

**LAPORAN PRAKTIKUM PEMROGRAMAN
BERORIENTASI OBJEK**

MODUL V

REKURSIF



Oleh:

NAMA : ARNANDA SETYA NOSA PUTRA

NIM : 2311102180

KELAS : IF 11 02

S1 TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

I. DASAR TEORI

Rekursif adalah teknik di mana sebuah fungsi memanggil dirinya sendiri untuk menyelesaikan masalah. Fungsi rekursif biasanya terdiri dari dua bagian utama: **basis kasus** (base case), yang menentukan kapan fungsi harus berhenti memanggil dirinya sendiri, dan **rekursi** (recursive case), yang memecah masalah menjadi sub-masalah yang lebih kecil dan memanggil fungsi tersebut kembali. Rekursi sangat berguna untuk masalah yang memiliki sifat berulang atau dapat dipecah menjadi versi yang lebih sederhana dari masalah yang sama, seperti perhitungan faktorial, bilangan Fibonacci, atau traversing struktur data seperti pohon atau graf.

	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	

Catatan:

1. Teknik rekursif merupakan salah satu alternatif untuk menggantikan struktur kontrol perulangan dengan memanfaatkan subprogram, baik itu fungsi maupun prosedur.
2. Untuk menghentikan proses rekursif, digunakan struktur percabangan (if-then).
3. Base-case adalah kondisi di mana proses rekursif berhenti. Base-case adalah elemen paling penting dan harus diketahui terlebih dahulu ketika membuat program rekursif. Membuat program rekursif tanpa mengetahui base-case terlebih dahulu tidak mungkin dilakukan.
4. Recursive-case adalah kondisi di mana fungsi memanggil dirinya sendiri. Recursive-case merupakan kebalikan atau negasi dari base-case.
5. Setiap algoritma rekursif selalu memiliki bentuk setara dalam algoritma iteratif.

Komponen Rekursif:

Algoritma rekursif terdiri dari dua komponen utama:

- **Base-case (Basis):** Bagian yang menghentikan proses rekursif, yang menjadi komponen paling penting dalam rekursi.
- **Recursive-case:** Bagian di mana subprogram dipanggil kembali

II. GUIDED

Source Code + Screenshot hasil program beserta penjelasan

SOAL

NO 1.

Source code:

```
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)
    }
}
```

Output :

```
PS C:\Users\HP> go run "d:\PRAKTIKUM ALPR
5
5
4
3
2
1
PS C:\Users\HP> █
```

Penjelasan :

Program di atas meminta input berupa bilangan bulat `n` dari pengguna, lalu mencetak bilangan tersebut secara menurun hingga mencapai 1 menggunakan fungsi rekursif bernama `baris`. Fungsi `baris(bilangan int)` akan mencetak nilai `bilangan`, lalu memanggil dirinya sendiri dengan parameter `bilangan - 1`. Proses rekursi ini akan berakhir saat `bilangan` bernilai 1, di mana angka 1 dicetak, dan rekursi berhenti. Program ini mendemonstrasikan penggunaan rekursi untuk mencetak bilangan secara menurun.

NO 2.

Source code:

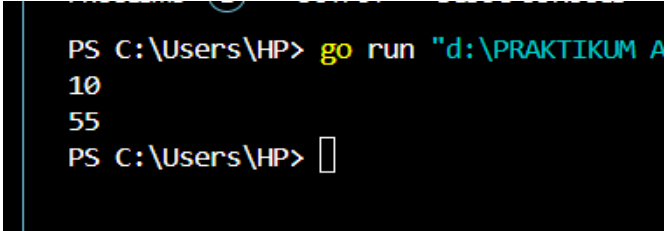
```
package main

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)
    }
}
```

Output :



```
PS C:\Users\HP> go run "d:\PRAKTIKUM A
10
55
PS C:\Users\HP> 
```

Penjelasan :

Program di atas menghitung penjumlahan dari 1 hingga `n` menggunakan rekursi. Pengguna memasukkan nilai `n`, kemudian fungsi rekursif `penjumlahan(n)` akan menjumlahkan nilai `n` dengan hasil penjumlahan dari `n-1` hingga mencapai nilai 1, yang menjadi kondisi dasar. Hasilnya dicetak sebagai output.

