LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK

MODUL V
REKURSIF

Oleh:

NAMA : KARTIKA PRINGGO HUTOMO

NIM : 2311102196

KELAS ; IF-11-02

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

Rekursi adalah sebuah konsep dalam matematika dan ilmu komputer yang memungkinkan suatu fungsi atau prosedur memanggil dirinya sendiri secara terus menerus. Fungsi atau prosedur yang memanggil dirinya sendiri disebut fungsi atau prosedur rekursif.

Fitur utama dari panggilan mandiri rekursif:
Poin utama dari rekursi adalah bahwa suatu fungsi atau prosedur memanggil dirinya sendiri dengan parameter yang berbeda. Hal ini memungkinkan Anda untuk membagi solusi dari suatu masalah yang kompleks menjadi sub-masalah yang lebih kecil dengan jenis yang sama.
Kasus dasar: Fungsi rekursif memerlukan kasus dasar, suatu kondisi terminasi yang menghentikan proses rekursif. Kasus dasar ini sangat penting. Hal ini karena tanpa kondisi dasar, fungsi tersebut akan memanggil dirinya sendiri tanpa henti, sehingga mengakibatkan kesalahan stack overflow.
Rekursi tail: Beberapa fungsi rekursif dapat diklasifikasikan sebagai rekursif tail.
Dalam hal ini, pernyataan terakhir yang dieksekusi berada di dalam isi fungsi, dan hasil yang dikembalikan oleh fungsi tersebut bukan bagian dari fungsi tersebut. Fungsi rekursif ekor biasanya tidak melakukan aktivitas selama fase inversi, sehingga kompiler modern dapat secara otomatis membuat kode untuk fase ini.

Kelebihan dan Kekurangan Rekursi Rekursi memiliki beberapa kelebihan: Lebih pendek dan mudah dipahami: Kode rekursif cenderung lebih pendek dan mudah dipahami karena konstruksi pemrograman yang sistematis.
Efektif dalam memecahkan masalah yang kompleks: Rekursif dapat menyelesaikan masalah yang kompleks dengan lebih efektif.
Namun, rekursi juga memiliki beberapa kelemahan: Memori besar: Menggunakan rekursi dapat meningkatkan penggunaan memori karena setiap pemanggilan fungsi memerlukan ruang pada tumpukan.
Kemungkinan kesalahan stack overflow: Kesalahan stack overflow dapat terjadi jika proses rekursif tidak dibatasi.
Dengan memahami karakteristik dan contoh implementasi rekursif, pemrogram dapat menerapkannya secara bermakna untuk memecahkan berbagai jenis masalah dalam programnya.

II. GUIDED

Guided 1

Source code

```go
package main

import "fmt"
```

```go
func main() {
    var n int
    fmt.Scan(&n) //membaca input pengguna
    baris(n)     //memanggil fungsi rekursif 'baris'
}
```

```go
func baris(bilangan int) {
    if bilangan == 1 { //base case: jika bilngan sama dengan
        fmt.Println(1)
    } else {
        fmt.Println(bilangan)
        baris(bilangan - 1)
    }
}
```

Output :

```
PS D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5> go run "d:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5\guided1.go"
4
4
3
2
1
PS D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5>
```

Penjelasan

Program ini mencetak angka dari n sampai 1 secara berurutan menggunakan **rekursi** (fungsi yang memanggil dirinya sendiri).

1. Program meminta input dari pengguna (n).
2. Fungsi baris dipanggil, yang:
    1. Jika n = 1, mencetak 1 dan berhenti.
    2. Jika n > 1, mencetak angka n, lalu memanggil dirinya sendiri dengan n-1, hingga mencapai 1.

