

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL V
PROSEDUR**



Disusun Oleh :

Muhammad Ihab Aufa Rafi / 2311102226

S1IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi, S.Kom., M.Kom.

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

1.1 Definisi Rekursif

Pada modul-modul sebelumnya sudah dijelaskan bahwa suatu subprogram baik fungsi atau prosedur bisa memanggil subprogram lainnya. Hal ini tidak menutup kemungkinan bahwa subprogram yang dipanggil adalah dirinya sendiri. Dalam pemrograman teknik ini dikenal dengan istilah rekursif.

Rekursif secara sederhana dapat diartikan sebagai cara menyelesaikan suatu masalah dengan cara menyelesaikan sub-masalah yang identik dari masalah utama. Sebagai contoh perhatikan prosedur cetak berikut ini!

	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	

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

```
1 package main
2 import "fmt"
3 func main(){
4     cetak(5)
5 }
6 func cetak(x int){
7     fmt.Println(x)
8     cetak(x+1)
9 }
```

Oleh karena itu bisanya ditambahkan struktur kontrol percabangan (If-then) untuk menghentikan proses rekursif ini. Kondisi ini disebut juga dengan base-case, artinya apabila kondisi base-case bernilai true maka proses rekursif akan berhenti. Sebagai contoh misalnya base case adalah ketika **x** bernilai 10 atau **x == 10**, maka tidak perlu dilakukan rekursif.

```
1 procedure cetak(in x:integer)
2 algoritma
3   if x == 10 then
4     output(x)
5   else
6     output(x)
7     cetak(x+1)
8   endif
9 endprocedure
```

1.2 Komponen Rekursif

Algoritma rekursif terdiri dari dua komponen utama:

1. Base-case (Basis), yaitu bagian untuk menghentikan proses rekursif dan menjadi komponen terpenting di dalam sebuah rekursif.
2. Recursive-case, yaitu bagian pemanggilan subprogramnya.

1.3 Contoh Program dengan menggunakan Rekursif

Contoh program dengan fungsi rekursif ada pada bagian Guided.

II. GUIDED

1. Soal Studi Case

Membuat baris bilangan dari 1 hingga n.

Sourcecode

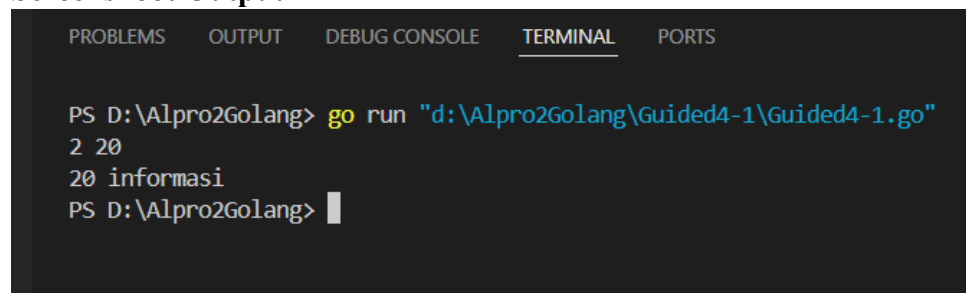
```
package main

import "fmt"

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)
}
```

Screenshoot Output



```
PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Guided4-1\Guided4-1.go"
20
20 informasi
PS D:\Alpro2Golang>
```

Deskripsi Program

Program ini adalah implementasi rekursif untuk mencetak bilangan secara mundur dari n hingga 1. Di awal program, kita mendefinisikan sebuah fungsi bernama **cetakMundur** yang menerima parameter n bertipe integer. Fungsi ini menggunakan konsep rekursi dimana base case (kasus dasar) terjadi ketika n mencapai nilai 1. Pada kondisi tersebut, program akan

mencetak angka 1 dan mengakhiri rekursi dengan perintah return. Jika nilai n belum mencapai 1, maka program akan mencetak nilai n saat ini diikuti dengan spasi, kemudian memanggil dirinya sendiri (rekursi) dengan parameter n-1. Proses ini akan terus berulang hingga mencapai base case. Penggunaan `fmt.Print` untuk nilai selain 1 dan `fmt.Println` untuk nilai 1 membuat output tercetak dalam satu baris dengan angka terakhir (1) diikuti baris baru.

