

LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL V
REKURSIF



Oleh:

NAMA : HAIKAL SATRIATAMA

NIM : 2311102066

KELAS : IF – 11 – 02

S1 TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024

I. DASAR TEORI

A. Rekursi adalah konsep penting dalam pemrograman yang memungkinkan penyelesaian masalah dengan cara yang elegan dan terstruktur. Namun, penting untuk memahami bagaimana dan kapan menggunakannya untuk memaksimalkan efisiensi dan menghindari masalah seperti penggunaan memori yang berlebihan.

	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	

1. Kasus Basis:

- Setiap fungsi rekursif harus memiliki satu atau lebih kondisi yang menghentikan pemanggilan rekursif. Ini disebut "kasus basis". Tanpa kasus basis, fungsi akan terus memanggil dirinya sendiri hingga menyebabkan "stack overflow."

2. Panggilan Diri:

- Fungsi rekursif memanggil dirinya sendiri dengan argumen yang lebih sederhana atau lebih kecil. Ini memungkinkan fungsi untuk menyelesaikan bagian-bagian dari masalah yang lebih besar secara bertahap.

3. Struktur Kode:

- Kode rekursif seringkali lebih bersih dan lebih mudah dibaca dibandingkan dengan iterasi (loop) untuk masalah tertentu, terutama untuk masalah yang secara alami memiliki struktur pohon, seperti pencarian di pohon atau perhitungan deret Fibonacci.

4. Penggunaan Memori:

- Setiap pemanggilan fungsi rekursif memerlukan ruang di stack. Jika kedalaman rekursi terlalu besar, ini dapat menyebabkan masalah penggunaan memori, seperti "stack overflow".

5. Efisiensi:

- Rekursi dapat menjadi kurang efisien dibandingkan dengan pendekatan iteratif, terutama untuk masalah yang memerlukan banyak pemanggilan berulang untuk sub-masalah yang sama. Dalam beberapa kasus, teknik seperti memoization atau dynamic programming dapat digunakan untuk mengoptimalkan rekursi dengan menyimpan hasil dari sub-masalah yang sudah dihitung.

6. Contoh Umum Penggunaan:

- Rekursi sering digunakan dalam algoritma pencarian (seperti binary search), pengurutan (seperti quicksort dan mergesort), perhitungan matematika (seperti faktorial, deret Fibonacci), dan traversing struktur data seperti pohon dan graf.

7. Debugging:

- Debugging kode rekursif dapat menjadi lebih menantang dibandingkan dengan kode iteratif. Pemahaman yang baik tentang bagaimana fungsi berinteraksi dan kapan fungsi berhenti sangat penting untuk mengidentifikasi masalah.

8. Batasan Rekursi:

- Banyak bahasa pemrograman memiliki batasan pada kedalaman rekursi. Misalnya, dalam Python, ada batasan default pada kedalaman stack rekursi (misalnya, sekitar 1000 pemanggilan). Ini dapat diubah dengan fungsi `sys.setrecursionlimit()` tetapi harus dilakukan dengan hati-hati.

9. Alternatif Rekursi:

- Dalam beberapa kasus, masalah yang dapat diselesaikan dengan rekursi juga dapat diselesaikan dengan pendekatan iteratif. Penting untuk mempertimbangkan konteks dan memilih pendekatan yang paling sesuai untuk kebutuhan spesifik.

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 P bertambah 1 (Increment by one) secara terus menerus tanpa henti.

II. GUIDED

NO 1.

Source code

```
package main

import "fmt"

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

func printDescending(n int) {
    if n <= 0 {
        return
    }
    fmt.Println(n)
    printDescending(n - 1)
}
```

Output :

```
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\GUIDED 1\Guided1.go"
5
5
4
3
2
1
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> █
```

Penjelasan :

Program ini secara efektif menghitung mundur dari nilai input, mencetak setiap angka pada baris baru, hingga mencapai angka 1. Jika inputnya nol atau negatif, program tidak menghasilkan output apa pun.

NO 2.

Source code

```
package main

import "fmt"

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

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

```
}  
}
```

Output :

```
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\GUIDED 2\Guided2.go"  
10  
55  
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> █
```

Penjelasan :

Program Go yang disediakan menghitung jumlah semua bilangan bulat dari 1 hingga bilangan bulat tertentu n menggunakan rekursif Program ini secara efektif menangani masukan non-positif dengan mengembalikan nilai 0 ketika n kurang dari atau sama dengan 0.

III. UNGUIDED

NO 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() {  
    fmt.Println("n: ")  
    for i := 0; i <= 10; i++ {  
        fmt.Printf("%d ", i)  
    }  
  
    fmt.Println() // New line  
    fmt.Print("Sn: ")
```

```

    for i := 0; i <= 10; i++ {
        fmt.Printf("%d ", fibonacci(i))
    }

    fmt.Println()
}

```

Output :

```

PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 1\Unguided1.go"
n: 0 1 2 3 4 5 6 7 8 9 10
Sn: 0 1 1 2 3 5 8 13 21 34 55
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5>

```

Penjelasan :

Program Go yang disediakan menghitung dan mencetak deret Fibonacci untuk 10 angka pertama (dari 0 hingga 10) menggunakan fungsi rekursif. Program ini menampilkan indeks beserta angka Fibonacci yang sesuai dalam format yang jelas. Program ini menyortir konsep-konsep utama seperti rekursi, output yang diformat, dan struktur kontrol.

