

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

MODUL V

REKURSIF



Oleh:

FAJAR FARIZQI AZMI

2311102192

IF-11-02

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Dasar Teori

REKURSIF

Rekursif adalah teknik pemrograman di mana sebuah fungsi memanggil dirinya sendiri untuk memecahkan masalah yang lebih besar. Setiap kali sebuah fungsi rekursif dipanggil, masalah dipecah menjadi submasalah yang lebih kecil, hingga mencapai kasus dasar (base case), yang menghentikan rekursi.

Dalam konteks rekursi:

- **Base Case:** Kondisi yang harus dipenuhi agar rekursi berhenti. Jika base case tidak ditangani, rekursi akan terus berjalan tanpa henti (infinite recursion).
- **Recursive Case:** Bagian di mana fungsi memanggil dirinya sendiri untuk memecahkan submasalah.

Rekursi sering digunakan dalam masalah yang dapat dipecah menjadi masalah yang lebih kecil dengan pola berulang, seperti menghitung faktorial, Fibonacci, dan pencarian dalam struktur data pohon.

	Notasi Algoritma	Notasi dalam bahasa GO
1	procedure cetak(in x:integer)	func cetak(x int){
2	algoritma	fmt.Println(x)
3	output(x)	cetak(x+1)
4	cetak(x+1)	}
5	endprocedure	

Apabila diperhatikan subprogram cetak() di atas, terlihat pada baris ke-4 terdapat pemanggilan subprogram cetak() kembali. Misalnya apabila kita eksekusi perintah cetak(5) maka akan menampilkan angka 5 6 7 8 9... dst tanpa henti. Artinya setiap pemanggilan subprogram cetak() nilai x akan selalu bertambah 1 (increment by one) secara terus menerus tanpa henti.

II. GUIDED

GUIDED 1

Source code

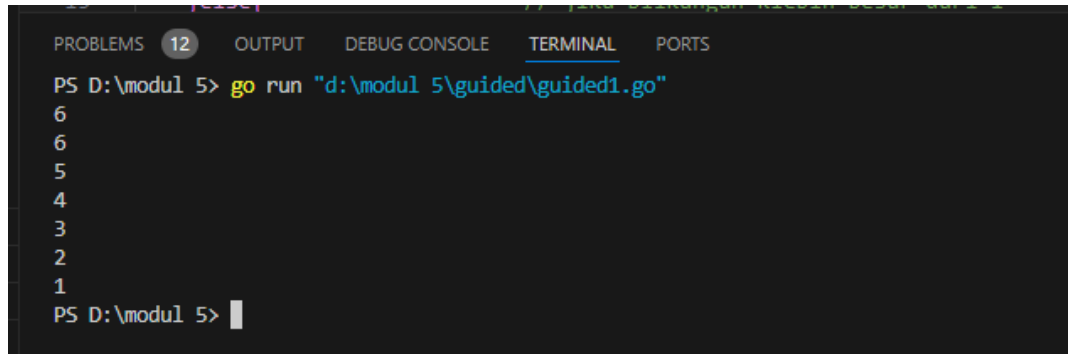
```
package main

import "fmt"

func main () {
    var n int
    fmt.Scan(&n) // memnbaca input pengguna
    baris (n)    // memanggil fungsi rekursif 'baris'
}

func baris(bilangan int) {
    if bilangan== 1 {      // base case : jika bilangan sama dengan 1
        fmt.Println(1)    // cetak angka 1
    }else{                // jika bilangan lebih besar dari 1
        fmt.Println(bilangan) // cetak bilangan saat ini
        baris (bilangan -1 ) // panggil fungsi baris '
    }
}
```

Screenshot output :



```
PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\modul 5> go run "d:\modul 5\guided\guided1.go"
6
6
5
4
3
2
1
PS D:\modul 5>
```

Deskripsi program :

Program ini menggunakan rekursi untuk mencetak bilangan secara menurun dari nilai n yang dimasukkan pengguna hingga 1. Program ini mencetak setiap bilangan mulai dari n ke bawah satu per satu, dan rekursi berakhir ketika bilangan mencapai 1.