2. Soal Studi Case

Menghitung hasil penjumlahan 1 hingga n.

Sourcecode

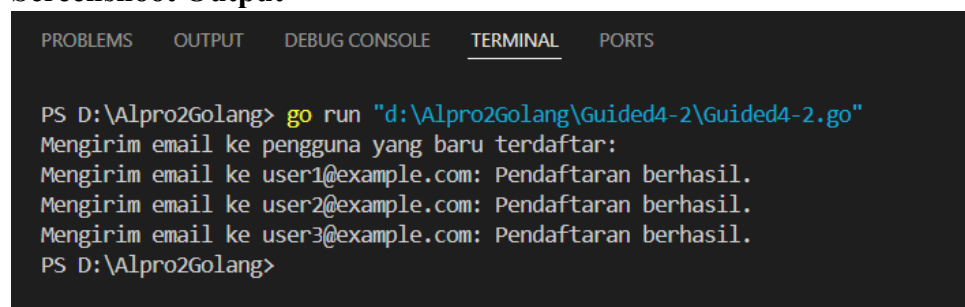
```
package main

import "fmt"

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

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

Screenshoot Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Guided4-2\Guided4-2.go"
Mengirim email ke pengguna yang baru terdaftar:
Mengirim email ke user1@example.com: Pendaftaran berhasil.
Mengirim email ke user2@example.com: Pendaftaran berhasil.
Mengirim email ke user3@example.com: Pendaftaran berhasil.
PS D:\Alpro2Golang>
```

Deskripsi Program

Program ini merupakan implementasi rekursif untuk menghitung jumlah deret bilangan dari 1 hingga n. Program menggunakan fungsi **jumlahRekursif** yang menerima parameter n bertipe integer. Fungsi ini menerapkan konsep rekursi dengan base case yang terjadi ketika n sama dengan 1, dimana fungsi akan langsung mengembalikan nilai 1 sebagai hasil akhir dari rekursi.

Ketika nilai n lebih besar dari 1, fungsi akan melakukan proses rekursif dengan mengembalikan nilai n ditambah dengan hasil pemanggilan fungsi **jumlahRekursif** dengan parameter n-1. Proses ini akan terus berlanjut hingga mencapai base case. Sebagai contoh, jika n=4, maka proses perhitungannya akan menjadi $4 + (3 + (2 + (1)))$ yang akan menghasilkan nilai 10, yang merupakan hasil penjumlahan dari $1+2+3+4$.

Di dalam fungsi **main**, program meminta input dari pengguna berupa sebuah bilangan bulat n. Setelah mendapatkan input, program akan memanggil fungsi **jumlahRekursif** dengan parameter n dan menampilkan hasilnya. Misalnya, jika pengguna memasukkan angka 5, program akan menghitung $1+2+3+4+5$ secara rekursif dan menampilkan hasil 15. Program ini merupakan alternatif rekursif untuk menghitung jumlah deret aritmatika dengan beda 1 dan suku pertama 1.

3. Soal Studi Case

Mencari nilai pangkat atau 2^n .

Sourcecode

```
package main

import "fmt"

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))
}
```

```
} }
```

