

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL VI
REKURSIF**



Disusun Oleh :

Ghilbran Alfaries Pryma

2311102267 S1IF-11-

05

Dosen Pengampu :

Arif Amrulloh, S.Kom., M.Kom

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Dasar Teori Rekursif dalam Golang

Rekursif adalah sebuah teknik dalam pemrograman di mana suatu fungsi memanggil dirinya sendiri hingga mencapai kondisi tertentu, yang dikenal sebagai base case, untuk menghentikan proses rekursif tersebut. Dalam pemrograman Golang, rekursif sering digunakan untuk menyelesaikan masalah yang dapat dipecah menjadi sub-masalah yang lebih kecil dan serupa, seperti perhitungan faktorial, deret Fibonacci, dan traversal struktur data pohon.

Konsep Dasar Rekursif

Pada rekursif, setiap kali fungsi memanggil dirinya sendiri, sistem akan menyimpan konteks pemanggilan fungsi sebelumnya di dalam call stack. Ketika kondisi dasar (base case) terpenuhi, proses pemanggilan berakhir, dan hasilnya dikembalikan secara berurutan sesuai dengan urutan pemanggilan yang tersimpan di dalam stack.

Berikut adalah dua komponen utama dalam fungsi rekursif:

- **Base Case:** Kondisi untuk menghentikan rekursi. Tanpa ini, fungsi akan terus memanggil dirinya sendiri tanpa henti, mengakibatkan stack overflow.
- **Recursive Case:** Bagian di mana fungsi memanggil dirinya sendiri. Biasanya, nilai yang diproses akan mengalami perubahan (misalnya, berkurang atau dibagi) hingga mencapai kondisi base case.

contoh sederhana dari implementasi rekursif pada fungsi faktorial dan deret Fibonacci:

```

package main import

"fmt"

func factorial(n int) int {

if n == 0 {      return 1

// Base case

}

return n * factorial(n-1) // Recursive case

}

func main() {   fmt.Println("Faktorial dari 5
adalah:", factorial(5)) }

```

Pada contoh ini, fungsi factorial akan memanggil dirinya sendiri dengan nilai nnn yang berkurang hingga mencapai 0, di mana 0! didefinisikan sebagai 1.

Kelebihan dan Kekurangan Rekursif

- **Kelebihan:**
Rekursif menawarkan solusi yang lebih sederhana dan elegan untuk masalah yang memiliki pola pemecahan yang berulang, seperti struktur data pohon.
Mampu memecah masalah kompleks menjadi sub-masalah yang lebih kecil dan serupa.
- **Kekurangan:**
Rekursi memerlukan lebih banyak memori karena setiap pemanggilan fungsi baru akan disimpan dalam call stack.

Tanpa kondisi dasar yang baik, rekursi dapat menyebabkan stack overflow dan kinerja yang kurang optimal.

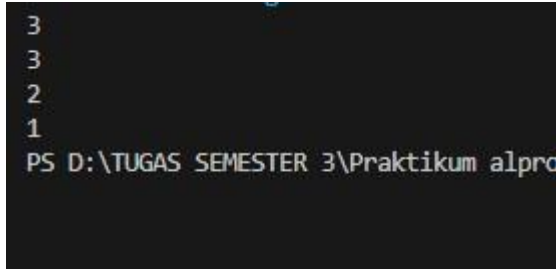
II. GUIDED I

Sourcecode

```
package main
import
"fmt"
func baris(bilangan
int) {    if bilangan
== 1 {
fmt.Println(1)
    } else {
fmt.Println(bilangan)
baris(bilangan - 1)
    }
}

//
func main() {
var n int
fmt.Scan(&n)
baris(n)
}
```

Screenshoot Output



```
3
3
2
1
PS D:\TUGAS SEMESTER 3\Praktikum alpro
```

Deskripsi Program

Program ini mencetak angka dari nilai yang dimasukkan oleh pengguna hingga 1 menggunakan rekursi.

1. Fungsi baris:

- Fungsi baris menerima angka sebagai parameter.
- Jika angka lebih besar dari 0, fungsi mencetak angka tersebut.
- Kemudian, fungsi memanggil dirinya sendiri dengan angka yang dikurangi 1.
- Proses ini berulang sampai angka mencapai 0, yang menjadi kondisi untuk menghentikan rekursi.