III. UNGUIDED

Source Code + Screenshot hasil program beserta penjelasan

SOAL 1

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 $S_n = S_{n-1} + S_{n-2}$. Berikut ini adalah contoh nilai deret fibonacci hingga suku ke-10. Buatlah program yang mengimplementasikan fungsi rekursif pada deret fibonacci tersebut.

Source code :

```
1  package main
2
3  import "fmt"
4
5  func main() {
6      var n int
7      fmt.Print("Masukkan nilai n: ")
8      fmt.Scan(&n)
9      fibonacciSeries(n)
10 }
11
12 func fibonacciSeries(n int) {
13     for i := 0; i <= n; i++ {
14         fmt.Print(fibonacci(i), " ")
15     }
16     fmt.Println()
17 }
18
19 func fibonacci(n int) int {
20     if n <= 1 {
21         return n
22     }
23     return fibonacci(n-1) + fibonacci(n-2)
24 }
25
```

Output :

```
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\go-build1811111111\main.go"
Masukkan nilai n: 10
0 1 1 2 3 5 8 13 21 34 55
PS C:\Users\HP>
```

Penjelasan :

Program di atas menerima input dari pengguna berupa bilangan bulat `n` dan menampilkan deret Fibonacci hingga elemen ke-`n`. Fungsi `fibonacciSeries(n)` mencetak setiap angka Fibonacci dari 0 hingga `n`, dengan memanggil fungsi rekursif `fibonacci(n)` yang menghitung nilai Fibonacci menggunakan rumus dasar: jika `n` adalah 0 atau 1, hasilnya adalah `n`; untuk nilai lebih besar, hasilnya adalah jumlah dari dua angka Fibonacci sebelumnya (`fibonacci(n-1) + fibonacci(n-2)`).

SOAL 2

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

Source code :

```
1  package main
2
3  import "fmt"
4
5  func main() {
6      var n int
7      fmt.Print("Masukkan nilai N: ")
8      fmt.Scan(&n)
9      fmt.Println(printbintang(n))
10 }
11
12 func printbintang(n int) string {
13     if n == 1 {
14         return "*"
15     } else {
16         return printbintang(n-1) + "\n" + cetakBintang(n)
17     }
18 }
19
20 func cetakBintang(n int) string {
21     bintang := ""
22     for i := 0; i < n; i++ {
23         bintang += "*"
24     }
25     return bintang
26 }
27
```

Output :

```
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\go-build1511111111\main.go"
Masukkan nilai N: 5
*
**
***
****
*****
PS C:\Users\HP> 
```

Penjelasan :

Program di atas bertujuan untuk mencetak pola bintang secara bertingkat berdasarkan input pengguna. Pengguna diminta untuk memasukkan nilai `n`, kemudian program menggunakan fungsi rekursif `printbintang(n)` untuk menghasilkan pola bintang. Fungsi `printbintang(n)` memanggil dirinya sendiri untuk membentuk baris bintang sebelumnya, lalu menambahkan baris bintang dengan panjang sesuai nilai `n` yang dihasilkan oleh fungsi `cetakBintang(n)`. Fungsi ini terus berulang hingga mencapai nilai dasar `n == 1`, di mana ia mengembalikan satu bintang.

SOAL 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).

Source code :

```
1 package main
2
3 import "fmt"
4
5 func main() {
6
7     var n int
8     fmt.Scan(&n)
9     fmt.Println("Faktor dari", n, "adalah:")
10    cetakFaktor(n, 1)
11 }
12
13 func cetakFaktor(n, i int) {
14     if i > n {
15         return
16     }
17
18     if n%i == 0 {
19         fmt.Print(i, " ")
20     }
21
22     cetakFaktor(n, i+1)
23 }
```

Output :

```
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\temp
12
Faktor dari 12 adalah:
1 2 3 4 6 12
PS C:\Users\HP> 
```

Penjelasan :

Program di atas mencari dan mencetak faktor-faktor dari bilangan bulat `n` yang dimasukkan oleh pengguna. Fungsi utama `main` menerima input `n` dan memanggil fungsi rekursif `cetakFaktor`, yang secara berulang memeriksa apakah bilangan `i` (dimulai dari 1) adalah faktor dari `n` (jika `n % i == 0`). Jika ya, `i` dicetak. Proses ini berlanjut hingga semua faktor dari 1 hingga `n` ditemukan, kemudian program berhenti.

