

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

MODUL 5

MATERI



Oleh:

ABISAR FATHIR

103112400068

12-IF-01

**S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO**

2025

I. DASAR TEORI

Dasar Teori - Modul 5: Rekursi

Rekursi adalah teknik pemrograman di mana sebuah fungsi memanggil dirinya sendiri untuk menyelesaikan suatu masalah dengan membaginya menjadi sub-masalah yang lebih kecil. Dalam rekursi, terdapat dua komponen utama, yaitu **base case** dan **recursive case**.

Base case adalah kondisi yang digunakan untuk menghentikan rekursi agar tidak berjalan tanpa batas. Contohnya dalam pencetakan angka secara rekursif, base case dapat berupa kondisi ketika angka telah mencapai batas tertentu.

```
}func cetak(x int) {  
    if x == 10 {  
        return // Rekursi berhenti saat x mencapai 10  
    }  
    fmt.Println(x)  
    cetak(x + 1) // Pemanggilan rekursif
```

Recursive case adalah bagian di mana fungsi memanggil dirinya sendiri untuk menyelesaikan bagian kecil dari masalah utama. Misalnya dalam perhitungan faktorial, fungsi akan terus memanggil dirinya sendiri dengan nilai yang lebih kecil hingga mencapai base case.

```
func faktorial(n int) int {  
    if n == 0 {  
        return 1 // Base case: 0! = 1
```

```
}  
  
return n * faktorial(n - 1) // Recursive case  
  
}
```

Rekursi memiliki beberapa keunggulan dibandingkan perulangan biasa. Teknik ini sering digunakan untuk menyelesaikan masalah yang memiliki pola pemecahan yang berulang, seperti pencarian dalam struktur data pohon, perhitungan deret Fibonacci, atau pencarian jalur dalam graf.

II. GUIDED

1. Program ke 1

Source Code:

```
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("masukkan bilangan : ")

fmt.Scanln(&base)

fmt.Print("masukkan pangkat: ")

fmt.Scanln(&exp)

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

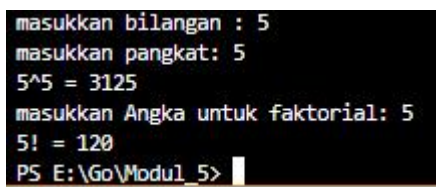
fmt.Print("masukkan Angka untuk faktorial: ")

fmt.Scanln(&n)

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

}
```

Output Program:



```
masukkan bilangan : 5
masukkan pangkat: 5
5^5 = 3125
masukkan Angka untuk faktorial: 5
5! = 120
PS E:\Go\Modul_5>
```

Deskripsi Program: Program ini menggunakan rekursi untuk menghitung pangkat dan faktorial dari suatu bilangan yang dimasukkan oleh pengguna. Program akan meminta dua input:

1. Bilangan (base) dan eksponen (exp) untuk menghitung pangkat menggunakan fungsi rekursif.
2. Bilangan (n) untuk menghitung faktorial menggunakan fungsi rekursif.

2. Program ke 2

Source Code:

```
package main

import "fmt"

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

func faktorialrekursif(n int) int {
    if n == 0 || n == 1 {
        return 1
    }
```

```

}

return n * faktorialrekursif(n-1)

}

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

fmt.Print("Masukkan bilangan: ")

fmt.Scanln(&base)

fmt.Print("masukkan pangkat: ")

fmt.Scanln(&exp)


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

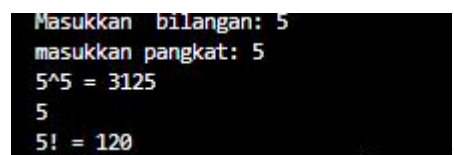
fmt.Scanln(&n)

fmt.Printf("%d! = %d\n", n, faktorialrekursif(n))

}

```

Output Program



```

Masukkan bilangan: 5
masukkan pangkat: 5
5^5 = 3125
5
5! = 120

```

Deskripsi Program: Program digunakan untuk menghitung pangkat dan faktorial menggunakan rekursi.

Pengguna akan memasukkan angka sebagai input, lalu program akan

menghitung pangkat dan faktorial dari angka tersebut, kemudian menampilkan hasilnya di layar.

