmt.Scan(&x, &y)

    diDalamLingkaran1 := didalam(cx1, cy1, r1, x, y)
    diDalamLingkaran2 := didalam(cx2, cy2, r2, x, y)

    if diDalamLingkaran1 && diDalamLingkaran2 {
        fmt.Println("Titik di dalam lingkaran 1 dan 2")
    } else if diDalamLingkaran1 {
        fmt.Println("Titik di dalam lingkaran 1")
    } else if diDalamLingkaran2 {
        fmt.Println("Titik di dalam lingkaran 2")
    } else {
        fmt.Println("Titik di luar lingkaran 1 dan 2")
    }
}
```

```
}  
}
```

Hasil Program

```
PS E:\ALPRO\PRATIKUM ST2\MODUL 3-4\modul3> go run "e:\ALPRO\PRATIKUM ST2\MODUL 3-4\modul3\unguided3.go"  
5 8 2  
2 2 3  
5 8  
Titik di dalam lingkaran 1  
PS E:\ALPRO\PRATIKUM ST2\MODUL 3-4\modul3> |
```

Penjelasan Program

Fungsi jarak(a, b, c, d):

- Menghitung jarak antara dua titik (a, b) dan (c, d) menggunakan rumus Euclidean

Fungsi didalam(cx, cy, r, x, y):

- Menentukan apakah sebuah titik (x, y) berada di dalam lingkaran yang memiliki titik pusat (cx, cy) dan radius r.
- Titik dianggap berada di dalam lingkaran jika jaraknya dari pusat lingkaran kurang dari atau sama dengan radius.

Logika Program:

- Program meminta pengguna memasukkan koordinat pusat dan radius dari dua lingkaran, serta koordinat sebuah titik sembarang.
- Kemudian, program mengecek apakah titik tersebut berada di dalam lingkaran 1, lingkaran 2, atau di luar kedua lingkaran, dan menampilkan hasilnya sesuai kondisi.

MODUL IV

II. GUIDED

1.Source Code

package main

```
import (  
    "fmt"  
)  
  
func cetakNFibo(n int) {  
    var f1, f2, f3 int  
  
    f1 = 0  
  
    f2 = 1  
  
    for i := 1; i <= n; i++ {  
        fmt.Println(f3)  
  
        f3 = f1 + f2  
  
        f1 = f2  
  
        f2 = f3  
    }  
}  
  
func main() {  
    var x int  
  
    x = 5  
  
    cetakNFibo(x)
```

```
cetakNFibo(100)

}
```

Hasil Program

```
PS E:\ALPRO\PRATIUM ST2\MODUL 3-4> go run "e:\ALPRO\PRATIUM ST2\MODUL 3-4\guided1.go"
0
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
6765
10946
17711
28657
46368
75025
121393
196418
317811
514229
832040
1346269
2178309
3524578
5702887
9227465
14930352
24157817
39088169
63245986
102334155
165580141
267914296
433494437
701408733
1134903170
1836311903
2971215073
4807526976
7778742049
12586269025
20365011074
32951280099
```

Penjelasan Program

1. Program dimulai dengan mendeklarasikan paket utama package main, yang menunjukkan bahwa ini adalah titik masuk untuk program Go. Selanjutnya, paket fmt diimpor untuk memungkinkan penggunaan fungsi-fungsi yang terkait dengan format dan input/output, seperti fmt.Println(). Fungsi cetakNFibo didefinisikan untuk mencetak deret Fibonacci sebanyak n suku. Di dalam fungsi ini, tiga variabel integer f1, f2, dan f3 dideklarasikan dan diinisialisasi. f1 dimulai dengan 0, dan f2 dimulai dengan 1, yang merupakan dua suku pertama dari deret Fibonacci.
2. Fungsi cetakNFibo menggunakan loop for untuk iterasi dari 1 hingga n, di mana n adalah jumlah suku Fibonacci yang ingin dicetak. Di dalam loop, program mencetak nilai f3, yang awalnya 0. Setelah itu, f3 diupdate dengan jumlah dari f1 dan f2, yang merupakan cara untuk menghasilkan suku Fibonacci berikutnya. Setelah perhitungan, f1 diupdate menjadi nilai f2, dan f2 diupdate menjadi nilai f3. Proses ini diulang hingga semua suku yang diminta dicetak.
3. Fungsi main berfungsi sebagai titik masuk utama untuk menjalankan program. Di sini, sebuah variabel integer x dideklarasikan dan diinisialisasi dengan nilai 5. Fungsi cetakNFibo(x) dipanggil untuk mencetak 5 suku pertama dari deret Fibonacci. Setelah itu, fungsi yang sama dipanggil dengan argumen 100, yang berarti program akan mencetak 100 suku pertama dari deret Fibonacci. Dengan pemanggilan fungsi ini, pengguna akan melihat deret Fibonacci dicetak di konsol, dimulai dari 0 dan 1, diikuti oleh suku-suku berikutnya berdasarkan aturan deret Fibonacci.

III. UNGUIDED

1. Source Code

```
package main

import "fmt"

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

```
func permutation(n, r int) int {  
    return factorial(n) / factorial(n-r)  
}  
  
func combination(n, r int) int {  
    return factorial(n) / (factorial(r) * factorial(n-r))  
}  
  
func main() {  
    var a, b, c, d int  
  
    fmt.Scan(&a, &b, &c, &d)  
  
    perm_a_c := permutation(a, c)  
    comb_a_c := combination(a, c)  
  
    perm_b_d := permutation(b, d)  
    comb_b_d := combination(b, d)  
  
    fmt.Printf("%d %d\n", perm_a_c, comb_a_c)  
    fmt.Printf("%d %d\n", perm_b_d, comb_b_d)  
}
```