Screenshoot Output

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Guided4-3\Guided4-3.go"
Enter two integers: 8 4
Result from f2 (stored in c): 17
Result from f1: 7
PS D:\Alpro2Golang> 
```

Deskripsi Program

Program ini adalah implementasi rekursif untuk menghitung bilangan 2 dipangkatkan n (2^n). Fungsi utamanya bernama **pangkatDua** yang menerima parameter n bertipe integer. Program menggunakan konsep rekursi dengan base case yang terjadi ketika n sama dengan 0, dimana fungsi akan mengembalikan nilai 1 (karena $2^0 = 1$).

Untuk nilai n yang lebih besar dari 0, fungsi akan mengembalikan nilai 2 dikalikan dengan hasil pemanggilan rekursif **pangkatDua**($n-1$). Proses ini akan berlanjut hingga mencapai base case. Sebagai ilustrasi, jika $n=3$, maka proses perhitungannya akan menjadi $2 * (2 * (2 * (1)))$, yang akan menghasilkan nilai 8 ($2^3 = 8$). Setiap pemanggilan rekursif akan mengurangi nilai n sebanyak 1 hingga mencapai 0.

Di dalam fungsi **main**, program meminta input dari pengguna berupa bilangan bulat n yang akan menjadi pangkat dari 2. Setelah mendapatkan input, program memanggil fungsi **pangkatDua** dengan parameter n dan menampilkan hasilnya. Misalnya, jika pengguna memasukkan angka 4, program akan menghitung 2^4 secara rekursif dan menampilkan hasil 16. Program ini merupakan cara alternatif untuk menghitung pangkat dari 2 menggunakan pendekatan rekursif, dibandingkan dengan menggunakan perulangan biasa.

4. Soal Studi Case

Mencari nilai faktorial atau $n!$.

Sourcecode

```
package main

import "fmt"
```

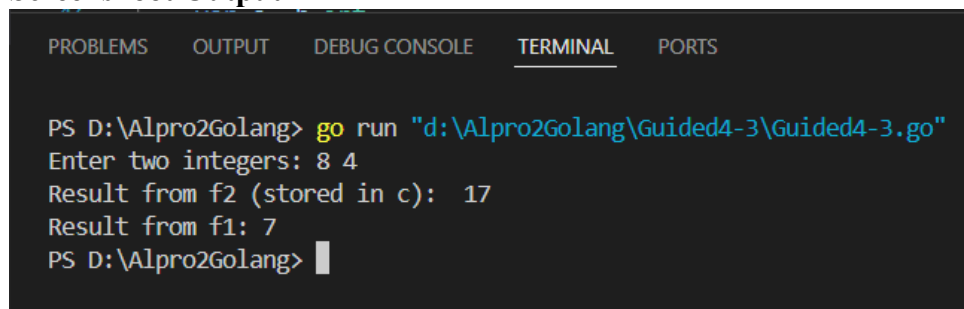
```

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.Scan(&n)
    fmt.Println("Hasil faktorial dari ", n, ":",
faktorial(n))
}

```

Screenshoot Output



```

PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Guided4-3\Guided4-3.go"
Enter two integers: 8 4
Result from f2 (stored in c): 17
Result from f1: 7
PS D:\Alpro2Golang>

```

Deskripsi Program

Program ini merupakan implementasi rekursif untuk menghitung nilai faktorial dari sebuah bilangan n ($n!$). Fungsi utamanya bernama **faktorial** yang menerima parameter n bertipe integer. Program menggunakan konsep rekursi dengan base case yang terjadi ketika n bernilai 0 atau 1, dimana fungsi akan mengembalikan nilai 1 (karena $0! = 1$ dan $1! = 1$).

Untuk nilai n yang lebih besar dari 1, fungsi akan mengembalikan nilai n dikalikan dengan hasil pemanggilan rekursif **faktorial**($n-1$). Proses ini akan terus berlanjut hingga mencapai base case. Sebagai contoh, jika $n=4$, maka proses perhitungannya akan menjadi $4 * (3 * (2 * (1)))$, yang akan menghasilkan nilai 24 ($4! = 24$). Setiap pemanggilan rekursif akan mengurangi nilai n sebanyak 1 hingga mencapai base case.

Di dalam fungsi **main**, program meminta input dari pengguna berupa bilangan bulat n yang akan dihitung nilai faktorialnya. Setelah mendapatkan input, program memanggil fungsi **faktorial** dengan parameter n dan menampilkan hasilnya. Misalnya, jika pengguna memasukkan angka 5, program akan menghitung $5!$ secara rekursif ($5 * 4 * 3 * 2 * 1$) dan menampilkan hasil 120. Program ini menunjukkan bagaimana konsep rekursi dapat digunakan untuk menghitung faktorial sebagai alternatif dari penggunaan loop iteratif.

III. UNGUIDED

1. Soal Studi Case

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

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

Sourcecode

