

**LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK**

**MODUL V
REKRUSIF**



**Telkom
University
PURWOKERTO**

Oleh:

Destia Ananda Putra

2311102176

IF-11-02

S1 TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

I. DASAR TEORI

Pengantar rekrusif adalah suatu subprogram baik fungsi atau prosedur bisa memanggil subprogram lainnya. Hal ini tidak menutup kemungkinan bahwa subprogram yang dipanggil adalah dirinya sendiri. Dalam pemrograman teknik ini dikenal dengan istilah rekursif. Rekursif secara sederhana dapat diartikan sebagai cara menyelesaikan suatu masalah dengan cara menyelesaikan sub-masalah yang identik dari masalah utama.

	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	

Catatan:

- Teknik rekursif ini merupakan salah satu alternatif untuk mengganti struktur kontrol perulangan dengan memanfaatkan subprogram (bisa fungsi ataupun prosedur).
- Untuk menghentikan proses rekursif digunakan percabangan (if-then).
- **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**.
- Setiap algoritma rekursif selalu memiliki padanan dalam bentuk algoritma iteratif.

Komponen Rekrusif

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

GUIDED

a. Source Code + Screenshot hasil program beserta penjelasan

1. Source Code

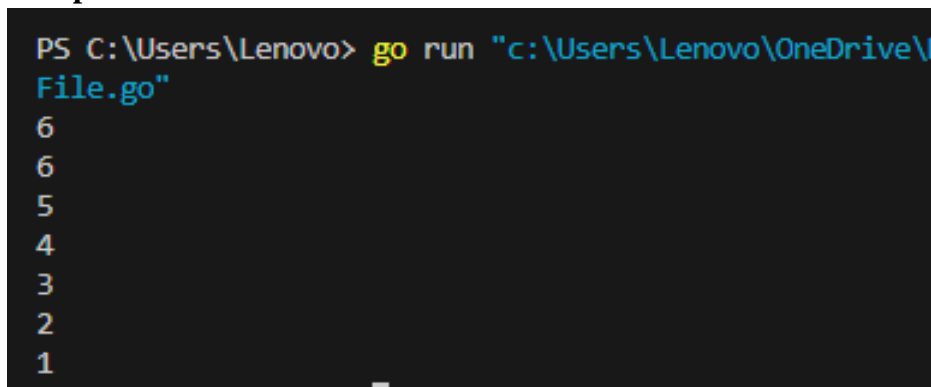
```
package main

import "fmt"

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

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

Output



```
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\OneDrive\
File.go"
6
6
5
4
3
2
1
```

Penjelasan

Program ini memasukkan angka n, lalu mencetak angka-angka dari n hingga 1 menggunakan fungsi rekursif baris . Fungsi tersebut akan mencetak nilai n, lalu memanggil dirinya sendiri dengan nilai n-1 hingga mencapai 1, di mana proses rekursi berhenti. mulai mencetak angka dari n (misalnya 6), kemudian terus mencetak angka yang lebih kecil hingga mencapai angka 1.

2. Source Code

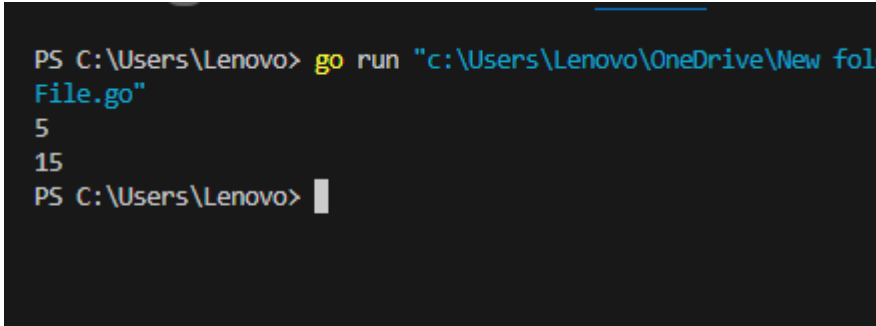
```
package main

