

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL V**

**REKURSIF**



Oleh:

**FEROS PEDROSA VALENTINO**

103112400055

IF-12-01

**S1 TEKNIK INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2025**

## **I. DASAR TEORI**

Rekursif secara sederhana dapat diartikan sebagai cara menyelesaikan suatu masalah dengan cara menyelesaikan sub-masalah yang identik dari masalah utama. Pengulangan Rekursif adalah teknik dalam pemrograman di mana suatu fungsi memanggil dirinya sendiri untuk menyelesaikan tugas yang lebih kecil hingga mencapai kondisi dasar yang menghentikan proses tersebut.

Biasanya ditambahkan struktur kontrol percabangan (if-then) untuk menghentikan proses rekursif ini. Kondisi ini disebut juga dengan base-case, artinya apabila kondisi base-case bernilai true maka proses rekursif akan berhenti. Base-case adalah kondisi proses rekursif berhenti. Base-case merupakan hal terpenting dan pertama yang harus diketahui ketika akan membuat program rekursif. Mustahil membuat program rekursif tanpa mengetahui base-case terlebih dahulu. Recursive-case adalah kondisi dimana proses pemanggilan dirinya sendiri dilakukan. Kondisi recursive-case adalah komplemen atau negasi dari base-case.

Teknik rekursif ini merupakan salah satu alternatif untuk mengganti struktur kontrol perulangan dengan memanfaatkan subprogram (bisa fungsi ataupun prosedur).

Algoritma rekursif terdiri dari dua komponen utama yaitu

- Base-case (Basis), yaitu bagian untuk menghentikan proses rekursif dan menjadi komponen terpenting di dalam sebuah rekursif.
- Recursive-case, yaitu bagian pemanggilan subprogramnya.

## II. GUIDED

### 1. Guided 1

Source code:

```
//Feros Pedrosa

package main

import "fmt"

// Fungsi iteratif untuk menghitung pangkat
func pangkatIteratif(base, exp int) int {
    hasil := 1
    for i := 0; i < exp; i++ {
        hasil *= base
    }
    return hasil
}

// Fungsi iteratif untuk menghitung faktorial
func faktorialIteratif(n int) int {
    hasil := 1
    for i := 2; i <= n; i++ {
        hasil *= i
    }
    return hasil
}

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

    //input pangkat
    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))

    //input faktorial
    fmt.Print("Masukkan angka untuk faktorial: ")
    fmt.Scanln(&n)

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

```
}
```

### Output:

```
PS C:\ALPRO\semester2_alpro2\alpro2_week4> go run "c:\ALPRO\semester2_alpro2\alpro2_week4\coso_iteratif-1\guided1.go"
Masukkan bilangan: 2
Masukkan pangkat: 3
2^3 = 8
Masukkan angka untuk faktorial: 4
4! = 24
```

### Deskripsi program:

Program diatas ditulis dalam bahasa Go dan berfungsi untuk menghitung pangkat dan faktorial menggunakan metode iteratif. Ada beberapa fungsi di program ini adalah fungsi pangkatIteratif yang menerima dua parameter yaitu base, exp lalu menghitung pemangkatan dengan melakukan perkalian berulang sebanyak nilai eksponen. dan fungsi faktorialIteratif yang menerima satu parameter yaitu n lalu menghitung faktorialnya dengan mengalikan bilangan dari 1 hingga n. Pada fungsi main deklarasikan variabel base, exp, n sebagai tipe data integer terlebih dahulu. Lalu program meminta untuk memasukkan bilangan dan memasukkan pangkat untuk operasi pangkat. Kemudian program menghitungnya dengan fungsi pangkatIteratif dan mencetaknya. Selanjutnya program meminta untuk memasukkan angka dan program akan menghitungnya menggunakan fungsi faktorialIteratif lalu mencetaknya.

## 2. Guided 2

Source code:

```
//Feros Pedrosa

package main

import "fmt"

// Fungsi rekursif untuk menghitung pangkat
func pangkatRekursif(base, exp int) int {
    if exp == 0 {
        return 1
    }
    return base * pangkatRekursif(base, exp-1)
}

// Fungsi rekursif untuk menghitung faktorial
func faktorialRekursif(n int) int {
    if n == 0 || n == 1 {
        return 1
    }
    return n * faktorialRekursif(n-1)
}

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

    // Input pangkat
    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))

    // Input faktorial
    fmt.Print("Masukkan angka untuk faktorial: ")
    fmt.Scan(&n)

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

## Output:

```
PS C:\ALPRO\semester2_alpro2\alpro2_week4> go run "c:\ALPRO\semester2_alpro2\alpro2_week4\coso_rekursif-2\guided2.go"
Masukkan bilangan: 3
Masukkan pangkat: 3
3^3 = 27
Masukkan angka untuk faktorial: 5
5! = 120
```