```
package main

import "fmt"

func fibonacci_226(n int) int {
    if n <= 1 {
        return n
    }
    return fibonacci_226(n-1) + fibonacci_226(n-2)
}

func main() {
    fmt.Printf("Deret Fibonacci: \n\n")
    fmt.Print("n : ")
    for i := 0; i <= 10; i++ {
        fmt.Printf("%4d", i)
    }
    fmt.Print("\nSn : ")
    for i := 0; i <= 10; i++ {
        fmt.Printf("%4d", fibonacci_226(i))
    }
}
```

Screenshoot Output

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Unguided5-1\Unguided5-1.go"
Deret Fibonacci:

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 D:\Alpro2Golang> |
```

Deskripsi Program

Program ini adalah implementasi deret Fibonacci dalam bahasa Go (Golang). Program tersebut terdiri dari dua fungsi utama yaitu **fibonacci_226** dan **main**. Fungsi **fibonacci_226** menggunakan pendekatan rekursif untuk menghitung nilai Fibonacci ke-n, dimana jika n kurang dari atau sama dengan 1, fungsi akan mengembalikan nilai n itu sendiri, namun jika tidak, fungsi akan memanggil dirinya sendiri dengan parameter n-1 dan n-2 yang kemudian dijumlahkan hasilnya.

Di dalam fungsi **main**, program akan mencetak deret Fibonacci dari indeks 0 hingga 10. Pertama-tama program mencetak baris header "Deret Fibonacci" diikuti dengan baris "n : " yang menampilkan indeks dari 0 sampai 10 dengan format lebar 4 karakter untuk setiap angka. Selanjutnya, program mencetak baris "Sn : " yang menampilkan nilai Fibonacci untuk setiap indeks tersebut, juga dengan format lebar 4 karakter per angka. Hasil yang ditampilkan akan membentuk tabel sederhana yang menunjukkan hubungan antara indeks n dan nilai Fibonacci ke-n.

Output program ini akan menampilkan dua baris angka: baris pertama menunjukkan indeks (0 sampai 10) dan baris kedua menunjukkan nilai deret Fibonacci yang sesuai untuk setiap indeks tersebut, dimana setiap angka dalam deret merupakan hasil penjumlahan dari dua angka sebelumnya dalam deret tersebut.

2. Soal Studi Case

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

No	Masukan	Keluaran
1	5	* ** ***

		**** *****
2	1	*
3	3	* ** ***

Sourcecode

```
package main

import (
    "fmt"
)

func cetakBintang_226(n int) string {
    if n == 0 {
        return ""
    }
    return "*" + cetakBintang_226(n-1)
}

func printPattern(baris, totalBaris int) {
    if baris > totalBaris {
        return
    }
    fmt.Println(cetakBintang_226(baris))
    printPattern(baris+1, totalBaris)
}

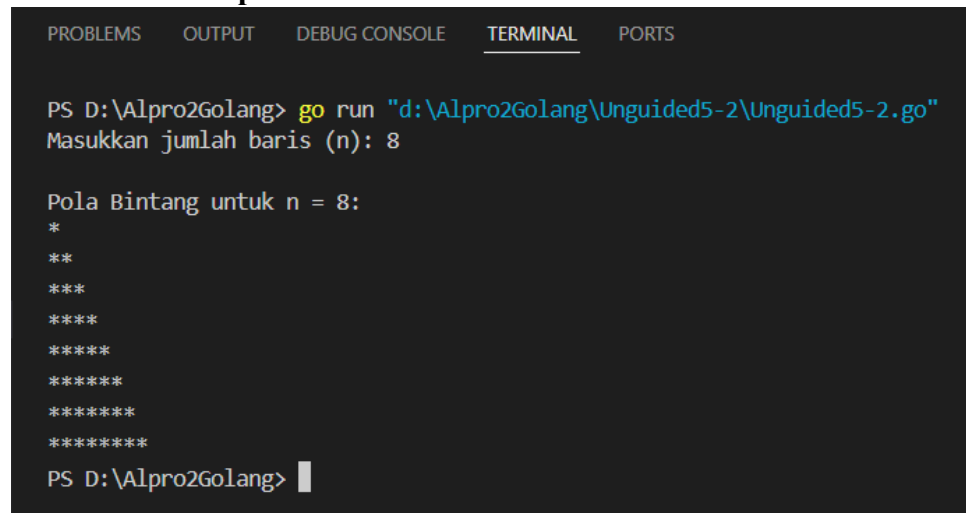
func main() {
    var n int

    for {
        fmt.Print("Masukkan jumlah baris (n): ")
        fmt.Scan(&n)

        if n > 0 {
            break
        }
        fmt.Println("Masukkan harus lebih besar dari 0!")
    }
}
```

```
fmt.Printf("\nPola Bintang untuk n = %d:\n", n)
printPattern(1, n)
}
```

