

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL 5
MATERI**



Oleh:

DAFFA TSAQIFNA FAUZTSANY

103112400032

S1 IF-12-01

**S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025**

I. DASAR TEORI

Rekursif

1. Pengertian Rekursif

Rekursif adalah teknik pemrograman di mana **fungsi memanggil dirinya sendiri** untuk menyelesaikan masalah secara bertahap (sub-masalah). Biasanya digunakan sebagai alternatif dari perulangan.

2. Komponen Rekursif:

1. Base-case: kondisi berhenti agar tidak rekursif terus-menerus.
2. Recursive-case: bagian di mana fungsi memanggil dirinya sendiri.

Contoh Sederhana Rekursif:

Cetak angka dari x ke 10

```
func cetak(x int) {  
    if x == 10 {  
        fmt.Println(x)  
    } else {  
        fmt.Println(x)  
        cetak(x + 1)  
    }  
}
```

Contoh Program Rekursif Lain:

- Penjumlahan 1 sampai n

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

- Faktorial

```
func faktorial(n int) int {  
    if n <= 1 {  
        return 1  
    }  
    return n * faktorial(n - 1)  
}
```

3. Catatan Penting:

- Wajib memiliki base-case agar tidak infinite loop.
- Setiap rekursif bisa diganti dengan bentuk iteratif (looping).
- Gunakan hanya jika logika lebih sederhana dibanding perulangan.

II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

1. GUIDED 1

Source Code:

```
package main

import "fmt"

// Fungsi iteratif untuk menghitung pangkat (base^exp)
func pangkatIteratif(base, exp int) int {
    hasil := 1
    for i := 0; i < exp; i++ {
        hasil *= base
    }
    return hasil
}

// Fungsi iteratif untuk menghitung faktorial (n!)
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))
}
```

Output:

```

PS D:\test bs> go run 'd:\test bs\lab shit\smsr 2\103112400032_MODUL 5\guided 5\guide
d-5-1.go'
Masukkan bilangan: 5
Masukkan pangkat: 2
5^2 = 25
Masukkan angka untuk faktorial: 4
4! = 24

```

Deskripsi Program:

Program ini digunakan untuk menghitung dua operasi matematika secara **iteratif**:

1. **Pangkat** (base^{exp}) — dihitung dengan mengalikan base sebanyak exp kali.
 2. **Faktorial** ($n!$) — dihitung dengan mengalikan bilangan dari 2 hingga n.
- Program menerima input dari pengguna untuk bilangan pangkat dan angka faktorial, lalu mencetak hasil perhitungannya satu per satu.

2. GUIDED 2

Source Code:

```

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
    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))
    fmt.Print("masukkan angka untuk faktorial: ")
    fmt.Scanln(&n)

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

```

```

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
    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))
    fmt.Print("masukkan angka untuk faktorial: ")
    fmt.Scanln(&n)

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

```

Output:

```

PS D:\test bs> go run 'd:\test bs\lab shit\smsr 2\103112400032_MODUL 5\guided 5\guide
d-5-2.go'
masukkan bilangan: 6
masukkan pangkat: 6
6^6 = 46656
masukkan angka untuk faktorial: 5
5! = 120

```

Deskripsi Program:

Program ini digunakan untuk menghitung **pangkat** dan **faktorial** menggunakan metode **rekursif**.

- Fungsi pangkatRekursif menghitung hasil perpangkatan dengan memanggil dirinya sendiri hingga eksponen mencapai 0.
 - Fungsi faktorialRekursif menghitung faktorial dengan memanggil dirinya sendiri hingga n sama dengan 1 atau 0.
- Program menerima input bilangan dan eksponen untuk perhitungan pangkat, serta satu angka untuk perhitungan faktorial, lalu menampilkan hasilnya.

