

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL V  
REKURSIF**



**Disusun Oleh :**

**Fahrial Aufa Ramadhan / 2311102241**

**IF-11-6**

**Dosen Pengampu :**

**ABEDNEGO DWI SEPTIADI**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## I. DASAR TEORI

Apa itu recursive function? Recursive function adalah sebuah function yang memanggil/mengeksekusi dirinya sendiri. Recursive function bisa dikatakan salah satu yang bisa kita gunakan untuk melakukan perulangan. Ketika menulis kode aplikasi, terkadang ada kasus dimana akan lebih mudah jika dilakukan dengan recursive function. Contoh penggunaan sederhana recursive function adalah ketika melakukan operasi factorial. Saat membuat recursive function, kita harus memastikan function tersebut dapat berhenti. Jika tidak, maka akan terkena error stack overflow atau melebihi limit stack karena function terus memanggil dirinya sendiri. Oleh karena itu, biasanya recursive function tidak langsung memanggil dirinya sendiri tetapi bergantung pada kondisi tertentu.

### CONTOH RECURSIVE

Berikut ini adalah contoh recursive function yang digunakan untuk melakukan operasi factorial

```
func doFactorial(value int) int {  
    if value == 1 {  
        return value  
    }  
    return value * doFactorial(value-1)  
}
```

Function tersebut menerima value dengan tipe integer dan mengembalikan data dengan tipe integer. Seperti yang sudah dijelaskan sebelumnya bahwa recursive function harus dapat berhenti, maka kita membuat pengkondisian untuk memeriksa jika value bernilai 1 maka kita langsung mengembalikannya tanpa memanggil function lagi. Sebaliknya, jika value tidak sama dengan satu maka value akan dikalikan dengan nilai kembalian dari function doFactorial dengan mengisi parameter value dikurang 1. Pengurangan ini bertujuan agar parameter yang dikirimkan pada eksekusi function berikutnya adalah berisi nilai pada urutan sebelumnya. Sehingga, jika kita memanggil function doFactorial dengan parameter 5 maka parameter berikutnya akan menjadi 4, kemudian 3, kemudian 2, dan 1. Akhirnya recursive function tidak akan dipanggil lagi.

## I. GUIDED

### Soal Studi Case

Membuat baris bilangan nilai n untuk cetak bilangan n hingga 1

### Sourcecode

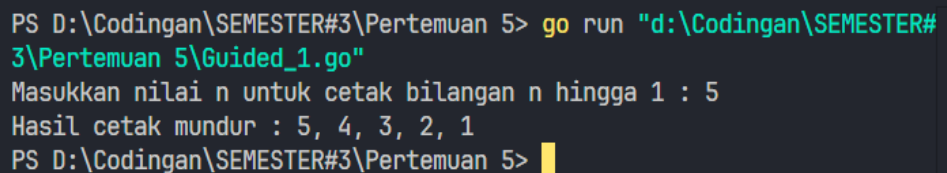
```
package main

import "fmt"

//fungsi mencetak bilangan dari n hingga 1
func cetakMundur(n int) {
    if n == 1 {
        fmt.Println(n)
        return
    }
    fmt.Print(n, " ")
    cetakMundur(n - 1)
}

func main() {
    var n int
    fmt.Print("Masukkan nilai n untuk cetak
bilangan n hingga 1 : ")
    fmt.Scanln(&n)
    fmt.Print("Hasil cetak mundur : ")
    cetakMundur(n)
}
```

### Screenshoot Output



```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#
3\Pertemuan 5\Guided_1.go"
Masukkan nilai n untuk cetak bilangan n hingga 1 : 5
Hasil cetak mundur : 5, 4, 3, 2, 1
PS D:\Codingan\SEMESTER#3\Pertemuan 5> 
```

### Deskripsi Program

mencetak deret bilangan mundur Fungsi cetakMundur memanggil dirinya sendiri dengan nilai input yang terus berkurang hingga mencapai kondisi dasar (n=1), lalu mencetak angka-angka tersebut secara terbalik dari nilai awal yang dimasukkan pengguna.