Screenshoot Output



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Unguided5-2\Unguided5-2.go"
Masukkan jumlah baris (n): 8

Pola Bintang untuk n = 8:
*
**
***
****
*****
*****
*****
*****

PS D:\Alpro2Golang>
```

Deskripsi Program

Program ini adalah implementasi pola bintang menggunakan bahasa Go (Golang) dengan pendekatan rekursif. Program ini memiliki tiga fungsi utama: **cetakBintang_226**, **printPattern**, dan **main**. Fungsi **cetakBintang_226** bertugas untuk menghasilkan string bintang (*) sebanyak n kali secara rekursif, dimana jika n sama dengan 0 maka akan mengembalikan string kosong, dan jika tidak maka akan menambahkan satu bintang dan memanggil dirinya sendiri dengan parameter n-1.

Fungsi **printPattern** mengatur bagaimana pola bintang akan ditampilkan baris per baris. Fungsi ini menerima dua parameter: baris (posisi baris saat ini) dan totalBaris (jumlah total baris yang diinginkan). Jika baris melebihi totalBaris, fungsi akan berhenti. Jika tidak, fungsi akan mencetak bintang untuk baris tersebut menggunakan fungsi **cetakBintang_226** dan kemudian memanggil dirinya sendiri dengan baris ditambah 1.

Dalam fungsi **main**, program meminta pengguna untuk memasukkan jumlah baris yang diinginkan. Program menggunakan loop untuk memastikan bahwa input yang diberikan lebih besar dari 0, jika input tidak valid maka akan terus meminta input hingga mendapatkan nilai yang valid. Setelah mendapatkan input yang valid, program akan mencetak pola bintang

dengan memanggil fungsi `printPattern` dimulai dari baris 1 hingga n. Hasilnya akan menampilkan pola bintang dimana setiap baris memiliki jumlah bintang yang sesuai dengan nomor barisnya, membentuk pola segitiga yang terdiri dari bintang.

3. Soal Studi Case

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).

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

Sourcecode

```
package main

import (
    "fmt"
)

func cariFaktor_226(n int, bilangan int) {
    if bilangan > n {
        return
    }

    if n%bilangan == 0 {
        fmt.Printf("%d ", bilangan)
    }

    cariFaktor_226(n, bilangan+1)
}

func main() {
    var n int

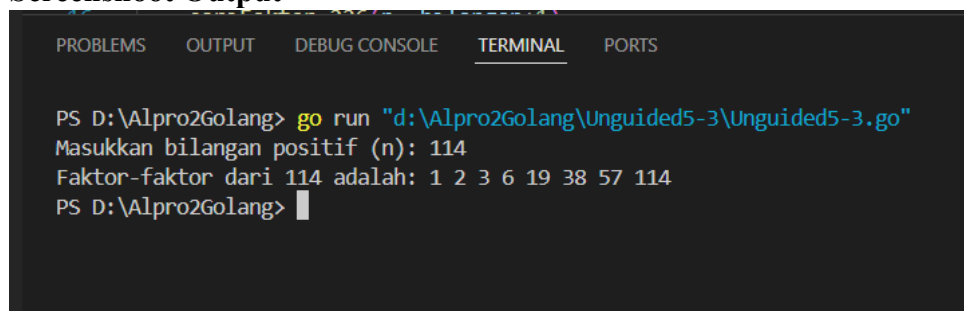
    for {
```

```

        fmt.Print("Masukkan bilangan positif (n): ")
        fmt.Scan(&n)
        if n > 0 {
            break
        }
        fmt.Println("Masukkan harus berupa bilangan positif!")
    }
    fmt.Printf("Faktor-faktor dari %d adalah: ", n)
    cariFaktor_226(n, 1)
}

```

Screenshoot Output