2. Fungsi main:

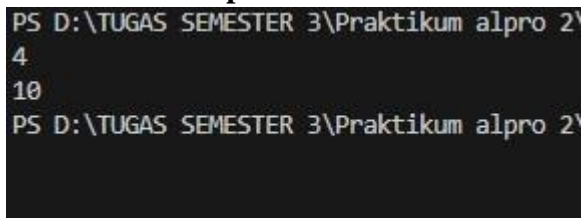
- Program dimulai dari fungsi main.
- Pengguna memasukkan sebuah angka, yang disimpan dalam variabel `n`.
- Fungsi baris dipanggil dengan `n` sebagai argumen, dan mulai mencetak angka dari `n` hingga 1.

GUIDED II

Sourcecode

```
package main
import
"fmt"
func penjumlahan(n int) int {
if n == 1 {           return 1
} else {           return n +
penjumlahan(n-1)
}
}
// func main() {      var n
int      fmt.Scan(&n)
fmt.Print(penjumlahan(n))
}
```

Screenshot Output



```
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\'
4
10
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\'
```

Deskripsi Program

Program ini menghitung jumlah dari angka n hingga 1 secara rekursif.

1. Fungsi penjumlahan:

- Jika n bernilai 1, fungsi mengembalikan 1.
- Jika n lebih besar dari 1, fungsi mengembalikan n ditambah hasil penjumlahan(n-1), sehingga jumlah semua angka dari n ke 1 dihitung.

2. Fungsi main:

- Pengguna memasukkan nilai n.
 - Fungsi penjumlahan(n) dipanggil dan hasilnya dicetak.
- Contoh: Jika $n = 5$, hasilnya adalah $5 + 4 + 3 + 2 + 1 = 15$

GUIDED III

Sourcecode

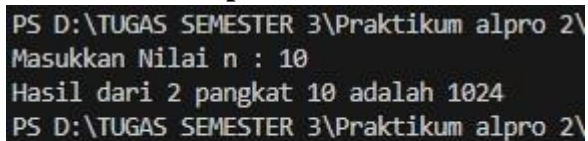
```
package main
import
"fmt"

//fungsi rekursif untuk menghitung
2^n func pangkat(n int) int {    if
n == 0 {        return 1    } else
{        return 2 * pangkat(n-1)
    }
}

//

func main() {
    var n int    fmt.Print("Masukkan Nilai n : ")
    fmt.Scan(&n)    fmt.Println("Hasil dari 2
pangkat", n, "adalah", pangkat(n))
}
```

Screenshoot Output



```
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\
Masukkan Nilai n : 10
Hasil dari 2 pangkat 10 adalah 1024
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\
```

Deskripsi Program

Program ini menghitung hasil dari (2^n) (2 pangkat n) menggunakan rekursi.

1. Fungsi pangkat:

- Jika n adalah 0, fungsi mengembalikan 1 (karena $(2^0 = 1)$).

- Jika n lebih dari 0, fungsi mengembalikan 2 dikalikan dengan hasil pangkat($n-1$), sehingga menghitung pangkat dua secara berulang hingga mencapai kondisi dasar.

2. Fungsi main:

- Program meminta pengguna memasukkan nilai n .
- Fungsi pangkat(n) dipanggil untuk menghitung hasil dari (2^n), dan hasilnya ditampilkan.

Contoh: Jika $n = 3$, hasilnya adalah ($2^3 = 8$).

GUIDED IV

Sourcecode

```
package main
import
"fmt"

//n = 5 //5*4*3*2*1 func
faktorial(n int) int {
if n == 0 || n == 1 {
return 1      } else {
return n * faktorial(n-1)
}
}

//
func main() {
var n int
fmt.Scan(&n)
    fmt.Println(faktorial(n))
}
```


Screenshoot Output

```
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul
4
24
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul
```

Deskripsi Program

Program ini menghitung faktorial dari sebuah angka n menggunakan rekursi.

1. Fungsi faktorial:

- Jika n adalah 0 atau 1, fungsi mengembalikan 1 (karena $0! = 1! = 1$), kondisi dasar).
- Jika n lebih besar dari 1, fungsi mengembalikan n dikalikan dengan hasil faktorial(n-1), sehingga terus mengalikan angka dari n ke 1.