GUIDED II

Source code

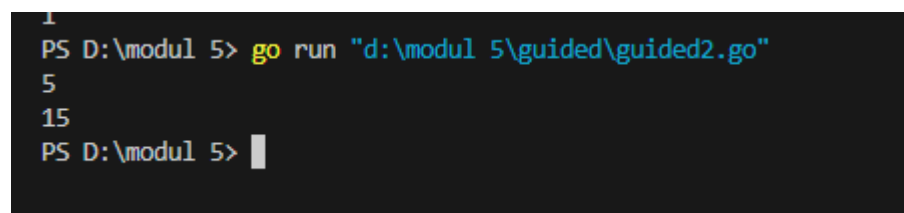
```
package main

import "fmt"

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

```
func penjumlahan(n int) int {  
    if n==1 {  
        return 1  
    } else {  
        return n + penjumlahan (n -1)  
    }  
}
```

Screenshot output :



```
1  
PS D:\modul 5> go run "d:\modul 5\guided\guided2.go"  
5  
15  
PS D:\modul 5> 
```

Deskripsi program :

Program ini menghitung penjumlahan bilangan bulat dari 1 hingga nilai n yang dimasukkan oleh pengguna menggunakan rekursi. Ini adalah implementasi dari penjumlahan deret bilangan bulat. Misalnya, jika pengguna memasukkan $n = 4$, maka hasilnya adalah 10 (karena $4 + 3 + 2 + 1$).

III. UNGUIDED

➤ UNGUIDED 1

Source code :

```
//fajar farizqi azmi  
// 2311102192  
  
package main  
  
import "fmt"  
  
func fibonacci(n int) int {  
    if n <= 1 {  
        return n // 2311102192  
    }  
    return fibonacci(n-1) + fibonacci(n-2)  
}  
  
func main() {  
    var n int
```

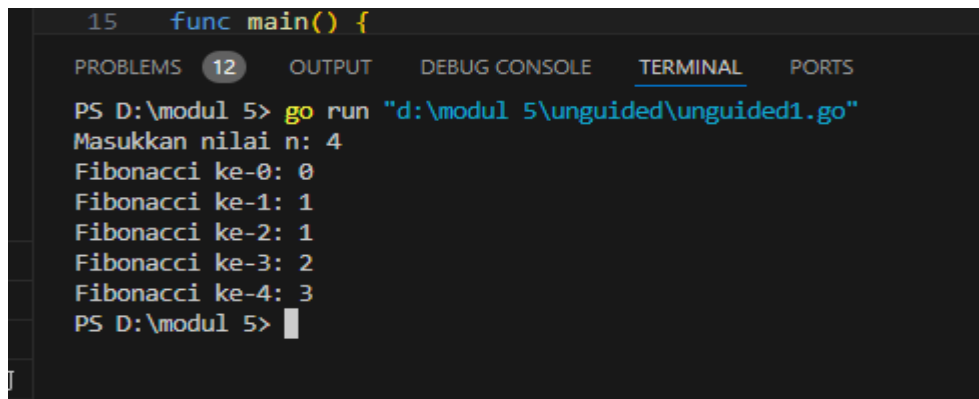
```

    fmt.Print("Masukkan nilai n: ")
    fmt.Scan(&n)

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

```

Screenshot output :



```

15 func main() {
PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\modul 5> go run "d:\modul 5\unguided\unguided1.go"
Masukkan nilai n: 4
Fibonacci ke-0: 0
Fibonacci ke-1: 1
Fibonacci ke-2: 1
Fibonacci ke-3: 2
Fibonacci ke-4: 3
PS D:\modul 5>

```

Deskripsi program :

Program ini menghitung dan mencetak deret bilangan Fibonacci hingga nilai ke-n yang dimasukkan oleh pengguna. Program menggunakan rekursi untuk menghitung setiap bilangan Fibonacci dan mencetaknya satu per satu hingga mencapai bilangan ke-n. Ini adalah contoh dari penggunaan rekursi untuk memecahkan masalah deret Fibonacci.

➤ UNGUIDED 2

Source code :