## II. GUIDED

### Soal Studi Case

Membuat baris bilangan nilai n untuk cetak bilangan n hingga 1

### Sourcecode

```
package main

import "fmt"

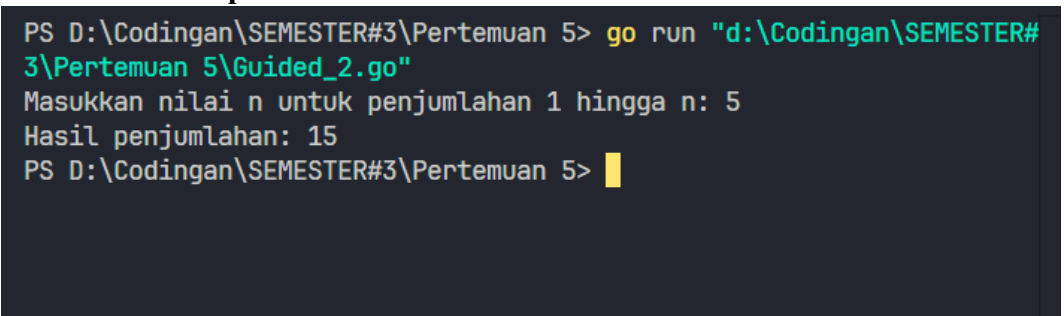
func jumlahRekursif(n int) int {
    if n == 1 {
        return 1
    }

    return n + jumlahRekursif(n-1)
}

func main() {
    var n int

    fmt.Print("Masukkan nilai n untuk penjumlahan 1
hingga n: ")
    fmt.Scanln(&n)
    fmt.Println("Hasil penjumlahan:", jumlahRekursif(n))
}
```

### Screenshot Output



```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#
3\Pertemuan 5\Guided_2.go"
Masukkan nilai n untuk penjumlahan 1 hingga n: 5
Hasil penjumlahan: 15
PS D:\Codingan\SEMESTER#3\Pertemuan 5> 
```

### Deskripsi Program

untuk menghitung jumlah bilangan bulat dari 1 hingga n. Fungsi jumlahRekursif memanggil dirinya sendiri secara berulang dengan nilai input yang terus berkurang (n-1), hingga mencapai kondisi dasar (n=1). Pada setiap pemanggilan, nilai n saat ini ditambahkan dengan hasil pemanggilan fungsi rekursif untuk n-1, sehingga pada akhirnya akan mengembalikan total penjumlahan dari 1 hingga n.

### III. GUIDED

#### Soal Studi Case

Mencari dua pangkat Base – Case:  $n == 0$

#### Sourcecode

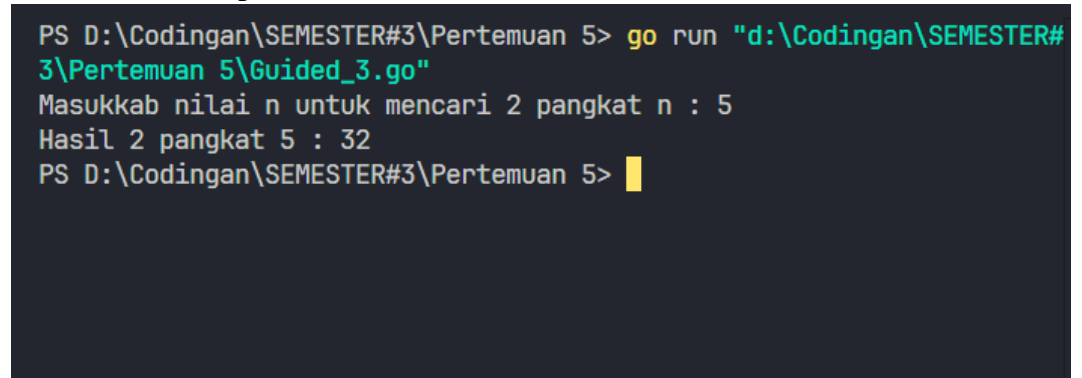
```
package main

import "fmt"

func pangkatDua(n int) int {
    if n == 0 {
        return 1
    }
    return 2 * pangkatDua(n-1)
}

func main(){
    var n int
    fmt.Print("Masukkan nilai n untuk mencari 2 pangkat n : ")
    fmt.Scanln(&n)
    fmt.Println("Hasil 2 pangkat", n, ":", pangkatDua(n))
}
```

#### Screenshoot Output



```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#3\Pertemuan 5\Guided_3.go"
Masukkan nilai n untuk mencari 2 pangkat n : 5
Hasil 2 pangkat 5 : 32
PS D:\Codingan\SEMESTER#3\Pertemuan 5> |
```

#### Deskripsi Program

menghitung 2 pangkat n. Fungsi pangkatDua secara rekursif memanggil dirinya sendiri dengan nilai input yang dikurangi satu hingga mencapai kondisi dasar ( $n=0$ ). Setiap pemanggilan mengalikan 2 dengan hasil pemanggilan rekursif berikutnya, sehingga pada akhirnya menghasilkan nilai 2 pangkat n