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)
    }
}
```

Output



```
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\OneDrive\New fol
File.go"
5
15
PS C:\Users\Lenovo> 
```

Penjelasan

Program ini menggunakan teknik yang disebut rekursif, program ini seperti memanggil dirinya sendiri berulang-ulang untuk menyelesaikan masalah yang lebih kecil. Seperti menghitung jumlah dari 1 sampai 5, program akan memecah masalah menjadi menghitung jumlah dari 1 sampai 4, lalu menambahkan 5. Program akan terus memanggil dirinya sendiri sampai mencapai kondisi tertentu, yaitu ketika angka yang ingin dihitung sudah mencapai 1. Saat mencapai 1, program akan langsung mengembalikan nilai 1.

II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

1. Source Code

```
package main
import "fmt"

func main() {
    var n int
    fmt.Print("Masukkan nilai n: ")
    fmt.Scan(&n)
    for i := 0; i <= n; i++ {
        fmt.Print(fibonacci(i), " ")
    }
}

func fibonacci(n int) int {
    if n == 0 {
        return 0
    } else if n == 1 {
        return 1
    } else {
        return fibonacci(n-1) + fibonacci(n-2)
    }
}
```

Output

```
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\OneD
"
Masukkan nilai n: 10
0 1 1 2 3 5 8 13 21 34 55
PS C:\Users\Lenovo> █
```

Penjelasan

Program menghasilkan deret Fibonacci hingga nilai n yang dimasukkan. Deret Fibonacci adalah urutan bilangan di mana setiap bilangan adalah jumlah dari dua bilangan di mana rekursif untuk menghitung deret Fibonacci. pemrograman di mana sebuah fungsi memanggil dirinya sendiri hingga mencapai kondisi dasar ($n = 0$ atau 1). hasilnya adalah n ; untuk nilai lebih besar, hasilnya adalah jumlah dari dua angka Fibonacci sebelumnya ($\text{fibonacci}(n-1) + \text{fibonacci}(n-2)$).

2. Source Code

```
package main

import "fmt"

func main() {
    var n int
    fmt.Print("Masukkan angka: ")
    fmt.Scan(&n)
    cetakBintang(n, 1)
}

func cetakBintang(total, current int) {
    if current > total {
        return
    }
}
```

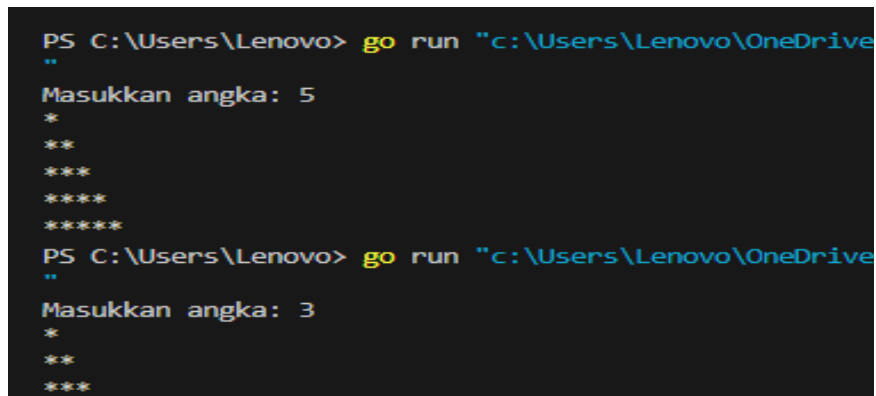
```

    }
    for i := 0; i < current; i++ {
        fmt.Print("*")
    }
    fmt.Println()
    cetakBintang(total, current+1)

    return
}

```

Output



```

PS C:\Users\Lenovo> go run "c:\Users\Lenovo\OneDrive
"
Masukkan angka: 5
*
**
***
****
*****
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\OneDrive
"
Masukkan angka: 3
*
**
***

```

Penjelasan

Program ini mencetak pola segitiga bintang bertingkat berdasarkan input pengguna menggunakan fungsi rekursif. Dimana memasukkan angka n yang menentukan jumlah baris dalam pola. Pada setiap baris, jumlah bintang yang dicetak bertambah sesuai dengan nomor barisnya, dimulai dari 1 bintang hingga n bintang. Program ini untuk mencetak pola sederhana, namun rekursi dapat digantikan dengan pendekatan iteratif untuk mengurangi overhead dan meningkatkan efisiensi terutama untuk nilai n yang besar. Seperti contoh nya $n=5$.

3. Source Code