```

// fajar farizqi azmi
// 2311102192

package main

```

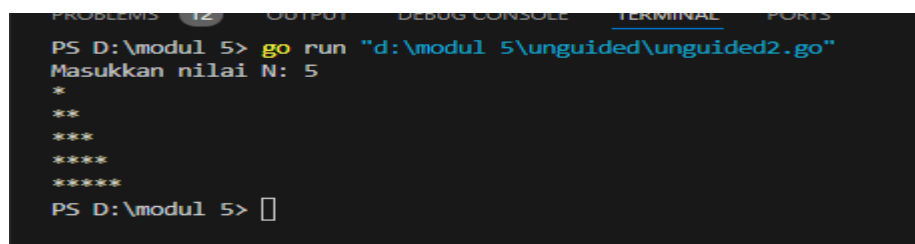
```
import "fmt"

func printbintang(n int) {
    if n == 1 {
        fmt.Println("*")
        return // 2311102192
    }
    printbintang(n - 1)
    for i := 0; i < n; i++ {
        fmt.Print("*")
    }
    fmt.Println()
}

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

    printbintang(n)
}
```

Screenshot output :

A screenshot of a terminal window showing the execution of a Go program. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The command 'go run "d:\modul 5\unguided\unguided2.go"' is entered, followed by the prompt 'Masukkan nilai N: 5'. The program outputs a series of asterisks: a single '*', two '**', three '***', four '****', and five '*****'. The prompt 'PS D:\modul 5>' is visible at the bottom.

```
PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\modul 5> go run "d:\modul 5\unguided\unguided2.go"
Masukkan nilai N: 5
*
**
***
****
*****
PS D:\modul 5> 
```


Deskripsi program :

Program ini menggunakan rekursi untuk mencetak pola segitiga bintang bertingkat, di mana jumlah bintang pada setiap baris meningkat dari satu hingga nilai n. Pendekatan rekursif memastikan bahwa bintang dicetak dari jumlah terkecil hingga terbesar.

➤ Unguided 3**Source code :**

```
//fajar farizqi azmi
// 2311102192

package main

import "fmt"

func findFactors(num, divisor int) {
    if divisor > num {
        return // 2311102192
    }

    if num%divisor == 0 {
```

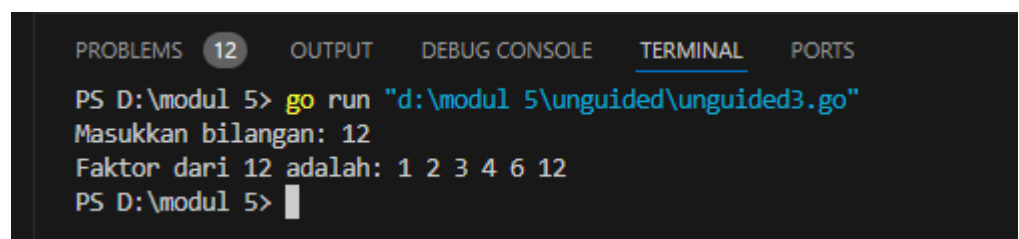
```
        fmt.Print(divisor, " ")
    }

    findFactors(num, divisor+1)
}

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

    fmt.Printf("Faktor dari %d adalah: ", number)
    findFactors(number, 1)
    fmt.Println()
}
```

Screenshot output :



```
PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\modul 5> go run "d:\modul 5\unguided\unguided3.go"
Masukkan bilangan: 12
Faktor dari 12 adalah: 1 2 3 4 6 12
PS D:\modul 5>
```

Deskripsi Program :

Program ini digunakan untuk menemukan dan mencetak semua faktor dari sebuah bilangan yang dimasukkan pengguna menggunakan pendekatan rekursif. Fungsi `findFactors` mengecek setiap pembagi dari 1 hingga `num`, dan mencetak pembagi yang valid (yaitu faktor dari bilangan tersebut).

➤ Unguided 4

Sourch code :

```
// fajar farizqi azmi
// 2311102192

package main

import "fmt"

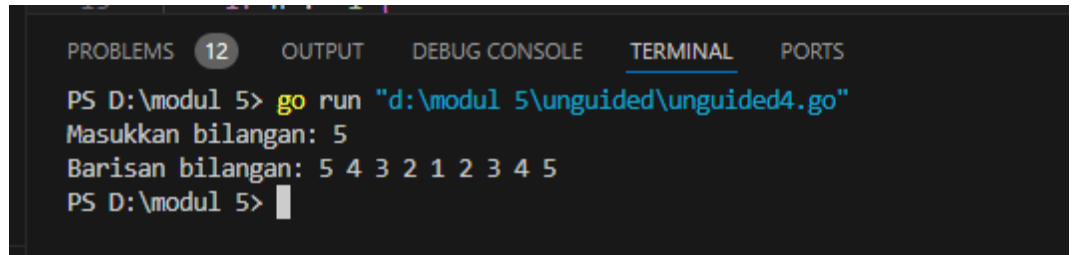
func printSequence(n int) {
    if n == 0 {
        return // 2311102192
    }
    fmt.Print(n, " ")
    printSequence(n - 1)
    if n != 1 {
        fmt.Print(n, " ")
    }
}