Hasil Program

```
PS E:\ALPRO\PRATIUM ST2\MODUL 3-4\modul3> go run "e:\ALPRO\PRATIUM ST2\MODUL 3-4\modul4\unguided1.go"
5 8 3 2
60 10
56 28
PS E:\ALPRO\PRATIUM ST2\MODUL 3-4\modul3> 
```

Penjelasan Program

- Permutasi $P(5, 3) = 60$, Kombinasi $C(5, 3) = 10$
- Permutasi $P(8, 2) = 56$, Kombinasi $C(8, 2) = 28$

2. Source Code

```
package main

import (
    "fmt"
    "math"
)

func hitungSkor(skor *int, soal *int, waktu []int)
{
    totalSoal := 0
    totalSkor := 0
    for _, w := range waktu {
        if w <= 300 {
            totalSoal++
            totalSkor += w
        }
    }
}
```



```
*soal = totalSoal

*skor = totalSkor
}

func main() {

    var namaPemenang string

    var soalPemenang, skorPemenang int

    skorPemenang = math.MaxInt32

    soalPemenang = -1


    for {

        var nama string

        var waktu [8]int


        fmt.Scan(&nama)

        if nama == "Selesai" {

            break

        }


        for i := 0; i < 8; i++ {

            fmt.Scan(&waktu[i])

        }


        var soal, skor int
```

```

        hitungSkor(&skor, &soal, waktu[:])

        if soal > soalPemenang || (soal ==
soalPemenang && skor < skorPemenang) {

            namaPemenang = nama

            soalPemenang = soal

            skorPemenang = skor

        }

    }

    fmt.Printf("%s %d %d\n", namaPemenang,
soalPemenang, skorPemenang)
}

```

Hasil Program

```

PS E:\ALPRO\PRATIUM ST2\MODUL 3-4> go run "e:\ALPRO\PRATIUM ST2\MODUL 3-4\modul4\unguided2.go"
Astuti 50 80 100 47 301 301 61 71
Bertha 10 26 50 60 65 21 301 301
Selesai
Bertha 6 232
PS E:\ALPRO\PRATIUM ST2\MODUL 3-4>

```

Penjelasan Program

1. Program ini dimulai dengan mengimpor paket yang diperlukan dan mendefinisikan fungsi utama main(), di mana nama peserta dan waktu pengerjaan soal akan dibaca. Prosedur hitungSkor digunakan untuk menghitung total soal yang diselesaikan dan total waktu yang dibutuhkan oleh setiap peserta. Dalam prosedur ini, setiap waktu yang lebih dari 301 menit (5 jam dan 1 menit) dianggap tidak valid, dan peserta tersebut tidak dihitung dalam skor. Peserta yang berhasil menyelesaikan soal dihitung berdasarkan waktu yang valid, dan

hasilnya disimpan dalam variabel yang akan digunakan untuk menentukan pemenang.

2. Setelah semua peserta diproses, program membandingkan jumlah soal yang diselesaikan oleh setiap peserta. Jika terdapat peserta yang menyelesaikan jumlah soal yang sama, pemenang ditentukan berdasarkan total waktu yang lebih rendah. Hasil akhir menampilkan nama pemenang, jumlah soal yang diselesaikan, dan total waktu yang digunakan. Dengan cara ini, program secara efisien menentukan pemenang berdasarkan kriteria yang ditetapkan.

3. Source Code

```
package main

import (
    "fmt"
)

func cetakDeret(n int) {
    for n != 1 {
        fmt.Print(n, " ")
        if n%2 == 0 {
            n = n / 4
        } else {
            n = 3*n + 1
        }
    }
    fmt.Print(1)
}
```

```

func main() {
    var n int

    fmt.Print("Masukkan bilangan bulat positif
(kurang dari 1000000): ")

    fmt.Scan(&n)

    if n > 0 && n < 1000000 {
        cetakDeret(n)
    } else {
        fmt.Println("Masukkan angka tidak valid.
Harap masukkan angka antara 1 dan 999999.")
    }
}

```

Hasil Program

```

PS E:\ALPRO\PRATIUM ST2\MODUL 3-4> go run "e:\ALPRO\PRATIUM ST2\MODUL 3-4\modul4\unguided3.go"
Masukkan bilangan bulat positif (kurang dari 1000000): 30
30 7 22 5 16 4 1
PS E:\ALPRO\PRATIUM ST2\MODUL 3-4>

```

Penjelasan Program

1. Penggunaan Fungsi dan Prosedur: Program ini menggunakan prosedur cetakDeret untuk mencetak deret bilangan berdasarkan nilai awal yang diberikan. Prosedur ini menggunakan loop untuk terus menghitung dan mencetak nilai n sesuai dengan aturan yang telah ditetapkan hingga mencapai nilai 1.
2. **Logika Deret:** Di dalam prosedur, terdapat kondisi untuk memeriksa apakah n genap atau ganjil. Jika n genap, nilai n akan dibagi 4, dan jika n ganjil, nilai n akan dihitung menggunakan rumus $3n + 1$. Setelah perhitungan, nilai n yang baru akan dicetak. Prosedur

berlanjut hingga n mencapai 1, di mana 1 juga akan dicetak sebagai suku terakhir.

3. **Validasi Input:** Di dalam fungsi main, ada validasi untuk memastikan bahwa nilai input berada dalam rentang yang ditentukan (1 hingga kurang dari 1.000.000). Jika nilai tidak valid, pesan kesalahan akan ditampilkan.