```
package main

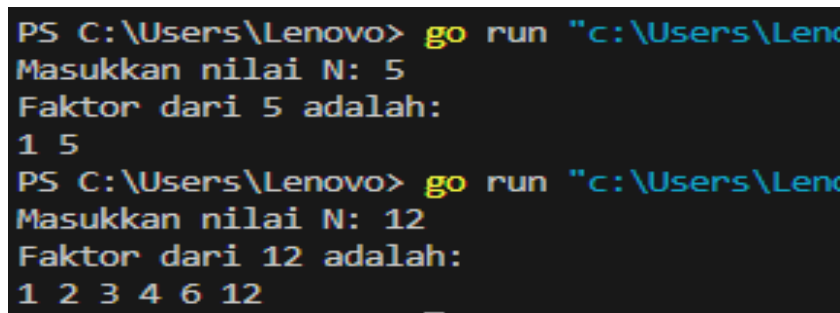
import "fmt"

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

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

    fmt.Println("Faktor dari", n, "adalah:")
    cetakFaktor(n, 1)
    fmt.Println()
    return
}
```

Output



```
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\source\main.go"
Masukkan nilai N: 5
Faktor dari 5 adalah:
1 5
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\source\main.go"
Masukkan nilai N: 12
Faktor dari 12 adalah:
1 2 3 4 6 12
```

Penjelasan

Program ini memasukkan sebuah angka n , lalu mencetak semua faktor dari angka tersebut menggunakan fungsi rekursif. Fungsi cetak Faktor memeriksa setiap bilangan i dari 1 hingga n , dan mencetak bilangan yang bisa membagi n tanpa sisa. Setiap kali ditemukan faktor, bilangan tersebut dicetak, dan fungsi memanggil dirinya sendiri dengan nilai i yang lebih besar. Proses ini terus berlanjut sampai

faktor ditemukan dan dicetak. Pendekatan rekursif yang digunakan membantu menyederhanakan proses pencarian faktor.

4. Source Code

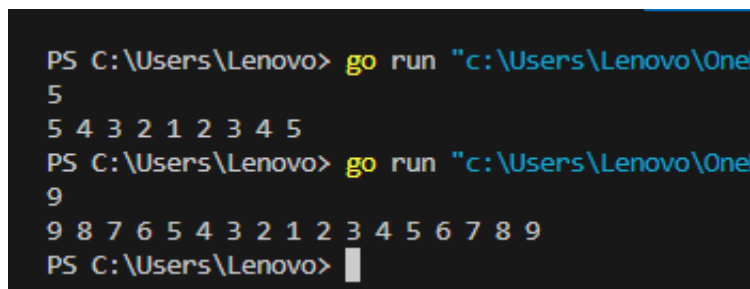
```
package main

import "fmt"

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

func baris(bilangan int) {
    if bilangan == 1 {
        fmt.Print(1, " ")
    } else {
        fmt.Print(bilangan, " ")
        baris(bilangan - 1)
        fmt.Print(bilangan, " ")
    }
}
```

Output



```
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\One
5
5 4 3 2 1 2 3 4 5
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\One
9
9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9
PS C:\Users\Lenovo> █
```

Penjelasan

Program ini mencetak pola angka dengan menggunakan rekursif. Setelah pengguna memasukkan angka *n*, fungsi *baris* mencetak angka tersebut dan menurunkannya secara bertahap hingga mencapai 1, lalu kembali mencetak angka dari 1 hingga *n*.

Fungsi ini bekerja dengan cara mencetak angka sebelum memanggil dirinya sendiri (rekursi) dengan angka yang dikurangi 1, lalu mencetak angka tersebut lagi setelah rekursi selesai, menghasilkan pola yang simetris.

5. Source Code

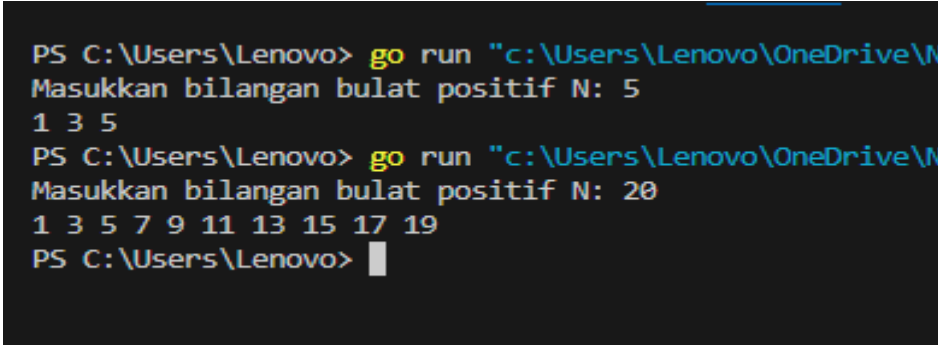
```
package main

