

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 5
REKURSIF**



Oleh:

MUHAMMAD FAUZAN

103112400064

12 IF 01

**S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO**

2025

I. DASAR TEORI

5.1 Pengantar Rekursif

Pada modul-modul sebelumnya sudah dijelaskan bahwa 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. Sebagai contoh, perhatikan prosedur cetak berikut ini.

Notasi Algoritma

```
procedure cetak(in x:integer)
algorithm
    output(x)
    cetak(x+1)
endprocedure
```

Notasi dalam bahasa Go

```
func cetak(x int){
    fmt.Println(x)
    cetak(x+1)
}
```

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. Oleh karena itu, biasanya ditambahkan struktur kontrol percabangan (`if-then`) untuk menghentikan proses rekursif ini.

5.2 Komponen Rekursif

Algoritma rekursif terdiri dari dua komponen utama:

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

5.3 Contoh Program dengan Rekursif

Berikut adalah beberapa contoh program rekursif dalam bahasa Go.

a. Membuat baris bilangan dari n hingga 1

Base-case: `bilangan == 1`

```
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)
    }
}
```

b. Menghitung hasil penjumlahan 1 hingga n

Base-case: `n == 1`

```
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)
    }
}
```

c. Mencari dua pangkat n atau 2ⁿ

Base-case: $n == 0$

```
package main
import "fmt"

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

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

d. Mencari nilai faktorial atau n!

Base-case: $n == 0$ atau $n == 1$

```
package main
import "fmt"

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

func faktorial(n int) int {
    if n == 0 || n == 1 {
        return 1
    }else{
        return n * faktorial(n-1)
    }
}
```

II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

Contoh 1

```
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

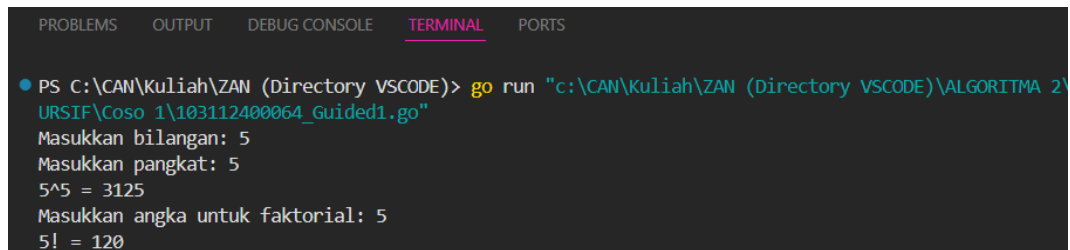
    //Input Pangkat
    fmt.Print("Masukkan bilangan: ")
    fmt.Scanln(&base)
    fmt.Print("Masukkan pangkat: ")
    fmt.Scanln(&exp)

    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))
}
```

Screenshots Output



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
● PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\
URSIF\Coso 1\103112400064_Guided1.go"
Masukkan bilangan: 5
Masukkan pangkat: 5
5^5 = 3125
Masukkan angka untuk faktorial: 5
5! = 120
```

// Foto hasil dari menjalankan code

Deskripsi: Program ini bertujuan untuk menghitung pangkat dan faktorial suatu bilangan menggunakan metode iteratif. Pengguna diminta memasukkan sebuah bilangan dan pangkatnya, kemudian program menghitung hasil perpangkatan dengan perulangan dan menampilkan hasilnya. Selanjutnya, pengguna diminta memasukkan angka untuk dihitung faktorialnya, yang juga dihitung menggunakan perulangan. Dengan cara ini, program memungkinkan perhitungan matematika sederhana secara efisien tanpa menggunakan rekursi.