```

PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Unguided5-3\Unguided5-3.go"
Masukkan bilangan positif (n): 114
Faktor-faktor dari 114 adalah: 1 2 3 6 19 38 57 114
PS D:\Alpro2Golang>

```

Deskripsi Program

Program ini adalah implementasi pencarian faktor dari sebuah bilangan menggunakan bahasa Go (Golang) dengan pendekatan rekursif. Program terdiri dari dua fungsi utama yaitu **cariFaktor_226** dan **main**. Fungsi **cariFaktor_226** dirancang untuk mencari dan menampilkan faktor-faktor dari suatu bilangan *n* secara rekursif. Fungsi ini menerima dua parameter: *n* (bilangan yang akan dicari faktornya) dan bilangan (nilai yang sedang diperiksa apakah merupakan faktor dari *n*). Jika bilangan lebih besar dari *n*, fungsi akan berhenti. Jika *n* bisa dibagi habis oleh bilangan (ditandai dengan $n \% \text{bilangan} == 0$), maka bilangan tersebut akan dicetak sebagai salah satu faktor. Setelah itu, fungsi akan memanggil dirinya sendiri dengan bilangan ditambah 1 untuk memeriksa kemungkinan faktor berikutnya.

Di dalam fungsi **main**, program meminta pengguna untuk memasukkan sebuah bilangan positif. Program menggunakan loop untuk memastikan bahwa input yang diberikan adalah bilangan positif (lebih besar dari 0). Jika pengguna memasukkan bilangan tidak positif, program akan terus meminta input hingga mendapatkan bilangan positif yang valid. Setelah mendapatkan input yang valid, program akan mencetak pesan yang

menunjukkan faktor-faktor dari bilangan tersebut dan memanggil fungsi **cariFaktor_226** dengan parameter n (bilangan yang dimasukkan) dan 1 sebagai nilai awal pencarian faktor. Hasilnya akan menampilkan semua bilangan yang merupakan faktor dari n, yaitu bilangan-bilangan yang membagi n tanpa sisa.

4. Soal Studi Case

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.

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

Sourcecode

```
package main

import (
    "fmt"
)

func cetakTurun_226(n int, bilangan int) {
    if bilangan < 1 {
        return
    }
    fmt.Printf("%d ", bilangan)
    cetakTurun_226(n, bilangan-1)
}

func cetakNaik_226(n int, bilangan int) {
    if bilangan > n {
        return
    }
    fmt.Printf("%d ", bilangan)
    cetakNaik_226(n, bilangan+1)
}
```



```

func main() {
    var n int

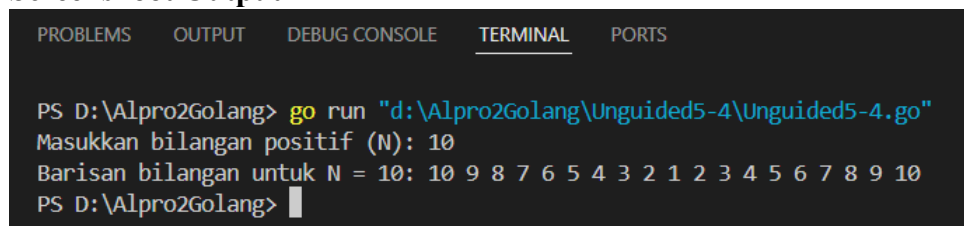
    for {
        fmt.Print("Masukkan bilangan positif (N): ")
        fmt.Scan(&n)

        if n > 0 {
            break
        }
        fmt.Println("Masukkan harus berupa bilangan positif!")
    }

    fmt.Printf("Barisan bilangan untuk N = %d: ", n)
    cetakTurun_226(n, n)
    cetakNaik_226(n, 2)
}

```

Screenshoot Output



