LAPORAN PRAKTIKUM
ALGORITMA PEMROGRAMAN 2
MODUL V
REKURSIF

Oleh :

Geranada Saputra Priambudi

2311102008

IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi S.Kom., M.Kom

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

I. DASAR TEORI

Rekursif adalah konsep pemrograman yang melibatkan pemanggilan fungsi dari dalam dirinya sendiri untuk menyelesaikan masalah yang lebih besar melalui pembagian menjadi sub-masalah yang lebih kecil. Dalam bahasa pemrograman Go, rekursif digunakan untuk menggantikan peran perulangan dalam kasus tertentu, terutama ketika pemrosesan sub-masalah serupa dibutuhkan. Setiap kali fungsi rekursif dipanggil, fungsi tersebut membuat "cabang" baru dari proses dengan menggunakan nilai parameter yang lebih kecil atau mendekati kondisi yang disebut sebagai "base case," yang menghentikan rekursi.

Pada contoh sederhana dalam Go, fungsi rekursif seperti factorial atau power memperlihatkan bagaimana proses rekursif bekerja. Fungsi `factorial(A int) int`, misalnya, mengembalikan hasil faktorial dari nilai A dengan melakukan perkalian nilai tersebut secara berulang hingga mencapai nilai dasar, yaitu 1, yang kemudian menghentikan rekursi. Pada setiap panggilan rekursif, Go menyimpan keadaan variabel pada stack, sehingga bisa kembali ke fungsi pemanggil setelah base case tercapai. Penggunaan rekursif ini membuat kode lebih ringkas dan terkadang lebih mudah dibaca dalam situasi tertentu.

Konsep rekursif pada Go juga erat kaitannya dengan induksi matematika, di mana setiap masalah yang diselesaikan memiliki dua komponen utama: basis atau base case dan induksi. Base case adalah kondisi ketika fungsi tidak perlu lagi memanggil dirinya sendiri, sedangkan induksi adalah bagian dari fungsi yang memanggil fungsi itu sendiri dengan nilai yang mendekati base case. Misalnya, dalam perhitungan 2 pangkat n, basisnya adalah ketika `n == 0`, sedangkan bagian induksi akan terus memanggil `power(n-1)` hingga mencapai basis tersebut.

Meskipun rekursif dapat menggantikan perulangan dalam banyak kasus, penggunaannya harus dilakukan dengan hati-hati dalam Go karena setiap pemanggilan fungsi membutuhkan alokasi memori pada stack. Hal ini bisa menyebabkan *stack overflow* jika rekursi terlalu dalam atau tidak memiliki base case yang jelas. Maka, meskipun rekursif memberikan solusi elegan untuk masalah seperti perhitungan faktorial atau pangkat, efisiensi dan batasan stack perlu dipertimbangkan untuk mencegah kesalahan eksekusi.

II. GUIDED

1. Source Code:

```
package main

import "fmt"

// Fungsi untuk 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 dari n hingga
1: ")
    fmt.Scanln(&n)
    fmt.Print("Hasil cetak mundur: ")
    cetakMundur(n)
}
```

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102
8_06\Modul5\Guided1\guided1.go"
Masukkan nilai n untuk cetak bilangan dari n hingga 1: 8
Hasil cetak mundur: 8 7 6 5 4 3 2 1
```

Penjelasan:

Program diatas menggunakan konsep rekursif untuk mencetak angka secara mundur dari angka yang dimasukkan pengguna hingga 1. Di dalamnya, ada fungsi bernama cetakMundur yang menerima satu angka, yaitu n. Fungsi ini akan mengecek, “Apakah n sudah sama dengan 1?”. Kalau iya, maka n langsung dicetak dan program berhenti (return). Tapi, kalau n belum 1, program akan mencetak angka n diikuti spasi, lalu memanggil dirinya sendiri (cetakMundur) dengan nilai n - 1 (jadi angkanya turun satu). Proses ini akan terus berulang sampai n akhirnya menjadi 1, lalu program berhenti. Di fungsi main, kita diminta memasukkan angka n, lalu program akan menjalankan cetakMundur dari angka itu ke bawah, misalnya kalau n dimasukkan 8, hasilnya akan jadi: 8 7 6 5 4 3 2 1.

2. Source Code:

```
package main

import "fmt"

// Fungsi untuk menghitung penjumlahan 1 hingga n
```

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

func main() {
    var n int
    fmt.Print("Masukkan nilai n untuk penjumlahan 1 hingga n: ")
    fmt.Scanln(&n)
    fmt.Println("Hasil penjumlahan:", jumlahRekursi(n))
}
```

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_
8_06\Modul5\Guided2\guided2.go"
Masukkan nilai n untuk penjumlahan 1 hingga n: 6
Hasil penjumlahan: 21
```

Penjelasan:

Program diatas menghitung jumlah angka dari 1 hingga n menggunakan fungsi rekursif bernama jumlahRekursi. Di dalam fungsi ini, kita memiliki parameter n yang menentukan hingga angka berapa jumlah akan dihitung. Pertama, fungsi akan memeriksa apakah nilai n sudah sama dengan 1. Jika ya, fungsi akan mengembalikan nilai 1 sebagai dasar (base case), yang menandakan akhir dari proses rekursif. Namun, jika n lebih besar dari 1, program akan menambahkan n dengan hasil dari pemanggilan fungsi jumlahRekursi dengan nilai n-1. Dengan cara ini, setiap pemanggilan fungsi mengurangi n hingga mencapai 1, di mana semua hasil akan dijumlahkan satu per satu. Di bagian main, pengguna diminta memasukkan angka n, dan program akan memanggil fungsi jumlahRekursi untuk menghitung dan mencetak hasilnya. Contohnya, jika n diinput sebagai 6, maka hasil penjumlahannya adalah 1 + 2 + 3 + 4 + 5 + 6, yang menghasilkan 21.

3. **Source Code:**

```
package main

import "fmt"

// Fungsi untuk mencari 2 pangkat n
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))
}
```

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi
8_06\Modul5\Guided3\guided3.go"
Masukkan nilai n untuk mencari 2 pangkat n: 3
Hasil 2 pangkat 3 : 8
```

Penjelasan:

Program diatas menggunakan fungsi rekursif untuk menghitung nilai 2 pangkat n. Di dalam fungsi pangkatDua, jika nilai n sama dengan 0, maka program langsung mengembalikan nilai 1, karena apa pun yang dipangkatkan 0 hasilnya adalah 1 (ini adalah base case). Jika n lebih dari 0, fungsi akan mengembalikan hasil perkalian 2 dengan pangkatDua(n-1), yang artinya mengurangi nilai n satu per satu hingga mencapai 0. Dengan setiap pemanggilan, 2 dikalikan secara berturut-turut sesuai nilai n. Pada fungsi main, pengguna diminta memasukkan angka n, dan program akan menghitung serta menampilkan hasil 2 pangkat n. Sebagai contoh, jika n adalah 3, maka perhitungannya adalah 2 * 2 * 2, yang menghasilkan 8.

4. **Source Code:**

```
package main

import "fmt"

// Fungsi untuk menghitung faktorial n!
func faktorial(n int) int {
    if n == 0 || n == 1 {
        return 1
    }
    return n * faktorial(n-1)
}

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

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2
8_06\Modul5\Guided4\guided4.go"
Masukkan nilai n untuk mencari faktorial n!: 5
Hasil faktorial dari 5 : 120
```

Penjelasan:

Program diatas menghitung faktorial dari angka n menggunakan fungsi rekursif bernama faktorial. Fungsi ini bekerja dengan mengecek apakah n bernilai 0 atau 1. Jika ya, maka fungsi langsung mengembalikan 1, karena faktorial dari 0 dan 1 sama-sama 1 (ini adalah base case). Namun, jika n lebih dari 1, fungsi akan mengalikan n dengan hasil dari faktorial(n-1), yang berarti fungsi memanggil dirinya sendiri dengan nilai n yang dikurangi satu. Proses ini berulang hingga n mencapai 1, kemudian hasil perkalian dari setiap tahap dikalikan untuk mendapatkan hasil akhir. Di fungsi main, pengguna diminta memasukkan nilai n, lalu hasil faktorial dihitung dan ditampilkan. Misalnya, jika n adalah 5, maka hasil faktorialnya adalah 5 * 4 * 3 * 2 * 1, yang menghasilkan 120.