Contoh 2

```
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)
}

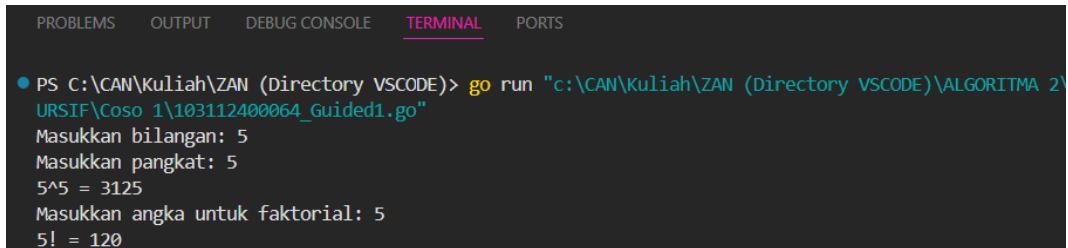
func main() {
    var base, exp, n int
    // Input pangkat
    fmt.Print("Masukkan bilangan: ")
    fmt.Scanln(&base)
    fmt.Print("Masukkan pangkat: ")
    fmt.Scanln(&exp)

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

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

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

Screenshots Output



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
● PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\
URSIF\Coso 1\103112400064_Guided1.go"
Masukkan bilangan: 5
Masukkan pangkat: 5
5^5 = 3125
Masukkan angka untuk faktorial: 5
5! = 120
```

// Foto hasil dari menjalankan code

Deskripsi: Program ini bertujuan untuk menghitung pangkat dan faktorial suatu bilangan menggunakan metode rekursif. Dalam perhitungan pangkat, program meminta pengguna memasukkan sebuah bilangan dan eksponennya, kemudian menghitung hasil perpangkatan dengan memanggil fungsi rekursif hingga eksponen mencapai nol. Sedangkan dalam perhitungan faktorial, pengguna diminta memasukkan angka yang dihitung faktorialnya menggunakan pendekatan rekursif hingga mencapai nilai dasar satu atau nol.

III. UNGUIDED

Source Code + Screenshot hasil program beserta penjelasan

Soal 1

```
//MUHAMMAD FAUZAN
//103112400064

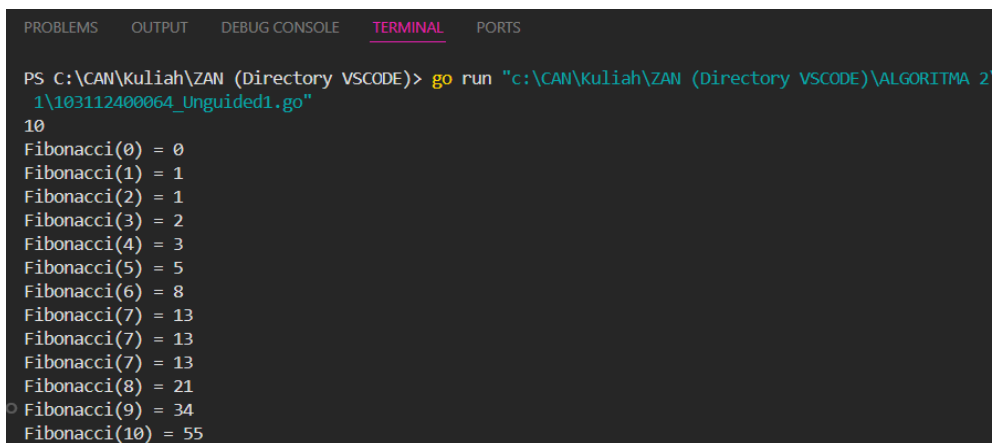
package main

import "fmt"

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

func main() {
    for indeks := 0; indeks <= 10; indeks++ {
        fmt.Printf("Fibonacci(%d) = %d\n", indeks, nilaiFibonacci(indeks))
    }
}
```

Screenshots Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2
1\103112400064_Unguided1.go"
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(7) = 13
Fibonacci(7) = 13
Fibonacci(8) = 21
Fibonacci(9) = 34
Fibonacci(10) = 55
```

// Foto hasil dari menjalankan code

Deskripsi: Program ini menghitung dan menampilkan deret Fibonacci hingga suku ke-10 menggunakan rekursi, di mana setiap suku dihitung sebagai jumlah dua suku sebelumnya hingga mencapai nilai dasar nol atau satu.

Soal 2