import "fmt"

func cetakGanjil(n, current int) {
    if current > n {
        return
    }
    fmt.Print(current, " ")
    cetakGanjil(n, current+2)
}

func main() {
    var n int
    fmt.Print("Masukkan bilangan bulat positif N: ")
    fmt.Scan(&n)
    cetakGanjil(n, 1)
}
```

Output



```
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\OneDrive\N
Masukkan bilangan bulat positif N: 5
1 3 5
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\OneDrive\N
Masukkan bilangan bulat positif N: 20
1 3 5 7 9 11 13 15 17 19
PS C:\Users\Lenovo> █
```

Penjelasan

Program ini mencetak bilangan ganjil dari 1 hingga bilangan n yang dimasukkan oleh pengguna menggunakan fungsi rekursif. Setelah pengguna memasukkan nilai n, fungsi cetakGanjil memulai pencetakan dari 1, kemudian memanggil dirinya sendiri dengan menambah angka sebesar 2 (berpindah ke bilangan ganjil berikutnya) sampai angka yang dicetak melebihi n. Rekursi berhenti saat bilangan ganjil berikutnya lebih besar dari n. Contoh jika pengguna memasukkan n = 5, output yang dihasilkan adalah: 1 3 5

6. Source Code

```
package main

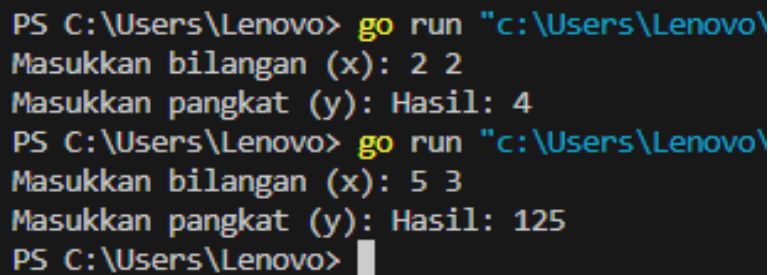
import "fmt"

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

    hasil := pangkat(x, y)
    fmt.Println("Hasil:", hasil)
}

func pangkat(x, y int) int {
    if y == 0 {
        return 1
    }
    return x * pangkat(x, y-1)
}
```

Output



```
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\
Masukkan bilangan (x): 2 2
Masukkan pangkat (y): Hasil: 4
PS C:\Users\Lenovo> go run "c:\Users\Lenovo\
Masukkan bilangan (x): 5 3
Masukkan pangkat (y): Hasil: 125
PS C:\Users\Lenovo> █
```

Penjelasan

Program ini menghitung hasil perpangkatan bilangan x dipangkatkan dengan y menggunakan rekursi. Setelah pengguna memasukkan nilai x (bilangan dasar) dan y (pangkat), program memanggil fungsi pangkat untuk menghitung nilai x^y . Fungsi ini bekerja dengan rekursi, di mana jika y bernilai 0, ia mengembalikan 1 (karena bilangan apa pun dipangkatkan dengan 0 hasilnya 1). Jika tidak, fungsi mengalikan x dengan hasil

pemanggilan fungsi pangkat(x , $y-1$) hingga y mencapai 0. program ini menghitung perpangkatan dengan cara mengalikan bilangan dasar secara berulang menggunakan rekursif sampai pangkatnya habis, sehingga menghasilkan operasi perpangkatan yang tepat.