## IV. GUIDED

### Soal Studi Case

Mencari nilai faktorial atau  $n!$ . Base-Case:  $n == 0$  atau  $n == 1$

### Sourcecode

```
package main

import "fmt"

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

func main() {
    var n int
    fmt.Print("Masukkan bilangan non-negatif: ")
    fmt.Scan(&n)

    if n < 0 {
        fmt.Println("Faktorial tidak didefinisikan untuk bilangan negatif")
    } else {
        fmt.Println("Hasil faktorial:", faktorial(n))
    }
}
```

### Screenshoot Output

```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#3\Pertemuan 5\Guided_4.go"
Masukkan bilangan non-negatif: 4
Hasil faktorial: 24
PS D:\Codingan\SEMESTER#3\Pertemuan 5> 
```

### Deskripsi Program

menghitung faktorial dari sebuah bilangan bulat non-negatif menggunakan rekursi. Fungsi faktorial memanggil dirinya sendiri dengan nilai input yang dikurangi satu hingga mencapai kondisi dasar ( $n=0$  atau  $n=1$ ). Setiap pemanggilan mengalikan nilai  $n$  saat ini dengan hasil pemanggilan rekursif untuk  $n-1$ , sehingga menghasilkan perkalian beruntun dari  $n$  hingga 1

## V. UNGUIDED

### 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  $SS-1+S-2$  Berikut ini adalah contoh nilai deret fibonacci hingga suku ke-10. Buatlah program yang mengimplementasikan fungsi rekursif pada deret fibonacci tersebut.

### Sourcecode

```
package main

import "fmt"

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

func main() {
    var n int
    fmt.Print("Masukkan bilangan non-negatif: ")
    fmt.Scan(&n)

    if n < 0 {
        fmt.Println("Faktorial tidak didefinisikan untuk bilangan negatif")
    } else {
        fmt.Println("Hasil faktorial:", faktorial(n))
    }
}
```

## Screenshoot Output

```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#3\Pertemuan 5\Unguided_1.go"
0 1 2 3 4 5 6 7 8 9 10
0 1 1 2 3 5 8 13 21 34 55
PS D:\Codingan\SEMESTER#3\Pertemuan 5> 
```

## Deskripsi Program

Program ini mendemonstrasikan penggunaan rekursi untuk menghitung faktorial dari sebuah bilangan bulat non-negatif. Fungsi faktorial memanggil dirinya sendiri dengan nilai input yang terus berkurang ( $n-1$ ) hingga mencapai kondisi dasar ( $n=0$  atau  $n=1$ ).

## VI. UNGUIDED

### Soal Studi Case

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

### Sourcecode

```
package main

import "fmt"

func printStars(n int) {
    if n > 0 {
        fmt.Print("*")
        printStars(n - 1)
    }
}

func printPattern(n int) {
    if n > 0 {
        printPattern(n - 1)
        printStars(n)
        fmt.Println()
    }
}

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

    fmt.Println("Pola bintang:")
}
```



```
        printPattern(n)
    }
```

### Screenshot Output

```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#3\Pertemuan 5\Unguided_2.go"
Masukkan jumlah baris: 5
Pola bintang:
*
**
***
****
*****
PS D:\Codingan\SEMESTER#3\Pertemuan 5> █
```

### Deskripsi Program

mencetak pola segitiga bintang. Fungsi printStars mencetak bintang secara rekursif sebanyak n kali. Fungsi printPattern memanggil dirinya sendiri secara rekursif untuk mencetak baris-baris pola, lalu memanggil printStars untuk mencetak bintang pada setiap baris

## VII. UNGUIDED

### Soal Studi 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 N ya).

### Sourcecode

```
package main

import "fmt"

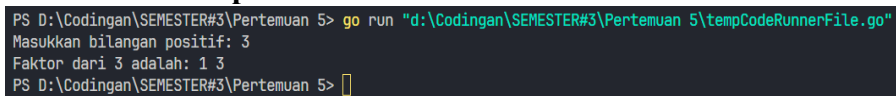
func faktor(n, bagi int) {
    if bagi > n {
        return
    }
    if n%bagi == 0 {
        fmt.Printf("%d ", bagi)
    }
    faktor(n, bagi+1)
}

func main() {
    var n int
```

```
        fmt.Print("Masukkan bilangan positif: ")
        fmt.Scan(&n)

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

### Screenshoot Output



```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#3\Pertemuan 5\tempCodeRunnerFile.go"
Masukkan bilangan positif: 3
Faktor dari 3 adalah: 1 3
PS D:\Codingan\SEMESTER#3\Pertemuan 5> 
```

### Deskripsi Program

suatu bilangan bulat positif. Fungsi faktor menggunakan rekursi untuk memeriksa setiap bilangan dari 1 hingga n. Jika n habis dibagi oleh bagi, maka bagi dicetak sebagai faktor.

## VIII. UNGUIDED

### 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

## Sourcecode