```

PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Unguided5-4\Unguided5-4.go"
Masukkan bilangan positif (N): 10
Barisan bilangan untuk N = 10: 10 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9 10
PS D:\Alpro2Golang>

```

Deskripsi Program

Program ini adalah implementasi mencetak barisan bilangan secara menurun dan naik menggunakan bahasa Go (Golang) dengan pendekatan rekursif. Program ini memiliki tiga fungsi utama: **cetakTurun_226**, **cetakNaik_226**, dan **main**. Fungsi **cetakTurun_226** dirancang untuk mencetak bilangan secara menurun dari n hingga 1, dimana fungsi akan berhenti jika bilangan kurang dari 1. Setiap kali fungsi dipanggil, ia akan mencetak nilai bilangan saat ini dan kemudian memanggil dirinya sendiri dengan bilangan dikurangi 1. Sementara itu, fungsi **cetakNaik_226** bertugas mencetak bilangan secara naik dari 2 hingga n, dimana fungsi akan berhenti jika bilangan melebihi n. Setiap pemanggilan fungsi akan mencetak nilai bilangan saat ini dan kemudian memanggil dirinya sendiri dengan bilangan ditambah 1.

Di dalam fungsi **main**, program meminta pengguna untuk memasukkan sebuah bilangan positif N. Program menggunakan loop untuk memastikan bahwa input yang diberikan adalah bilangan positif (lebih besar dari 0). Jika pengguna memasukkan bilangan tidak positif, program akan terus meminta input hingga mendapatkan bilangan positif yang valid. Setelah mendapatkan input yang valid, program akan mencetak barisan bilangan dengan memanggil fungsi **cetakTurun_226** untuk mencetak bilangan dari N hingga 1, dilanjutkan dengan memanggil fungsi **cetakNaik_226** untuk mencetak bilangan dari 2 hingga N. Hasilnya akan menampilkan barisan bilangan yang menurun dari N ke 1 dan kemudian naik dari 2 ke-N, membentuk pola yang unik dimana angka 1 hanya muncul sekali di tengah barisan.

5. Soal Studi Case

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.

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

Sourcecode

```
package main

import (
    "fmt"
)

func cetakGanjil_226(n int, bilangan int) {
    if bilangan > n {
        return
    }

    if bilangan%2 != 0 {
        fmt.Printf("%d ", bilangan)
    }
}
```

```

        cetakGanjil_226(n, bilangan+1)
    }

func main() {
    var n int

    for {
        fmt.Print("Masukkan bilangan positif (N): ")
        fmt.Scan(&n)

        if n > 0 {
            break
        }
        fmt.Println("Masukkan harus berupa bilangan positif!")
    }

    fmt.Printf("Bilangan ganjil dari 1 hingga %d: ", n)
    cetakGanjil_226(n, 1)
}

```

Screenshoot Output

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Unguided5-5\Unguided5-5.go"
Masukkan bilangan positif (N): 15
Bilangan ganjil dari 1 hingga 15: 1 3 5 7 9 11 13 15
PS D:\Alpro2Golang>

```

Deskripsi Program

Program ini adalah implementasi untuk mencetak bilangan ganjil dalam rentang tertentu menggunakan bahasa Go (Golang) dengan pendekatan rekursif. Program terdiri dari dua fungsi utama yaitu **cetakGanjil_226** dan **main**. Fungsi **cetakGanjil_226** dirancang untuk mencetak bilangan ganjil dari 1 hingga **n** secara rekursif. Fungsi ini menerima dua parameter: **n** (batas atas rentang bilangan) dan **bilangan** (nilai yang sedang diperiksa). Jika bilangan lebih besar dari **n**, fungsi akan berhenti. Fungsi memeriksa apakah suatu bilangan adalah ganjil dengan menggunakan operator modulo (%), dimana jika bilangan dibagi 2 memiliki sisa (tidak habis dibagi 2), maka bilangan tersebut akan dicetak. Setelah itu,

fungsi akan memanggil dirinya sendiri dengan bilangan ditambah 1 untuk memeriksa bilangan berikutnya.