III. UNGUIDED

1. Deret fibonanci 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 $S_n = S_{n-1} + S_{n-2}$. Berikut ini adalah contoh nilai deret fibonanci hingga suku ke-10.

n	0	1	2	3	4	5	6	7	8	9	10
S_n	0	1	1	2	3	5	8	13	21	34	55

Buatlah program yang mengimplementasikan fungsi rekursif pada deret fibonanci tersebut.

Source Code:

```
package main

import "fmt"

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

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

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

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPrian
8_06\Modul5\Unguided1\tempCodeRunnerFile.go"
Masukkan n: 10
Deret Fibonacci hingga suku ke- 10
S0: 0
S1: 1
S2: 1
S3: 2
S4: 3
S5: 5
S6: 8
S7: 13
S8: 21
S9: 34
S10: 55
```

Penjelasan:
Program diatas untuk menghitung dan menampilkan deret Fibonacci hingga suku ke-n yang diinput oleh pengguna. Pertama, ada fungsi fibonacci yang menggunakan pendekatan rekursif untuk menghitung nilai suku Fibonacci. Jika nilai n yang dimasukkan adalah 0, maka fungsi mengembalikan 0, dan jika n adalah 1, fungsi mengembalikan 1. Untuk nilai n yang lebih besar, fungsi akan memanggil dirinya sendiri untuk menghitung dua suku sebelumnya dan menjumlahkannya. Dalam fungsi main, program meminta pengguna untuk memasukkan nilai n dan kemudian mencetak deret Fibonacci dari suku ke-0 hingga suku ke-n dengan format yang jelas, sehingga pengguna bisa melihat suku Fibonacci beserta nilainya secara terstruktur. Misalnya, jika pengguna memasukkan 10, program akan menampilkan suku dari S0 hingga S10 dengan nilai masing-masing, seperti S0: 0, S1: 1, S2: 1, dan seterusnya hingga S10: 55.

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

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	5	* ** *** **** *****
2	1	*
3	3	* ** ***

Source Code:

```
package main

import "fmt"

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

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

    fmt.Println("Pola bintang:")
    polaBintang(n, 1)
}
```

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputr
8_06\Modul5\Unguided2\unguided2.go"
Masukkan nilai N: 5
Pola bintang:
*
**
***
****
*****
```

```
PS C:\Users\Asus\Documents\GeranadaSaputraP
8_06\Modul5\Unguided2\unguided2.go"
Masukkan nilai N: 1
Pola bintang:
*
```