```
//MUHAMMAD FAUZAN
//103112400064

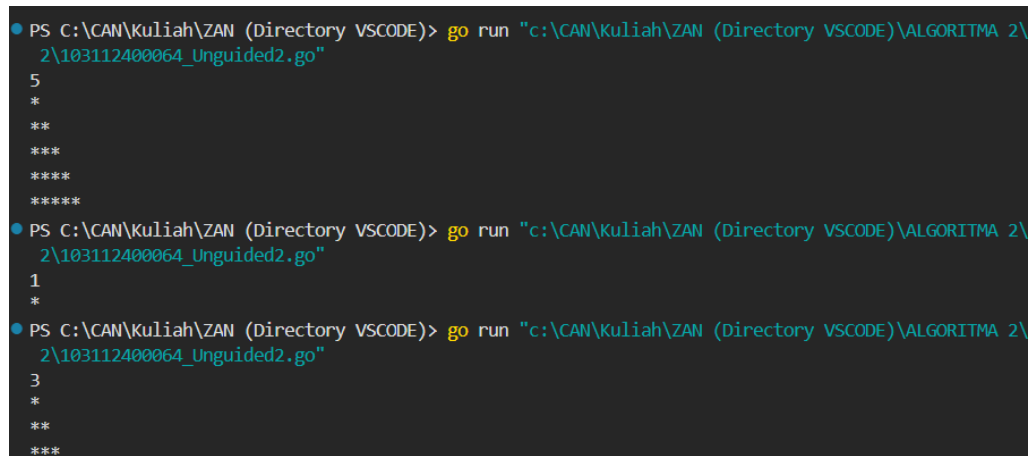
package main

import "fmt"

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

func main() {
    var N int
    fmt.Scan(&N)
    printStars(N)
}
```

Screenshots Output



```
PS C:\CAN\kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\2\103112400064_Unguided2.go"
5
*
**
***
****
*****

PS C:\CAN\kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\2\103112400064_Unguided2.go"
1
*

PS C:\CAN\kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\2\103112400064_Unguided2.go"
3
*
**
***
```

// Foto hasil dari menjalankan code

Deskripsi: Program ini bertujuan untuk mencetak pola bintang secara bertingkat menggunakan metode rekursif.

Soal 3

```
//MUHAMMAD FAUZAN
//103112400064

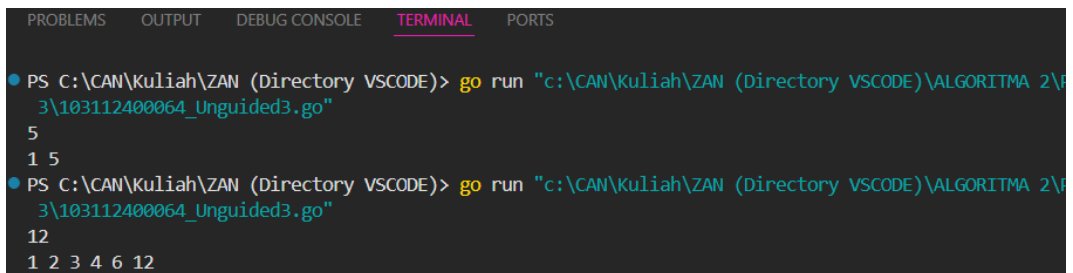
package main

import "fmt"

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

func main() {
    var N int
    fmt.Scan(&N)
    faktorRekursif(N, 1)
    fmt.Println()
}
```

Screenshots Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
● PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\F
3\103112400064_Unguided3.go"
5
1 5
● PS C:\CAN\Kuliah\ZAN (Directory VSCODE)> go run "c:\CAN\Kuliah\ZAN (Directory VSCODE)\ALGORITMA 2\F
3\103112400064_Unguided3.go"
12
1 2 3 4 6 12
```

// Foto hasil dari menjalankan code

Deskripsi: Program ini bertujuan untuk menampilkan semua faktor dari sebuah bilangan menggunakan metode rekursif. Fungsi rekursif akan memeriksa setiap angka dari 1 hingga bilangan tersebut, mencetak angka yang merupakan faktor, lalu memanggil dirinya sendiri dengan nilai yang bertambah satu hingga mencapai batas.

IV. KESIMPULAN

V. REFERENSI

Modul Praktikum Algoritma dan Pemrograman 2. (2025). *Modul 5: Rekursif*.
Fakultas Informatika, Telkom University.