Di dalam fungsi **main**, program meminta pengguna untuk memasukkan sebuah bilangan positif N. Program menggunakan loop untuk memastikan bahwa input yang diberikan adalah bilangan positif (lebih besar dari 0). Jika pengguna memasukkan bilangan tidak positif, program akan terus meminta input hingga mendapatkan bilangan positif yang valid. Setelah mendapatkan input yang valid, program akan mencetak pesan yang menunjukkan bilangan ganjil dari 1 hingga N dan memanggil fungsi **cetakGanjil_226** dengan parameter N (batas atas) dan 1 sebagai nilai awal. Hasilnya akan menampilkan semua bilangan ganjil dalam rentang tersebut secara berurutan, dimulai dari 1 hingga bilangan ganjil terakhir yang tidak melebihi N.

6. Soal Studi Case

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".

No	Masukan	Keluaran
1	2 2	4
2	5 3	125

Sourcecode

```
package main

import "fmt"

func cetakPangkat_226(x, y int) int {
    if y == 0 {
        return 1
    }
    return x * cetakPangkat_226(x, y-1)
}
```

```

}

func main() {
    var x, y int

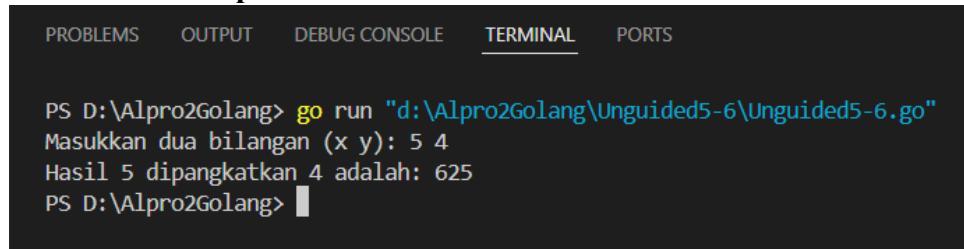
    fmt.Print("Masukkan dua bilangan (x y): ")
    fmt.Scan(&x, &y)

    result := cetakPangkat_226(x, y)

    fmt.Printf("Hasil %d dipangkatkan %d adalah: %d\n", x,
y, result)
}

```

Screenshoot Output



```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\Alpro2Golang> go run "d:\Alpro2Golang\Unguided5-6\Unguided5-6.go"
Masukkan dua bilangan (x y): 5 4
Hasil 5 dipangkatkan 4 adalah: 625
PS D:\Alpro2Golang>

```

Deskripsi Program

Program ini adalah implementasi perhitungan pangkat dengan pendekatan rekursif. Program terdiri dari dua fungsi utama yaitu **cetakPangkat_226** dan **main**. Fungsi **cetakPangkat_226** dirancang untuk menghitung hasil perpangkatan dari suatu bilangan secara rekursif. Fungsi ini menerima dua parameter: **x** (bilangan yang akan dipangkatkan) dan **y** (pangkat). Jika **y** sama dengan 0, fungsi akan mengembalikan nilai 1 sesuai dengan aturan matematika bahwa bilangan berapapun dipangkatkan 0 hasilnya adalah 1. Jika tidak, fungsi akan mengalikan **x** dengan hasil pemanggilan dirinya sendiri dengan parameter **y** dikurangi 1, sehingga membentuk proses perkalian berulang sebanyak **y** kali.

Di dalam fungsi **main**, program meminta pengguna untuk memasukkan dua bilangan **x** dan **y**, dimana **x** adalah bilangan yang akan dipangkatkan dan **y** adalah pangkatnya. Program kemudian memanggil fungsi **cetakPangkat_226** dengan kedua bilangan tersebut sebagai parameter dan menyimpan hasilnya dalam variabel **result**. Terakhir, program mencetak hasil perhitungan dalam format yang menunjukkan bilangan **x**, pangkat **y**, dan hasil perpangkatannya. Output program akan menampilkan kalimat

yang memberitahu user bilangan apa yang dipangkatkan berapa dan hasil akhir dari perpangkatan tersebut.