```
package main

import (
    "fmt"
)

func urutan(n, current int) {
    if current < 1 {
        return
    }

    fmt.Print(current, " ")

    if current > 1 {
        urutan(n, current-1)
    }

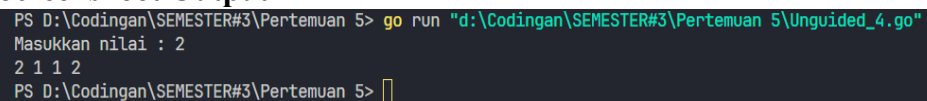
    fmt.Print(current, " ")
}

func main() {
    var n int

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

    urutan(n, n)
    fmt.Println()
}
```

## Screenshot Output



```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#3\Pertemuan 5\Unguided_4.go"
Masukkan nilai : 2
2 1 1 2
PS D:\Codingan\SEMESTER#3\Pertemuan 5> 
```

## Deskripsi Program

menghasilkan urutan bilangan unik dengan pola rekursif. Fungsi urutan mencetak bilangan current , kemudian memanggil dirinya sendiri secara rekursif dengan current-1 hingga mencapai 0. Setelah pemanggilan rekursif selesai, fungsi mencetak kembali bilangan current. Pola ini menghasilkan output berupa urutan bilangan menurun lalu naik kembali, membentuk pola cermin. Contoh, jika input n adalah 5, outputnya adalah: 5 4 3 2 1 1 2 3 4 5.

## IX. UNGUIDED

### 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.

### Sourcecode

```
package main

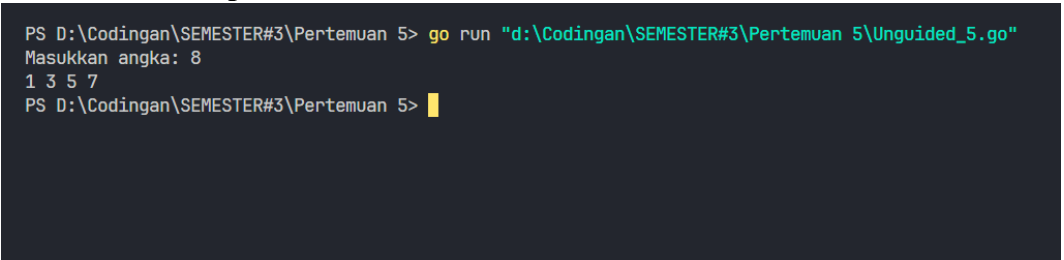
import (
    "fmt"
)

func ganjil(n, current int) {
    if current <= n {
        fmt.Print(current, " ")
        ganjil(n, current+2)
    }
}

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

    ganjil(n, 1)
    fmt.Println()
}
```

### Screenshoot Output



```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#3\Pertemuan 5\Unguided_5.go"
Masukkan angka: 8
1 3 5 7
PS D:\Codingan\SEMESTER#3\Pertemuan 5> 
```

### Deskripsi Program

Program ini untuk menghasilkan deret bilangan ganjil dari 1 hingga suatu batas yang ditentukan oleh pengguna. Fungsi `ganjil` mencetak bilangan ganjil saat ini dan secara rekursif memanggil dirinya sendiri dengan nilai berikutnya yang lebih besar dua, sehingga menghasilkan deret bilangan ganjil.

## X. UNGUIDED

### 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".

### Sourcecode

```
package main

import (
    "fmt"
)

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

    return x * pangkat(x, y-1)
}

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

    result := pangkat(x, y)
    fmt.Printf("%d pangkat %d adalah %d\n", x, y,
result)
}
```

### Screenshoot Output

```
PS D:\Codingan\SEMESTER#3\Pertemuan 5> go run "d:\Codingan\SEMESTER#3\Pertemuan 5\Unguided_6.go"
Masukkan nilai x: 3 4
Masukkan nilai y: 3 pangkat 4 adalah 81
PS D:\Codingan\SEMESTER#3\Pertemuan 5> █
```

**Deskripsi Program**

mengimplementasikan fungsi rekursif untuk menghitung pangkat suatu bilangan. Fungsi pangkat menerima dua argumen integer,  $x$  (basis) dan  $y$  (eksponen). Jika eksponen ( $y$ ) bernilai 0, fungsi mengembalikan 1. Jika tidak, fungsi mengembalikan hasil perkalian basis ( $x$ ) dengan pemanggilan rekursif dirinya sendiri dengan eksponen yang dikurangi 1 ( $y-1$ ).