2. Fungsi main:

- Program meminta pengguna memasukkan nilai n.
- Fungsi faktorial(n) dipanggil untuk menghitung faktorial, dan hasilnya ditampilkan.

Contoh: Jika $n = 5$, hasilnya adalah $5 * 4 * 3 * 2 * 1 = 120$.

III. UNGUIDED I

1. Soal Studi Case

Deret fibonacci adalah sebuah deret dengan nilai suku ke-0 dan ke-1 adalah 0 dan 1, dan nilai suku ke-n selanjutnya adalah hasil penjumlahan dua suku sebelumnya. Secara umum dapat diformulasikan $S_2 = S_{-1} + S_{-2}$ - Berikut ini adalah contoh nilai deret fibonacci hingga suku ke-10. Buatlah program yang mengimplementasikan fungsi rekursif pada deret fibonacci tersebut.

n	0	1	2	3	4	5	6	7	8	9	10
S_n	0	1	1	2	3	5	8	13	21	34	55

Sourcecode

```
package main
import
"fmt"

// Fungsi rekursif untuk menghitung deret
Fibonacci func fibonacci(n int) int {    if n <=
1 {
return n
}    return fibonacci(n-1) +
fibonacci(n-2)
}
func main() {
// Menghitung dan mencetak deret Fibonacci hingga suku
ke-10
for i := 0; i <= 10; i++ {
fmt.Printf("Fibonacci(%d) = %d\n", i, fibonacci(i))
}
}
```

Screenshoot Output

```
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\1>  
ul 5\UNGUIDED\1\main.go"  
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  
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\1>
```

Deskripsi Program

- Fungsi fibonacci adalah fungsi rekursif yang menerima parameter n dan mengembalikan nilai Fibonacci untuk n.
- Pada dasarnya, jika n adalah 0 atau 1, maka fungsi akan mengembalikan n (karena $\text{Fibonacci}(0) = 0$ dan $\text{Fibonacci}(1) = 1$).
- Jika n lebih dari 1, fungsi akan memanggil dirinya sendiri untuk menghitung nilai dari dua suku sebelumnya, yaitu $\text{fibonacci}(n-1) + \text{fibonacci}(n-2)$.
- Fungsi main akan memanggil fibonacci dari suku ke-0 hingga ke-10 dan mencetak hasilnya.

UNGUIDED II 2. Soal studi case

Buatlah sebuah program yang digunakan untuk menampilkan pola bintang berikut ini dengan menggunakan fungsi rekursif. N adalah masukan dari user.

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	5	* ** *** **** *****
2	1	*
3	3	* ** ***

Source code

```
package main
import
"fmt"

// Fungsi rekursif untuk menampilkan satu baris
bintang func printStars(n int) {    if n == 0 {
return
}
fmt.Print("*")
printStars(n - 1)
}

// Fungsi rekursif untuk menampilkan pola bintang
```

```

func printPattern(n, current int)
{
    if current > n {
return
    }    printStars(current)
fmt.Println()
printPattern(n, current + 1)
} func main() {    var n int
fmt.Print("Masukkan angka: ")
fmt.Scan(&n)

printPattern(n, 1)
}

```

Screenshoot Output

```

PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\2>
ul 5\UNGUIDED\2\main.go"
Masukkan angka: 5
*
**
***
****
*****
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\2>

```

Deskripsi Program

printStars: Fungsi ini adalah fungsi rekursif yang mencetak satu baris bintang sebanyak n.

- Jika n adalah 0, fungsi berhenti (return).
- Jika tidak, fungsi mencetak satu bintang (*) dan memanggil dirinya sendiri dengan n - 1.

printPattern: Fungsi ini adalah fungsi rekursif untuk mencetak pola bintang bertingkat hingga baris ke-n.

- Fungsi ini menerima dua parameter: n (jumlah total baris yang akan ditampilkan) dan current (baris saat ini yang sedang dicetak).
- Jika current lebih besar dari n, maka fungsi berhenti.
- Jika tidak, fungsi memanggil printStars(current) untuk mencetak bintang sesuai jumlah pada baris saat ini, lalu memanggil printPattern dengan current + 1 untuk mencetak baris berikutnya.

main: Program utama yang meminta input dari pengguna dan memanggil printPattern.

UNGUIDED III 3. Soal study case

Buatlah program yang mengimplementasikan rekursif untuk menampilkan faktor bilangan dari suatu N, atau bilangan yang apa saja yang habis membagi N.