3. Program ke 3

Source Code:

```
package main

import (
    "fmt"
    "math"
)

// Fungsi untuk menghitung luas permukaan tabung
func luasPermukaanTabung(r, t float64) float64 {
    return 2 * math.Pi * r * (r + t)
}

// Fungsi untuk menghitung volume tabung
func volumeTabung(r, t float64) float64 {
    return math.Pi * math.Pow(r, 2) * t
}

func main() {
    var r, t float64
```



```

// Input jari-jari dan tinggi tabung dengan validasi

fmt.Print("Masukkan jari-jari tabung: ")

_, errR := fmt.Scan(&r)

fmt.Print("Masukkan tinggi tabung: ")

_, errT := fmt.Scan(&t)


// Memeriksa apakah input valid

if errR != nil || errT != nil {

    fmt.Println("Input tidak valid! Harap masukkan angka yang
benar.")

    return

}


// Memeriksa apakah jari-jari dan tinggi bernilai positif

if r <= 0 || t <= 0 {

    fmt.Println("Jari-jari dan tinggi tabung harus lebih dari nol.")

    return

}


// Menghitung luas permukaan dan volume

luas := luasPermukaanTabung(r, t)

volume := volumeTabung(r, t)

```

```

        // Menampilkan hasil

        fmt.Println("=====")

        fmt.Printf("Luas Permukaan Tabung: %.2f satuan²\n", luas)

        fmt.Printf("Volume Tabung: %.2f satuan³\n", volume)

        fmt.Println("=====")
    }

```

Output Program:

```

Masukkan jari-jari tabung: 2
Masukkan tinggi tabung: 2
=====
Luas Permukaan Tabung: 50.27 satuan²
Volume Tabung: 25.13 satuan³
=====

```

Deskripsi Program: Program ini dibuat untuk menghitung pangkat dan faktorial menggunakan rekursi.

Pengguna akan memasukkan angka sebagai input, lalu program akan menghitung hasil pangkat dan faktorial menggunakan fungsi rekursif, kemudian menampilkan hasilnya di

III. UNGUIDED

1. Program ke 1

Source Code:

```
package main
```

```
import (
```

```
"fmt"

)

func fibonacci(n int) int {
    if n <= 1 {
        return n
    }
    return fibonacci(n-1) + fibonacci(n-2)
}

func main() {
    var jumlahSuku int
    fmt.Scan(&jumlahSuku)

    for i := 0; i < jumlahSuku; i++ {
        fmt.Printf("%d ", fibonacci(i))
    }
    fmt.Println()
}
```

Output Program:

```
10
0 1 1 2 3 5 8 13 21 34 55
```

Deskripsi Program:Program ini dibuat dan digunakan untuk menghitung serta mencetak deret Fibonacci menggunakan fungsi rekursif.

2. Program ke 2

Source Code:

```
package main

import "fmt"

func Star(n int, i int) {
    if i > n {
        return
    }

    for j := 0; j < i; j++ {
        fmt.Print("*")
    }
    fmt.Println()

    Star(n, i+1)
}
```

```
func main() {

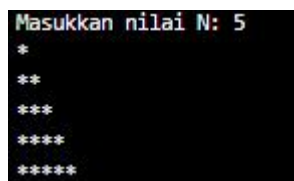
var N int

fmt.Scan(&N)


Star(N, 1)

}
```

Output Program:



```
Masukkan nilai N: 5
*
**
***
****
*****
```

Deskripsi Program: Program ini menggunakan rekursi untuk mencetak pola segitiga bintang berdasarkan input dari pengguna.

3. Program ke 3

Source Code:

```
package main

import (

"fmt"

)


func cetakDeret(n int) {
```

```
package main

import "fmt"

func faktorRekursif(N, i int) {
    if i > N {
        return
    }
    if N%i == 0 {
        fmt.Printf("%d ", i)
    }
    faktorRekursif(N, i+1)
}

func main() {
    var N int
    fmt.Scanln(&N)
    fmt.Print("Faktor: ")
    faktorRekursif(N, 1)
    fmt.Println()
}
```

Output Program:

```
5
Faktor: 1 5
```

Deskripsi Program: Program ini dibuat dalam Golang untuk menampilkan faktor bilangan dari sebuah bilangan N menggunakan rekursi.

IV. KESIMPULAN

Modul 5 Rekursi merupakan teknik pemrograman yang memungkinkan sebuah fungsi untuk memanggil dirinya sendiri guna menyelesaikan masalah yang lebih kecil hingga mencapai kondisi berhenti. Untuk memastikan rekursi tidak berjalan tanpa batas, diperlukan base case sebagai syarat berhenti.

Rekursi memiliki kelebihan dalam menyelesaikan masalah yang memiliki pola berulang, sehingga sering digunakan dalam pemrograman algoritma seperti faktorial, Fibonacci, dan eksplorasi struktur data pohon. Namun, rekursi juga memiliki kelemahan, yaitu penggunaan memori yang lebih besar dibandingkan iterasi karena menyimpan banyak pemanggilan fungsi dalam stack memori. Oleh karena itu, penggunaan rekursi harus dilakukan dengan mempertimbangkan efisiensi dan optimalisasi, seperti menggunakan teknik memoization atau tail recursion untuk menghindari penggunaan memori yang berlebihan.

V. REFERENSI