Guided 2

Source code

```go
package main

import "fmt"
```

```go
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 D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5> go run "d:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5\guided2.go"
5
15
PS D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5>
```

Penjelasan

Program ini melakukan penjumlahan dari 1 hingga angka n menggunakan rekursi. Mari kita lihat penjelasannya:
Fungsi main:
Program dimulai dengan deklarasi variabel n bertipe int.
Fungsi fmt.Scan(&n) digunakan untuk menerima input dari pengguna, yang menyimpan angka ke dalam variabel n.
Fungsi penjumlahan(n) dipanggil, yang menghitung total penjumlahan dari 1 hingga n, lalu hasilnya dicetak menggunakan fmt.Println.
Fungsi penjumlahan:
Fungsi ini adalah fungsi rekursif yang menghitung penjumlahan angka dari 1 sampai n.
Base Case: Jika n == 1, maka fungsi mengembalikan nilai 1 (karena penjumlahan hingga 1 adalah 1).
Recursive Case: Jika n > 1, maka fungsi akan mengembalikan n ditambah hasil dari penjumlahan(n-1)

III. UNGUIDED

1.

Source code

```
package main

import "fmt"

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

func fibonacci(n int) int {
    if n == 0 {
        return 0
    }
    if n == 1 {
        return 1
    }
 a, b := 0, 1
    for i := 2; i <= n; i++ {
        a, b = b, a+b
    }
    return b
}
```

Output

```
PS D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5> go run "d:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5\unguided1.go"
Masukkan nilai n: 10
0 1 1 2 3 5 8 13 21 34 55
PS D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5>
```

Penjelasan

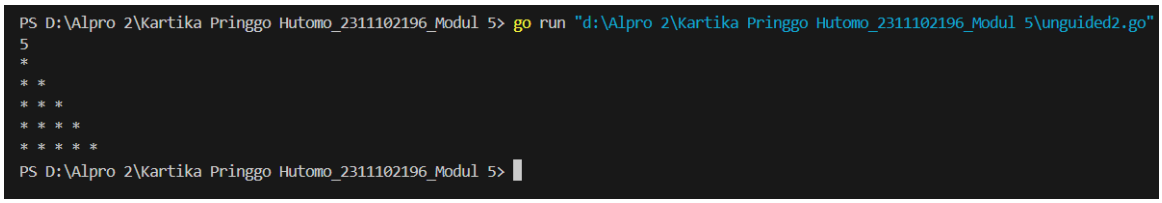
Program ini menghasilkan dan mengeluarkan deret Fibonacci dari nilai yang dimasukkan oleh pengguna dari 0 hingga n, menggunakan pendekatan berulang untuk efisiensi. Cara kerja: Input pengguna: Program meminta pengguna memasukkan nilai n sebagai batas atas deret Fibonacci. Perhitungan Fibonacci : Fungsi Fibonacci menggunakan dua variabel (a dan b) untuk menyimpan dua nilai Fibonacci sebelumnya. Awalnya, a dimulai pada 0 (Fibonacci ke-0) dan b dimulai pada 1 (Fibonacci ke-1). iterasi: Untuk setiap angka dari 2 hingga n, program memperbarui nilai a dan b. dimana a adalah nilai Fibonacci sebelumnya dan b adalah jumlah dari (n-1) dan (n-2). Nilai Fibonacci. Output: Program mencetak deret Fibonacci dari 0 hingga n. Keuntungan: Efisiensi: Pendekatan berulang ini jauh lebih efisien dibandingkan rekursi karena menghindari pemanggilan berulang ke fungsi tersebut. Hal ini membuat eksekusi menjadi lambat, terutama untuk nilai yang besar. n. Output yang sama: Pendekatannya berbeda dengan rekursi, namun hasil deret Fibonacci yang dihasilkan sama.

2.