Masukan terdiri dari sebuah bilangan bulat positif N.

Keluaran terdiri dari barisan bilangan yang menjadi faktor dari N (terurut dari 1 hingga Nya).

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	5	1 5
2	12	1 2 3 4 6 12

Source code

```
package main

import
(
    "fmt"
)
```

```
// Fungsi rekursif untuk mencari faktor
func findFactors(n, i int) {
    // Jika i lebih besar dari n, kita keluar dari
    rekursi    if i > n {        return
    }
    // Jika i adalah faktor dari n, kita
    cetak    if n%i == 0 {
    fmt.Printf("%d ", i)
    }
    // Rekursi dengan meningkatkan i
    findFactors(n, i+1)
} func main() {    var n int
fmt.Print("Masukkan bilangan positif: ")
fmt.Scan(&n)    fmt.Printf("Faktor dari
%d adalah: ", n)    findFactors(n, 1)
fmt.Println()
}
```

Screenshoot Output

```
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\3> go
ul 5\UNGUIDED\3\main.go"
Masukkan bilangan positif: 5
Faktor dari 5 adalah: 1 5
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\3> go
ul 5\UNGUIDED\3\main.go"
Masukkan bilangan positif: 12
Faktor dari 12 adalah: 1 2 3 4 6 12
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\3> █
```

Deskripsi program

- Fungsi findFactors memeriksa apakah i adalah faktor dari n. Jika iya, maka ia akan mencetaknya.
- Fungsi ini kemudian memanggil dirinya sendiri dengan i yang bertambah 1 hingga i melebihi n.
- Program utama menerima input dari pengguna dan memanggil findFactors untuk mencetak faktor-faktor dari angka tersebut.

UNGUIDED IV 4. Soal studi case

Buatlah program yang mengimplementasikan rekursif untuk menampilkan barisan bilangan tertentu.

Masukan terdiri dari sebuah bilangan bulat positif N.

Keluaran terdiri dari barisan bilangan dari N hingga 1 dan kembali ke N.

Contoh masukan dan keluaran:

Source code

```
package main
import
"fmt"

// Fungsi rekursif untuk menampilkan bilangan dari N ke
1 dan kembali ke N func printSequence(n, current int) {
// Cetak bilangan saat ini      fmt.Printf("%d ",
current)

    // Jika mencapai 1, mulai kembali naik ke
N    if current == 1 {      return
    }

    // Panggil rekursi untuk menurun
printSequence(n, current-1)

    // Cetak bilangan saat naik lagi dari 2 ke N
if current > 1 {
    fmt.Printf("%d ", current)
}
}

func main() {      var n int
fmt.Print("Masukkan bilangan positif: ")
fmt.Scan(&n)
```



```
        fmt.Printf("Barisan dari %d ke 1 dan kembali ke %d:", n, n)    printSequence(n, n)    fmt.Println()
    }
```

Screenshoot Output

```
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\4> go run "d:\ul 5\UNGUIDED\4\main.go"
Masukkan bilangan positif: 5
Barisan dari 5 ke 1 dan kembali ke 5: 5 4 3 2 1 2 3 4 5
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\4> go run "d:\ul 5\UNGUIDED\4\main.go"
Masukkan bilangan positif: 9
Barisan dari 9 ke 1 dan kembali ke 9: 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\4> 
```

Deskripsi Program

1. Fungsi printSequence akan mencetak angka saat ini (current).
2. Jika current adalah 1, maka rekursi berhenti (tidak turun lagi).
3. Jika tidak, rekursi akan menurun dengan current - 1 hingga mencapai 1.
4. Saat rekursi kembali, ia mencetak angka dari 2 hingga N untuk membuat urutan kembali ke atas.

UNGUIDED V 5. Soal studi case

Buatlah program yang mengimplementasikan rekursif untuk menampilkan barisan bilangan ganjil.

Masukan terdiri dari sebuah bilangan bulat positif N.

Keluaran terdiri dari barisan bilangan ganjil dari 1 hingga N.