SOAL 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

Source code :

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var n int
7     fmt.Scan(&n)
8     baris(n)
9     fmt.Println()
10 }
11
12 func baris(bilangan int) {
13     if bilangan == 1 {
14         fmt.Print(1, " ")
15     } else {
16         fmt.Print(bilangan, " ")
17         baris(bilangan - 1)
18         fmt.Print(bilangan, " ")
19     }
20 }
```

Output :

```
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\tempCodeRunnerFile.go"
5
5 4 3 2 1 2 3 4 5
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\tempCodeRunnerFile.go"
9
9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9
PS C:\Users\HP> 
```

Penjelasan :

Program di atas menggunakan rekursi untuk mencetak sebuah pola baris angka berdasarkan input pengguna. Fungsi `baris(bilangan int)` mencetak bilangan secara berurutan dari bilangan input hingga 1, lalu kembali naik ke bilangan input tersebut. Saat nilai bilangan mencapai 1, program mencetak 1 dan mulai kembali mencetak angka yang lebih besar hingga mencapai bilangan input awal. Contohnya, jika pengguna memasukkan angka 3, outputnya akan menjadi: `3 2 1 2 3`.

SOAL 5

Source code :

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var n int
7     fmt.Scan(&n)
8     barisGanjil(n)
9     fmt.Println()
10 }
11
12 func barisGanjil(bilangan int) {
13     if bilangan <= 0 {
14         return
15     }
16
17     barisGanjil(bilangan - 1)
18
19     if bilangan%2 != 0 {
20         fmt.Print(bilangan, " ")
21     }
22 }
```

Output :

```
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\tempC
5
1 3 5
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\tempC
20
1 3 5 7 9 11 13 15 17 19
PS C:\Users\HP> 
```

Penjelasan :

Program di atas merupakan program untuk mencetak bilangan ganjil dari `1` hingga `n` menggunakan pendekatan rekursif. Pertama, pengguna diminta untuk memasukkan nilai `n`. Fungsi `barisGanjil` kemudian dipanggil dengan `n` sebagai argumen. Di dalam fungsi tersebut, jika nilai `bilangan` kurang dari atau sama dengan `0`, fungsi akan berhenti. Jika tidak, fungsi memanggil dirinya sendiri dengan argumen `bilangan - 1`, sehingga menciptakan tumpukan rekursi. Setelah kembali dari pemanggilan rekursif, program memeriksa apakah `bilangan` adalah bilangan ganjil (jika `bilangan % 2 != 0`). Jika benar, bilangan tersebut akan dicetak ke layar. Dengan demikian, bilangan ganjil dari `1` hingga `n` dicetak dalam urutan meningkat.

SOAL 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".

Source code :

```
1  package main
2
3  import "fmt"
4
5  func main() {
6
7      var x, y int
8      fmt.Scan(&x)
9      fmt.Scan(&y)
10
11     fmt.Println(pangkat(x, y))
12 }
13
14 func pangkat(x, y int) int {
15     if y == 0 {
16         return 1
17     } else {
18         return x * pangkat(x, y-1)
19     }
20 }
```

Output :

```
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\tempCodeRu
2 2
4
PS C:\Users\HP> go run "C:\Users\HP\AppData\Local\Temp\tempCodeRu
5 3
125
PS C:\Users\HP> 
```

Penjelasan :

Program di atas merupakan implementasi sederhana dari fungsi rekursif untuk menghitung hasil dari bilangan `x` dipangkatkan dengan `y`. Di dalam fungsi `main`, program meminta pengguna untuk memasukkan dua bilangan bulat, `x` dan `y`. Setelah menerima input, program memanggil fungsi `pangkat`, yang melakukan perhitungan pangkat secara rekursif: jika `y` sama dengan 0, maka fungsi mengembalikan 1 (karena setiap bilangan dipangkatkan 0 adalah 1). Jika tidak, fungsi mengalikan `x` dengan hasil pemanggilan `pangkat(x, y-1)` hingga mencapai kondisi dasar. Akhirnya, hasil dari pangkat tersebut dicetak ke layar.