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

    fmt.Printf("Barisan bilangan: ")
    printSequence(number)
}
```

```
    fmt.Println()
}
```

Screenshot output :



```
PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\modul 5> go run "d:\modul 5\unguided\unguided4.go"
Masukkan bilangan: 5
Barisan bilangan: 5 4 3 2 1 2 3 4 5
PS D:\modul 5>
```

Deskripsi program :

Program ini mencetak urutan bilangan dari nilai n yang dimasukkan pengguna ke bawah hingga 1, lalu kembali mencetak bilangan tersebut ke atas hingga kembali ke n . Ini dilakukan secara rekursif, dengan pola pencetakan yang simetris.

➤ Unguided 5

Sourch code :

```
//fajar farizqi azmi
// 2311102192

package main

import "fmt"

func printOddNumbers(n int) {
    if n < 1 {
        return // 2311102192
    }
}
```

```

    printOddNumbers(n - 2)
    if n % 2 != 0 {
        fmt.Print(n, " ")
    }
}

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

    fmt.Printf("Bilangan ganjil dari 1 sampai %d: ",
number)
    printOddNumbers(number)
    fmt.Println()
}

```

Sceenshot output :

```

PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\modul 5> go run "d:\modul 5\unguided\unguided5.go"
Masukkan bilangan: 21
Bilangan ganjil dari 1 sampai 21: 1 3 5 7 9 11 13 15 17 19 21
PS D:\modul 5>

```

Deskripsi program :

Program ini menggunakan rekursi untuk mencetak semua bilangan ganjil dari 1 hingga bilangan n yang dimasukkan pengguna. Rekursi dilakukan dengan mengurangi nilai n sebanyak 2 di setiap langkah, sehingga hanya bilangan ganjil yang diproses dan dicetak.

➤ Unguided 6

Source code :

```
// fajar farizqi azmi
// 2311102192

package main

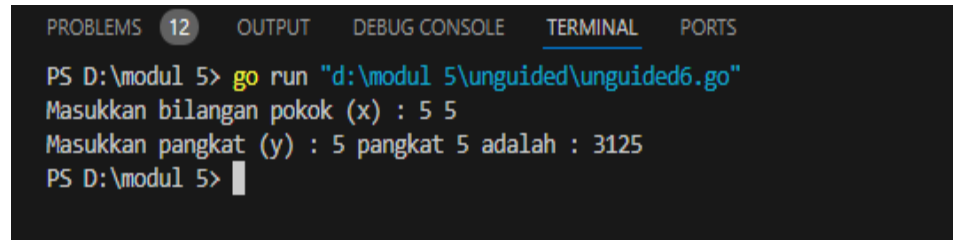
import "fmt"

func power(x, y int) int {
    if y == 0 {
        return 1 // 2311102192
    }
    return x * power(x, y-1)
}

func main () {
    var base, exponent int
    fmt.Print("Masukkan bilangan pokok (x) : ")
    fmt.Scan(&base)
    fmt.Print("Masukkan pangkat (y) : ")
    fmt.Scan(&exponent)

    result := power(base, exponent)
    fmt.Printf("%d pangkat %d adalah : %d\n", base, exponent, result)
}
```

Screenshot output :



```
PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\modul 5> go run "d:\modul 5\unguided\unguided6.go"
Masukkan bilangan pokok (x) : 5 5
Masukkan pangkat (y) : 5 pangkat 5 adalah : 3125
PS D:\modul 5>
```

Deskripsi program :

Program ini menghitung hasil pangkat dari suatu bilangan pokok x dipangkatkan dengan y menggunakan rekursi. Proses rekursi dilakukan dengan mengalikan bilangan pokok secara berulang hingga pangkatnya berkurang ke 0, dengan mengembalikan hasil 1 saat $y == 0$. Program ini memberikan solusi sederhana untuk menghitung pangkat dengan rekursi.