Source code

```
package main

import (
	"fmt"
)

func cetakBintang(n int) {
	if n == 0 {
		return
	}
	fmt.Print("* ")
	cetakBintang(n - 1)
}

func polaBintang(n int) {
	for i := 1; i <= n; i++ {
		cetakBintang(i)
		fmt.Println()
	}
}

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

Output



Penjelasan

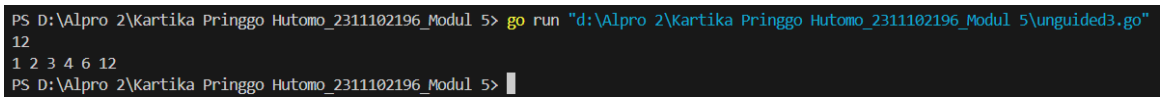
Program ini mencetak pola bintang bersarang dimana jumlah bintang di setiap baris bertambah dari 1 menjadi n. Pola ini dihasilkan menggunakan dua fungsi . Satu fungsi mencetak garis bintang, dan fungsi lainnya mengatur jumlah garis yang akan dicetak. Cara kerja program: Input pengguna: Program meminta pengguna memasukkan nilai n yang menunjukkan jumlah baris bintang yang akan dicetak. Fungsi printStars(n int): Fungsi ini mencetak n bintang (*) secara rekursif pada satu baris. Jika n == 0, fungsi berhenti (berbasis rekursi). Jika n > 0, fungsi akan mengembalikan tanda bintang dan memanggil dirinya sendiri pada n-1. Fungsi patternStar(n int): Fungsi ini menggunakan perulangan for untuk mengatur jumlah baris yang akan dicetak. Pada setiap iterasi, fungsi printStar dipanggil dengan jumlah bintang bertambah dari 1 menjadi n, lalu fmt.Println() dipanggil untuk mencetak baris baru. Program utama: Memanggil fungsi patternStar dengan input pengguna untuk mencetak pola.

3.

Sourcode

```
package main

import "fmt"

func rekursif(n int) {
	for i := 1; i <= n; i++ {
		if n%i == 0 {
			fmt.Print(i, " ")
		}
	}
}


func main() {
	var a int
	fmt.Scan(&a)
	rekursif(a)
}
```

Output



Penjelasan

Program ini dirancang untuk mencetak semua faktor bilangan bulat positif n yang dimasukkan pengguna. Faktor adalah bilangan bulat yang dapat membagi n tanpa sisa. Perilaku Program: Input Pengguna: Program meminta pengguna memasukkan bilangan bulat positif (a) menggunakan input standar. Fungsi rekursif (n int): Fungsi ini melakukan iterasi dari 1 ke n menggunakan perulangan for. Pada setiap iterasi, fungsi memeriksa apakah i (nilai iterasi saat ini) merupakan faktor n dengan memeriksa apakah n % i == 0. Jika benar, saya akan dikeluarkan sebagai salah satu elemen. Program utama: Fungsi rekursif dipanggil menggunakan nilai n yang dimasukkan oleh pengguna dan semua elemen dikeluarkan.

4.

Sorce code

```
package main

import (
	"fmt"
)


func printSequence(n int) {
	for current := 1; current <= n; current++ {
		fmt.Print(current, " ")
	}
	for current := n - 1; current >= 1; current-- {
		fmt.Print(current, " ")
	}
}
func main() {
	var N int


	fmt.Print("Masukkan angka: ")
	fmt.Scan(&N)


	printSequence(N)
	fmt.Println()
}
```

Output

```
PS D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5> go run "d:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5\unguided4.go"
Masukkan angka: 9
1 2 3 4 5 6 7 8 9 8 7 6 5 4 3 2 1
PS D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5>
```

Penjelasan

Program mencetak rangkaian angka dari 1 sampai n, lalu dari n-1 kembali ke 1. Dimana n adalah bilangan bulat positif yang dimasukkan oleh pengguna. Dengan cara ini, program menciptakan pola numerik simetris. Perilaku Program: Input Pengguna: Program meminta pengguna memasukkan bilangan bulat positif (N) menggunakan input standar. Fungsi printSequence(n int): Perulangan pertama: Menggunakan perulangan for, fungsi ini mencetak angka 1 hingga n diikuti dengan spasi. Perulangan kedua: Setelah mencetak angka hingga n, fungsi menjalankan perulangan kedua dan mencetak angka n-1 hingga 1, termasuk spasi. Ini menciptakan efek simetris. Program utama: Setelah pengguna memasukkan nilai N, fungsi printSequence dipanggil untuk mencetak urutan angka yang diperlukan.

5.

Source code