```
8_06\Modul5\Unguided2\unguided2.go"
Masukkan nilai N: 3
Pola bintang:
*
**
***
```

Penjelasan:

Program diatas adalah sebuah program dalam bahasa Go yang berfungsi untuk menampilkan pola bintang berdasarkan input dari pengguna. Pertama, program meminta pengguna untuk memasukkan sebuah angka N, yang menentukan jumlah baris bintang yang akan dicetak. Fungsi polaBintang kemudian dipanggil dengan dua parameter: n (nilai yang dimasukkan oleh pengguna) dan baris (yang dimulai dari 1). Di dalam fungsi ini, jika nilai baris lebih besar dari n, maka fungsi akan berhenti. Jika tidak, program akan mencetak sejumlah bintang yang sesuai dengan nilai baris, lalu melanjutkan ke baris berikutnya dengan memanggil dirinya sendiri dan menambah nilai baris sebesar 1. Ini terus berlanjut hingga semua baris bintang dicetak sesuai dengan jumlah yang diminta. Hasilnya adalah pola bintang yang terstruktur, di mana setiap baris memiliki jumlah bintang yang meningkat satu per satu hingga mencapai N.

3. 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).
   Contoh masukan dan keluaran:

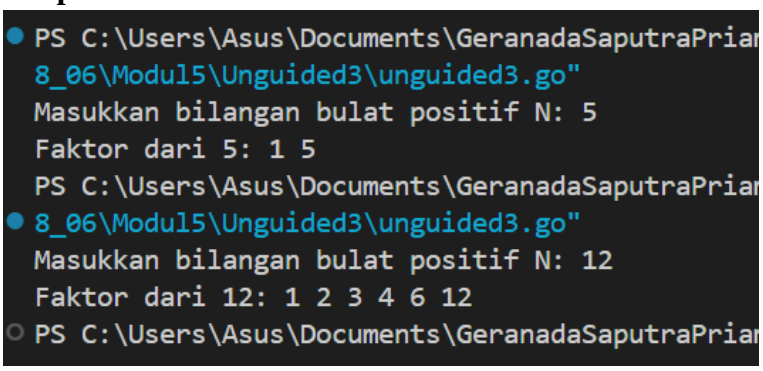
No	Masukan	Keluaran
1	5	1 5
2	12	1 2 3 4 6 12

Source Code:

```
package main

import "fmt"

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

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

    fmt.Print("Faktor dari ", n, ": ")
    faktorBilangan(n, 1)
```

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

Output:



Penjelasan:

Program diatas adalah program dalam bahasa Go yang berfungsi untuk mencari dan menampilkan faktor dari sebuah bilangan bulat positif yang dimasukkan oleh pengguna. Ketika program dijalankan, pengguna diminta untuk memasukkan bilangan bulat positif N. Fungsi faktorBilangan kemudian dipanggil dengan dua parameter: n (bilangan yang dicari faktornya) dan i (yang mulai dari 1). Di dalam fungsi, pertama-tama dicek apakah i sudah melebihi n; jika iya, maka fungsi berhenti. Jika tidak, program memeriksa apakah n dapat dibagi oleh i tanpa sisa (menggunakan operasi modulus). Jika i adalah faktor dari n, maka i dicetak. Setelah itu, fungsi memanggil dirinya sendiri dengan menambahkan 1 pada i untuk memeriksa faktor berikutnya. Proses ini berlanjut hingga semua faktor dari n tercetak. Hasilnya, ketika pengguna memasukkan bilangan seperti 5 atau 12, program akan menampilkan faktor-faktor yang relevan dengan bilangan tersebut.

4. 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.
   Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	5	5 4 3 2 1 2 3 4 5
2	9	9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9

Source Code:

```
package main

import "fmt"

func barisan(n int, current int) {
    if current < 1 {
```

```
        return
    }
    fmt.Print(current, " ")
    barisan(n, current-1)
    fmt.Print(current, " ")
}

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

    fmt.Print("Hasil: ")
    barisan(n, n)
    fmt.Println()
}
```

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambu
8_06\Modul5\Unguided4\unguided4.go"
Masukkan bilangan bulat positif N: 5
Hasil: 5 4 3 2 1 1 2 3 4 5
PS C:\Users\Asus\Documents\GeranadaSaputraPriambu
8_06\Modul5\Unguided4\unguided4.go"
Masukkan bilangan bulat positif N: 9
Hasil: 9 8 7 6 5 4 3 2 1 1 2 3 4 5 6 7 8 9
```

Penjelasan:

Program di atas merupakan program dalam bahasa Go dan bertujuan untuk menampilkan barisan bilangan dari suatu bilangan bulat positif N yang dimasukkan oleh pengguna, dimulai dari N hingga 1, lalu kembali ke N lagi. Pertama, pengguna diminta memasukkan nilai N. Program ini menggunakan fungsi rekursif barisan, yang menerima dua parameter: n (bilangan awal) dan current (bilangan saat ini yang sedang dicetak). Pada setiap panggilan fungsi, jika current masih lebih dari atau sama dengan 1, fungsi akan mencetak nilai current, kemudian memanggil dirinya sendiri dengan nilai current - 1 untuk melanjutkan ke angka berikutnya hingga mencapai 1. Setelah mencapai angka 1, fungsi kembali mencetak setiap nilai current saat keluar dari rekursi, sehingga terbentuk urutan naik kembali hingga N. Misalnya, jika pengguna memasukkan 5, program ini akan menghasilkan output 5 4 3 2 1 1 2 3 4 5.

5. 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.

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	5	1 3 5
2	20	1 3 5 7 9 11 13 15 17 19

Source Code:

```
package main

import "fmt"

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

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

    fmt.Print("Bilangan ganjil dari 1 hingga ", n, ": ")
    bilanganGanjil(n, 1)
    fmt.Println()
}
```

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008
8_06\Modul5\Unguided5\unguided5.go"
Masukkan bilangan bulat positif N: 5
Bilangan ganjil dari 1 hingga 5: 1 3 5
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008
8_06\Modul5\Unguided5\unguided5.go"
Masukkan bilangan bulat positif N: 20
Bilangan ganjil dari 1 hingga 20: 1 3 5 7 9 11 13 15 17 19
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_2311102008
```

Penjelasan:

Program di atas untuk menampilkan bilangan ganjil dari 1 hingga N menggunakan fungsi rekursif. Program pertama-tama meminta pengguna memasukkan sebuah bilangan bulat positif N. Kemudian, program memanggil fungsi bilanganGanjil yang menerima dua parameter: n, yaitu batas bilangan maksimum, dan current, yaitu bilangan ganjil yang dimulai dari 1. Di dalam fungsi ini, jika current lebih besar dari n, proses rekursi dihentikan. Namun, jika current masih dalam batas n, program mencetak nilai current, lalu memanggil kembali fungsi bilanganGanjil dengan nilai current ditambah 2, untuk melanjutkan ke bilangan ganjil berikutnya.

Proses ini berulang hingga mencapai batas n, sehingga semua bilangan ganjil dari 1 hingga N ditampilkan. Misalnya, jika pengguna memasukkan 5, hasilnya adalah "1 3 5"; jika N adalah 20, hasilnya adalah "1 3 5 7 9 11 13 15 17 19".

6. 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".

   Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	5	1 5
2	12	1 2 3 4 6 12

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.Print("Masukkan x: ")
    fmt.Scan(&x)
    fmt.Print("Masukkan y: ")
    fmt.Scan(&y)

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

Output:

```
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_231110200
8_06\Modul5\Unguided6\unguided6.go"
Masukkan x: 2
Masukkan y: 2
Hasil 2 pangkat 2 adalah: 4
PS C:\Users\Asus\Documents\GeranadaSaputraPriambudi_231110200
8_06\Modul5\Unguided6\unguided6.go"
Masukkan x: 5
Masukkan y: 3
Hasil 5 pangkat 3 adalah: 125
```

Penjelasan:

Program diatas menghitung hasil pemangkatan dari dua bilangan bulat x dan y dengan menggunakan fungsi rekursif. Pertama, pengguna diminta memasukkan nilai x (basis) dan y (pangkat). Fungsi pangkat(x, y) kemudian dijalankan untuk menghitung hasil x pangkat y. Di dalam fungsi rekursif ini, terdapat kondisi dasar di mana jika y bernilai 0, maka fungsi akan mengembalikan nilai 1 (karena bilangan apa pun dipangkatkan 0 adalah 1). Jika y lebih dari 0, fungsi akan mengalikan x dengan hasil dari pemanggilan fungsi pangkat(x, y-1), yang secara rekursif mengurangi nilai y hingga mencapai 0. Setelah perhitungan selesai, hasil pemangkatan ditampilkan ke pengguna.