III. UNGUIDED

Source Code + Screenshot hasil program beserta penjelasan

1. UNGUIDED 1

Source Code:

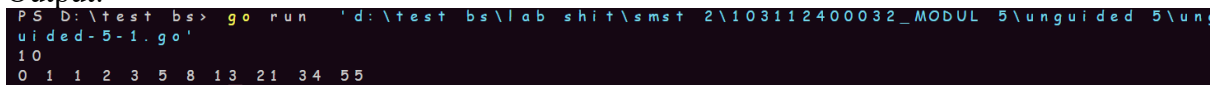
```
package main

import "fmt"

func Fibonacci(n int) int {
    if n <= 1 {
        return n
    }
    return Fibonacci(n-1) + Fibonacci(n-2)
}

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

Output:



```
P S D:\test bs> go run 'd:\test bs\lab shift\smst 2\103112400032_MODUL 5\unguided 5\un
guided-5-1.go'
0
0 1 1 2 3 5 8 13 21 34 55
```

Deskripsi Program:

Program ini digunakan untuk mencetak deret **Fibonacci** hingga suku ke-n menggunakan metode **rekursif**.

Fungsi Fibonacci(n) mendefinisikan nilai:

- $F(0) = 0$
- $F(1) = 1$
- $F(n) = F(n-1) + F(n-2)$ untuk $n > 1$

Program membaca satu bilangan dari input, lalu mencetak deret Fibonacci dari $F(0)$ hingga $F(n)$ secara berurutan.

2. UNGUIDED 2

Source Code:

```
package main

import "fmt"
```

```

func lines(x, y int) {
    if x == y {
        return
    }
    star(y)
    lines(x, y+1)
}
func star(x int) {
    if x < 0 {
        fmt.Println()
        return
    }
    fmt.Print("*")
    star(x - 1)
}

func main() {
    var x int
    fmt.Scan(&x)
    lines(x, 0)
}

```

Output:

Deskripsi Program:

Program ini digunakan untuk mencetak **pola segitiga menurun terbalik** berbentuk bintang (*) secara **rekursif**.

Fungsi `lines(x, y)` memanggil `star(y)` untuk mencetak $y + 1$ bintang pada setiap baris, dimulai dari 0 hingga $x - 1$.

Fungsi `star(x)` mencetak bintang sebanyak $x + 1$ lalu memanggil dirinya sendiri hingga mencapai nilai negatif, kemudian pindah ke baris baru.

Hasil akhir adalah segitiga horizontal yang dimulai dari 1 bintang hingga x baris dengan masing-masing bertambah panjang.

3. UNGUIDED 3

Source Code:

```

package main

```

```

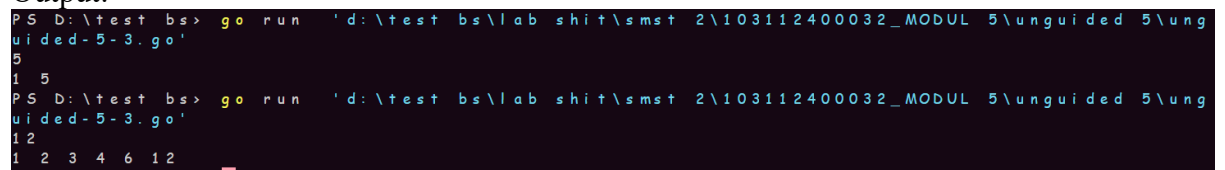
import "fmt"

func factorcheck(x, y int) {
    if x%y == 0 {
        fmt.Print(y, " ")
    }
    if x == y {
        return
    }
    factorcheck(x, y+1)
}

func main() {
    var x int
    fmt.Scan(&x)
    factorcheck(x, 1)
}

```

Output:



```

PS D:\test bs> go run 'd:\test bs\lab shit\sms1 2\103112400032_MODUL 5\unguided 5\unguided-5-3.go'
5
PS D:\test bs> go run 'd:\test bs\lab shit\sms1 2\103112400032_MODUL 5\unguided 5\unguided-5-3.go'
1 2 3 4 6 12

```

Deskripsi Program:

Program ini digunakan untuk mencetak semua **faktor pembagi** dari suatu bilangan x menggunakan **rekursi**.

Fungsi factorcheck(x, y) memeriksa apakah y adalah faktor dari x (yaitu $x \% y == 0$), lalu mencetaknya.

Pemanggilan rekursif dilakukan dengan menambah nilai y satu per satu hingga $y == x$.

Program akan mencetak semua bilangan yang habis membagi x dari 1 hingga x.

4. UNGUIDED 1

Source Code:

```

package main

import "fmt"

import "fmt"

func back(x, y int) {
    fmt.Print(x)
    if x == y {
        return
    }
    back(x+1, y)
}

func foward(x, y int) {
    fmt.Print(x)

```



```

    if x == 1 {
        back(x+1, y)
        return
    }
    foward(x-1, y)
}
func main() {
    var x int
    fmt.Scan(&x)
    foward(x, x)
}

```

Output:

```

PS D:\test bs> go run 'd:\test bs\lab shit\sms 2\103112400032_MODUL 5\unguided 5\unguided-5-4.go'
5
543212345
PS D:\test bs> go run 'd:\test bs\lab shit\sms 2\103112400032_MODUL 5\unguided 5\unguided-5-4.go'
9
98765432123456789

```

Deskripsi Program:

Program ini digunakan untuk mencetak **pola angka maju-mundur** dari 1 hingga x menggunakan **rekursi**.

Fungsi foward(x, y) mencetak angka secara menurun dari x ke 1.

Begitu mencapai 1, fungsi back(x+1, y) dipanggil untuk mencetak angka naik kembali dari 2 hingga x, membentuk pola simetris.

Hasil akhirnya adalah urutan angka menurun ke 1 lalu naik kembali ke nilai awal, seperti cermin.

5. UNGUIDED 1

Source Code:

```

package main

import "fmt"

func ganjil(x int, y int) {
    if y > x {
        return
    }
    if y%2 == 1 {
        fmt.Printf("%d ",y)
    }
    ganjil(x, y+1)
}

func main() {
    var x int
    fmt.Scan(&x)
    ganjil(x, 1)
}

```

Output:

```
PS D:\test bs> go run 'd:\test bs\lab shit\smsr 2\103112400032_MODUL 5\unguided 5\unguided-5-5.go'
5
1 3 5
PS D:\test bs> go run 'd:\test bs\lab shit\smsr 2\103112400032_MODUL 5\unguided 5\unguided-5-5.go'
20
1 3 5 7 9 11 13 15 17 19
```

Deskripsi Program:

Program ini digunakan untuk mencetak semua **bilangan ganjil** dari 1 hingga x secara **rekursif**.

Fungsi ganjil(x, y) memeriksa apakah y adalah ganjil ($y \% 2 == 1$) dan mencetaknya.

Pemanggilan rekursif dilakukan dengan menambah y satu per satu hingga $y > x$, sehingga seluruh bilangan ganjil dalam rentang tersebut ditampilkan.

6. UNGUIDED 1

Source Code:

```
package main

import "fmt"

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

func main() {
    var x, y int
    fmt.Scan(&x, &y)
    fmt.Print(pangkat(x, y))
}
```

Output:

```
PS D:\test bs> go run 'd:\test bs\lab shit\smsr 2\103112400032_MODUL 5\unguided 5\unguided-5-6.go'
2 2
4
PS D:\test bs> go run 'd:\test bs\lab shit\smsr 2\103112400032_MODUL 5\unguided 5\unguided-5-6.go'
5 3
125
```

Deskripsi Program:

Program ini digunakan untuk menghitung **nilai pangkat** dari suatu bilangan x^y menggunakan **rekursi**.

Fungsi pangkat(x, y) akan mengembalikan 1 jika $y == 0$ (karena setiap bilangan berpangkat nol bernilai 1), dan jika tidak, akan mengalikan x dengan hasil rekursi pangkat(x, y-1) hingga y mencapai 0.