NO 2.

Source code

```

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
    for {
        fmt.Print("Masukkan angka: ")
        fmt.Scan(&N)
        if N == 0 {
            fmt.Println("Program selesai.")
            break
        }
        printStars(N)
    }
}

```

Output :

```

PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 2\Unguided2.go"
Masukkan angka: 5
*
**
***
****
*****
Masukkan angka: 1
*
Masukkan angka: 3
*
**
***

```

Penjelasan :

Program ini menggunakan rekursi untuk mencetak pola bintang berdasarkan input dari pengguna. Dengan memanfaatkan struktur kontrol seperti loop dan kondisi, program ini menyediakan interaksi yang jelas dengan pengguna dan menghentikan eksekusi berdasarkan input yang diberikan.

NO 3.

Source code

```

package main

```



```

import "fmt"

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

func main() {
    var N int
    fmt.Print("Masukkan angka N: ")
    fmt.Scan(&N)
    fmt.Print("Faktor dari ", N, ": ")
    findFactors(N, 1)
    fmt.Println()
}

```

Output :

```

PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 3\Unguided3.go"
Masukkan angka N: 5
Faktor dari 5: 1 5
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 3\Unguided3.go"
Masukkan angka N: 12
Faktor dari 12: 1 2 3 4 6 12
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> █

```

Penjelasan :

Program ini meminta pengguna untuk memasukkan sebuah angka, kemudian mencetak semua faktor dari angka tersebut.

NO 4.

Source code

```

package main

```

```

import "fmt"

func printSequence(n, current int) {
    if current > n {
        return
    }
    fmt.Print(current, " ")
    printSequence(n, current+1)
    if current < n {
        fmt.Print(current, " ")
    }
}

func main() {
    var N int
    fmt.Print("Masukkan angka: ")
    fmt.Scan(&N)
    printSequence(N, 1)
    fmt.Println()
}

```

Output :

```

PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 4\Unguided4.go"
Masukkan angka: 7
1 2 3 4 5 6 7 6 5 4 3 2 1
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 4\Unguided4.go"
Masukkan angka: 10
1 2 3 4 5 6 7 8 9 10 9 8 7 6 5 4 3 2 1
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> █

```

Penjelasan :

Program ini mendefinisikan sebuah fungsi rekursif bernama `printSequence` yang mencetak urutan angka dari 1 hingga `n` dan kemudian mencetak urutan tersebut secara terbalik (dari `n-1` hingga 1). Fungsi ini dipanggil dalam fungsi `main` setelah pengguna memasukkan angka `N`.

NO 5.

Source code

```

package main

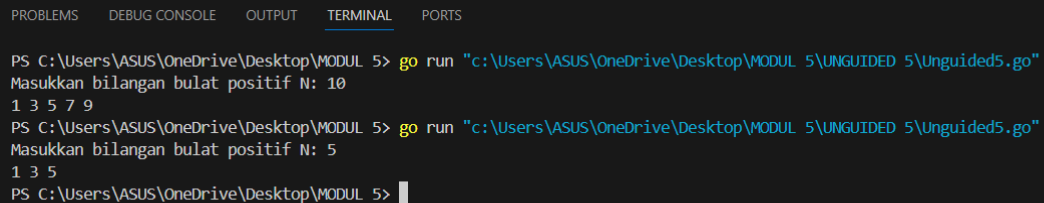
import "fmt"

func tampilkanGanjil(n int) {
    for i := 1; i <= n; i += 2 {
        fmt.Print(i, " ")
    }
}

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

```

Output :



```

PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 5\Unguided5.go"
Masukkan bilangan bulat positif N: 10
1 3 5 7 9
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 5\Unguided5.go"
Masukkan bilangan bulat positif N: 5
1 3 5
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5>

```

Penjelasan :

Program ini merupakan contoh sederhana dalam bahasa pemrograman Go yang berfungsi untuk mencetak semua bilangan ganjil dari 1 hingga bilangan bulat positif yang dimasukkan oleh pengguna. Dengan menggunakan fungsi dan loop, program ini menunjukkan cara dasar untuk mengelola input dan menghasilkan output.

NO 6.

Source code

```

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)
    fmt.Printf("Hasil dari %d pangkat %d adalah %d\n", x, y, pangkat(x,
        y))
}

```

Output :

```

PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 6\Unguided6.go"
Masukkan nilai x: 5
Masukkan nilai y: 2
Hasil dari 5 pangkat 2 adalah 25
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> go run "c:\Users\ASUS\OneDrive\Desktop\MODUL 5\UNGUIDED 6\Unguided6.go"
Masukkan nilai x: 10
Masukkan nilai y: 2
Hasil dari 10 pangkat 2 adalah 100
PS C:\Users\ASUS\OneDrive\Desktop\MODUL 5> █

```

Penjelasan :

Program ini berfungsi untuk menghitung nilai pangkat dari bilangan bulat x dipangkatkan dengan y (yaitu x^y). Program ini menggunakan pendekatan rekursif untuk menghitung pangkat dan berinteraksi dengan pengguna melalui input dan output.