Contoh masukan dan keluaran:

Source code

```
package main
import
(
    "fmt"
)

// Fungsi rekursif untuk menampilkan bilangan ganjil dari
1 hingga N func printOddSequence(n, current int) {
    // Jika current melebihi N, kita keluar dari
rekursi    if current > n {        return
    }
    // Cetak bilangan ganjil saat ini
fmt.Printf("%d ", current)
    // Panggil rekursi dengan menambah current sebesar 2
untuk mendapatkan bilangan ganjil berikutnya
printOddSequence(n, current+2)
} func
main() {
var n int
    fmt.Print("Masukkan bilangan positif: ")
fmt.Scan(&n)
    fmt.Printf("Barisan bilangan ganjil dari 1 hingga %d:
", n)    printOddSequence(n, 1)    fmt.Println()
}
```

Screenshoot Output

```
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\5> go run
ul 5\UNGUIDED\5\main.go"
Masukkan bilangan positif: 5
Barisan bilangan ganjil dari 1 hingga 5: 1 3 5
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\5> go run
ul 5\UNGUIDED\5\main.go"
Masukkan bilangan positif: 20
Barisan bilangan ganjil dari 1 hingga 20: 1 3 5 7 9 11 13 15 17 19
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\5> █
```

Deskripsi Program

- Fungsi printOddSequence dimulai dari 1 dan menampilkan bilangan ganjil.
- Rekursi berhenti ketika current melebihi N.
- Setiap kali fungsi dipanggil kembali, nilai current ditambah 2 untuk mendapatkan bilangan ganjil berikutnya

UNGUIDED VI 6. Soal studi case

Buatlah program yang mengimplementasikan rekursif untuk mencari hasil pangkat dari dua buah bilangan.

Masukan terdiri dari bilangan bulat x dan y.

Keluaran terdiri dari hasil x dipangkatkan y.

Catatan: diperbolehkan menggunakan asterik "*", ", tapi dilarang menggunakan import "math". **Contoh masukan dan keluaran:**

Source code

```
package main
import
"fmt"

func main() {    var x, y int
fmt.Print("Masukkan bilangan x: ")
fmt.Scanln(&x)

    fmt.Print("Masukkan bilangan y: ")
fmt.Scanln(&y)
    result := power(x, y)    fmt.Printf("Hasil
pangkat %d ^ %d adalah %d\n", x, y, result)
} func power(x, y int)
int {    if y == 0 {
return 1
    }    return x *
power(x, y-1)
}
```

Screenshoot Output

```
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\6> go run
ul 5\UNGUIDED\6\main.go"
Masukkan bilangan x: 2
Masukkan bilangan y: 2
Hasil pangkat 2 ^ 2 adalah 4
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\6> go run
ul 5\UNGUIDED\6\main.go"
Masukkan bilangan x: 5
Masukkan bilangan y: 3
Hasil pangkat 5 ^ 3 adalah 125
PS D:\TUGAS SEMESTER 3\Praktikum alpro 2\Modul 5\UNGUIDED\6> |
```

Deskripsi Program

Program ini menghitung pangkat dari bilangan `x` dengan eksponen `y`, di mana `x` dan `y` dimasukkan oleh pengguna.

- Program meminta pengguna untuk memasukkan nilai `x` (bilangan pokok) dan `y` (eksponen).
- Fungsi `power(x, y)` dipanggil untuk menghitung hasil dari `x` pangkat `y` menggunakan rekursi:
- Jika `y` sama dengan 0, fungsi mengembalikan 1 (karena bilangan pangkat 0 adalah 1).
- Jika tidak, fungsi mengembalikan $x * \text{power}(x, y-1)$, yaitu mengalikan `x` dengan hasil `power` dari eksponen yang berkurang satu.
- Hasilnya dicetak dalam bentuk `Hasil pangkat x^y adalah result`.

Contoh: Jika $x = 2$ dan $y = 3$, program akan mencetak `Hasil pangkat 2^3 adalah 8`.

IV. DAFTAR PUSTAKA

Donovan, A. A., & Kernighan, B. W. (2015). The Go Programming Language. Addison-Wesley.

Pike, R. (2010). Go Programming Language. golang.org. Retrieved from <https://golang.org/doc/>

Wampler, B., & Soni, K. (2018). Functional Programming in Go. Manning Publications.

Suri, S. (2021). Concurrency in Go: Tools and Techniques for Developers. O'Reilly Media.

Go Documentation. (n.d.). Tour of Go: Recursion. Retrieved from <https://tour.golang.org/>