```
package main

import "fmt"

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


func barisGanjil(bilangan int) {
	for i := 1; i <= bilangan; i++ {
		if i%2 != 0 {
			fmt.Print(i, " ")
		}
	}
}
```

Output

```
PS D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5> go run "d:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5\unguided5.go"
20
1 3 5 7 9 11 13 15 17 19
PS D:\Alpro 2\Kartika Pringgo Hutomo_2311102196_Modul 5>
```

Penjelasan

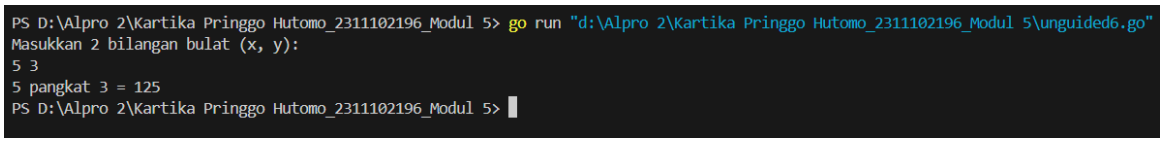
Program ini dirancang untuk mencetak semua bilangan ganjil dari 1 sampai n. n adalah bilangan bulat positif yang dimasukkan oleh pengguna. Program ini menggunakan pendekatan berulang untuk mencapai hasil yang diinginkan. Cara kerja program: Input pengguna: Program meminta pengguna memasukkan bilangan bulat positif (n) menggunakan input standar. Fungsi Baris Ganjil (angka int): For Loop: Fungsi ini menggunakan perulangan dari 1 ke angka, memeriksa setiap angka. Periksa bilangan ganjil: Di dalam perulangan, fungsi memeriksa apakah bilangan saat ini (i) ganjil dengan kondisi i%2 ! = 0. Jika kondisi ini terpenuhi, nomor akan dicetak. Program utama: Ketika pengguna memasukkan nilai n, fungsi baris ganjil dipanggil dan semua bilangan ganjil terkait dicetak.

6.

Source code

```
package main

import "fmt"

func power(x, y int) int {
	result := 1
	for i := 0; i < y; i++ {
		result *= x
	}
	return result
}

func main() {
	fmt.Println("Masukkan 2 bilangan bulat (x, y):")
	var x, y int
	fmt.Scanln(&x, &y)
	result := power(x, y)
	fmt.Printf("%d pangkat %d = %d\n", x, y, result)
}
```

Output



Penjelasan

Program ini dirancang untuk menghitung dan mencetak hasil penjumlahan bilangan bulat x menjadi bilangan bulat y. dimana x dan y adalah nilai yang dimasukkan oleh pengguna. Cara kerja program: Input pengguna: Program meminta pengguna memasukkan dua bilangan bulat (x dan y) menggunakan input standar. Fungsi power(x, y int) int: Fungsi ini menghitung x pangkat y menggunakan pendekatan berulang. Inisialisasi variabel: Variabel hasil diinisialisasi dengan nilai 1. Perulangan For: Perulangan dimulai dari 0 hingga y-1. Pada setiap iterasi, nilai yang dihasilkan dikalikan dengan x, yang secara bertahap meningkatkan nilai kinerja. Hasil yang dikembalikan: Saat perulangan selesai, fungsi mengembalikan nilai hasil yang merupakan hasil x yang dipangkatkan y. Program utama: Ketika pengguna memasukkan nilai x dan y, fungsi eksponensial dipanggil dan hasilnya ditampilkan dalam format yang jelas.