## Deskripsi program:

Program diatas ditulis dalam bahasa Go dan berfungsi untuk menghitung pangkat dan faktorial menggunakan metode rekursif. Program ini memiliki fungsi pangkatRekursif untuk menghitung hasil perpangkatan dengan mendasarkan perhitungannya pada kasus dasar ketika eksponen(exp) bernilai 0, yang mengembalikan 1, serta kasus rekursif yang mengalikan base dengan hasil pemanggilan fungsi itu sendiri dengan exp-1 dan fungsi faktorialRekursif untuk menghitung faktorial suatu bilangan dengan menggunakan prinsip rekursi, di mana kasus dasar terjadi saat n bernilai 0 atau 1, yang mengembalikan 1, sedangkan kasus rekursif mengalikan n dengan pemanggilan fungsi itu sendiri dengan parameter n-1. Pada fungsi main deklarasikan variabel base, exp, n sebagai tipe data integer. Lalu program meminta untuk memasukkan bilangan dan pangkat untuk dihitung menggunakan fungsi pangkatRekursif kemudian dicetak. Dan program juga meminta untuk memasukkan angka untuk faktorial untuk dihitung menggunakan fungsi faktorialRekursif kemudian dicetak.

### III. UNGUIDED

#### 1. Unguided 1

Source code:

```
//Feros Pedrosa

package main

import "fmt"

// Fungsi rekursif untuk menghitung nilai Fibonacci
func fibonacci(n int) int {
    if n == 0 {
        return 0
    } else if n == 1 {
        return 1
    }
    return fibonacci(n-1) + fibonacci(n-2)
}

func main() {
    var n int

    // Meminta pengguna untuk memasukkan nilai n
    fmt.Print("Masukkan suku Fibonacci yang diinginkan (n): ")
    fmt.Scan(&n)

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

Output:

```
PS C:\ALPRO\semester2_alpro2\alpro2_week4> go run "c:\ALPRO\semester2_alpro2\alpro2_week4\latsol1-modul5\unguided1.go"
Masukkan suku Fibonacci yang diinginkan (n): 10
Fibonacci(0) = 0
Fibonacci(1) = 1
Fibonacci(2) = 1
Fibonacci(3) = 2
Fibonacci(4) = 3
Fibonacci(5) = 5
Fibonacci(6) = 8
Fibonacci(7) = 13
Fibonacci(8) = 21
Fibonacci(9) = 34
Fibonacci(10) = 55
```

Deskripsi program:

Program diatas ditulis dalam bahasa Go dan berfungsi untuk menghitung dan menampilkan hasil perhitungan deret fibonacci menggunakan rekursif. Pertama definisikan fungsi fibonacci yang menggunakan

rekursif untuk menghitung nilai fibonacci berdasarkan rumus  $S_n = S(n-1) + S(n-2)$  dengan kondisi dasar suku ke nol adalah nol dan suku ke satu adalah satu. Pada fungsi main deklarasikan variabel n sebagai tipe data integer lalu program meminta untuk memasukkan suku fibonacci yang diinginkan (n). Program akan mencetak setiap suku Fibonacci mulai dari  $S(0)$  hingga Suku yang diinputkan oleh pengguna menggunakan perulangan for, dengan setiap iterasi memanggil fungsi fibonacci(i) untuk menghitung nilai Fibonacci ke-i. Hasil akhirnya ditampilkan dalam format Fibonacci(i) = nilai.



## 2. Unguided 2

Source code:

```
//Feros Pedrosa

package main

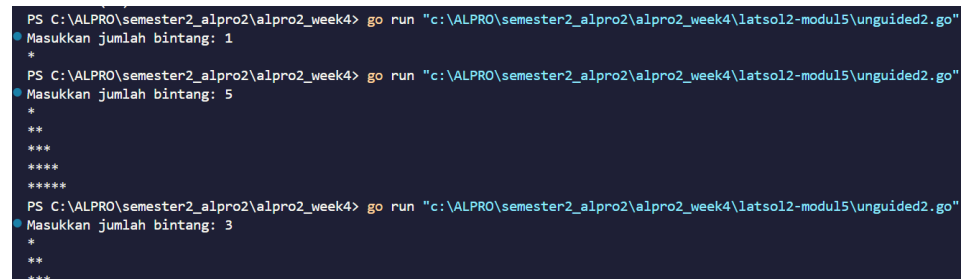
import "fmt"

func cetakBintang(n int, current int) {
    if current > n {
        return
    }
    for i := 0; i < current; i++ {
        fmt.Print("*")
    }
    fmt.Println()
    cetakBintang(n, current+1)
}

func main() {
    var n int
    fmt.Print("Masukkan jumlah bintang: ")
    fmt.Scan(&n)

    cetakBintang(n, 1)
}
```

Output:



```
PS C:\ALPRO\semester2_alpro2\alpro2_week4> go run "c:\ALPRO\semester2_alpro2\alpro2_week4\latsol2-modul5\unguided2.go"
Masukkan jumlah bintang: 1
*
PS C:\ALPRO\semester2_alpro2\alpro2_week4> go run "c:\ALPRO\semester2_alpro2\alpro2_week4\latsol2-modul5\unguided2.go"
Masukkan jumlah bintang: 5
*
**
***
****
*****
PS C:\ALPRO\semester2_alpro2\alpro2_week4> go run "c:\ALPRO\semester2_alpro2\alpro2_week4\latsol2-modul5\unguided2.go"
Masukkan jumlah bintang: 3
*
**
***
```

Deskripsi program:

Program diatas ditulis dalam bahasa Go dan berfungsi untuk mencetak pola bintang yang membentuk segitiga secara rekursif berdasarkan jumlah yang dimasukkan. Pertama deklarasikan fungsi cetakBintang untuk mencetak bintang sebanyak nilai current pada setiap baris dan memanggil dirinya sendiri secara rekursif dengan current+1 hingga mencapai nilai n. Jika current lebih besar dari n fungsi akan berhenti. Pada fungsi main deklarasikan variabel n sebagai tipe data integer lalu program meminta untuk memasukkan jumlah bintang. Setelah itu,

fungsi cetakBintang(n, 1) dipanggil untuk memulai pencetakan bintang dari baris pertama hingga baris ke-n secara bertahap.

### 3. Unguided 3

Source code:

```
//Feros Pedrosa

package main

import "fmt"

func faktorRecursive(n int, i int) {
    if i > n {
        return
    }
    if n%i == 0 {
        fmt.Print(i, " ")
    }
    faktorRecursive(n, i+1)
}

func main() {
    var N int

    // Menerima input dari user
    fmt.Print("Masukkan bilangan: ")
    fmt.Scan(&N)

    fmt.Printf("Faktor dari %d: ", N)
    faktorRecursive(N, 1)
}
```

Output:

```
PS C:\ALPRO\semester2_alpro2\alpro2_week4> go run "c:\ALPRO\semester2_alpro2\alpro2_week4\latsol3-modul5\unguided3.go"
Masukkan bilangan: 5
Faktor dari 5: 1 5
PS C:\ALPRO\semester2_alpro2\alpro2_week4> go run "c:\ALPRO\semester2_alpro2\alpro2_week4\latsol3-modul5\unguided3.go"
Masukkan bilangan: 12
Faktor dari 12: 1 2 3 4 6 12
```

Deskripsi program:

Program diatas ditulis dalam bahasa Go dan berfungsi untuk menampilkan semua faktor dari bilangan yang diinputkan menggunakan rekursif. Pertama deklarasikan fungsi faktorRecursive(n int, i int) untuk mencari dan mencetak faktor dari bilangan n. Fungsi ini akan mengecek apakah i adalah faktor dari n dengan kondisi  $n \% i$  adalah 0 jika benar maka nilai i akan dicetak. Setelah itu, fungsi akan memanggil dirinya sendiri dengan nilai i+1 hingga i melebihi n, yang menjadi base case untuk menghentikan rekursi. Pada fungsi main deklarasikan variabel N sebagai tipe data integer lalu pengguna diminta untuk memasukkan bilangan bulat positif. Kemudian, program mencetak teks "Faktor dari

N: " dan memanggil fungsi faktorRecursive(N, 1) untuk memulai pencarian faktor dari 1 hingga N.

#### **IV. KESIMPULAN**

Rekursif adalah teknik pemrograman di mana suatu fungsi memanggil dirinya sendiri untuk menyelesaikan sub-masalah dari suatu permasalahan hingga mencapai kondisi dasar (base-case). Rekursif digunakan sebagai alternatif dari perulangan dalam berbagai algoritma, seperti perhitungan pangkat, faktorial, deret Fibonacci, pencetakan pola bintang, dan pencarian faktor suatu bilangan. Penerapan rekursif perlu memperhatikan kondisi dasar yang tepat agar program tidak mengalami infinite loop. Penggunaan rekursif sering kali membuat kode menjadi lebih elegant dan mudah dipahami, terutama untuk permasalahan yang secara natural memiliki struktur rekursif.

## **V. REFERENSI**

Modul 5 – Praktikum Alpro 2