

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 6
REKURSIF**



Disusun Oleh :

M. Haidar Akhbiyani / 2311102276

S1-IF-11-06

Dosen Pengampu :

Abednego Dwi Septiadi

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

A. DEFINISI UMUM (ITERATIF VS REKURSIF)

- Pendekatan Iteratif : Pendekatan dengan memberikan konsep *looping* terhadap sebuah instruksi. *Loop* tetap diberi **batas tertentu**, ketika sudah melewati batas maka *looping* akan terhenti. Pendekatan Iteratif ini juga biasanya melibatkan **variabel lokal** dalam sebuah intruksinya.
- Pendekatan Rekursif : Pendekatan dengan fokus penggunaan *method* dengan instruksi berupa manggil dirinya sendiri sehingga terciptalah perulangan dan akan berhenti pada **kasus dasar** tertentu.

B. ILUSTRASI DASAR (ITERATIF VS REKURSIF)

- Permasalahan : Dewa dan Dewi sedang menonton pertunjukkan teater. Diketahui bahwa disediakan beberapa barisan tempat duduk untuk menonton teater. Diketahui pula, sang Dewi meninggalkan barisan tempat duduk karena ingin memberi air mineral di kantin luar tetapi ia lupa baris ke berapa tempat ia duduk waktu itu. Lalu, Dewi pun bertanya ke Dewa mengenai nomor baris tempat duduk mereka tetapi Dewa pun lupa.
- Pertanyaan : Selesaikan permasalahan tersebut secara pendekatan iteratif dan pendekatan rekursif !
- Solusi Iteratif

```
func hitungBarisDewa(posisiDewa int) {  
    nomorBaris := 1  
  
    for nomorBaris <= posisiDewa {
```

```

        fmt.Printf("Baris ke-%d\n", nomorBaris)

        if nomorBaris == posisiDewa {

            break

        }

        nomorBaris++

    }

}

func main() {

    hitungBarisDewa(5)

}

```

- **Solusi Rekursif**

```

package main

import (

    "fmt"

)

// Fungsi rekursif untuk mengetahui nomor baris Dewa

func nomorBaris(posisi int) int {

    // Basis rekursi: Jika posisi 1, maka nomor
    barisnya adalah 1

    if posisi == 1 {

        return 1

    }

}

```

```

        // Rekursi: Dapatkan nomor baris dari orang yang
duduk di depan

        barisDepan := nomorBaris(posisi - 1)


        // Nomor baris orang di belakang adalah
barisDepan + 1

        return barisDepan + 1
    }

func main() {

    posisiDewa := 5 // Misal posisi Dewa adalah
baris ke-5

    nomorBarisDewa := nomorBaris(posisiDewa)

    fmt.Printf("Dewa duduk di baris nomor: %d\n",
nomorBarisDewa)

}

```

C. STRATEGI PENDEKATAN REKURSIF

Ada dua hal yang menjadi concern kita jika ingin menyelesaikan masalah secara pendekatan rekursif, yaitu :

- Base Case : kasus paling sederhana dari suatu permasalahan.
- Reccurence Relation : Bagaimana hubungan rekursif dari permasalahan ini dengan permasalahan yang lebih kecil?

D. CONTOH SOAL

Buatlah sebuah algoritma untuk menyelesaikan permasalahan $n!$. Dengan definisi bahwa $n! = n * (n-1) * (n-2) * \dots * 3 * 2 * 1$, dengan $n \geq 0$ dan n suatu bilangan bulat.

Source code

```
package main

import (
    "fmt"
    "bufio"
    "os"
    "strconv"
)

// Fungsi utama
func main() {
    // Membaca input dari pengguna
    scanner := bufio.NewScanner(os.Stdin)
    fmt.Print("Masukkan angka: ")
    scanner.Scan()
    input, _ := strconv.Atoi(scanner.Text())

    // Menghitung faktorial menggunakan fungsi rekursif
    hasil := faktorialDari(input)
    fmt.Printf("Hasil dari %d faktorial adalah %d\n",
        input, hasil)
```

```

}

// Fungsi rekursif untuk menghitung faktorial
func faktorialDari(n int) int {
    // Basis rekursif
    if n == 1 {
        return 1
    }
    // Rekursi
    return n * faktorialDari(n-1)
}

```

Strategi awal tentu dalam hal ini ialah memahami definisi dari faktorial itu sendiri, Anda bisa menyelesaikan suatu masalah secara rekursif jika dan hanya jika Anda paham definisi umum dari permasalahan itu sendiri. Yang kedua, tentukan base case-nya. Kita tahu bahwa sebenarnya nilai faktorial akan tetap ketika sudah dikalikan dengan 1. Yang dimana kita tahu bahwa $1! = 1$. Yang ketiga, tentukan Recurrence Relation-nya. Untuk kasus $n > 1$, dalam mencari nilai $n!$ kita harus cari dulu dong nilai $(n-1)$ nya. Lalu, tinggal kalikan saja $\rightarrow n * (n-1)$. Nah, sekarang kita sudah tahu apa relasi rekursifnya.

II. GUIDED

1. Soal Studi Case

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)
}
```

Output

```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laparak Modul 5\Guided 1\main.go"
Masukkan nilai n untuk cetak bilangan dari n hingga 1: 5
Hasil cetak mundur: 5 4 3 2 1
PS D:\P Alpro 2> █
```

Deskripsi

Program diatas dibuat untuk menghitung penjumlahan hitung mundur dengan menggunakan fungsi rekursif, yang dimana terdapat `func cetakMundur(n int)` memiliki kondisi base case dimana jika input (**n**) == **1**, dan memiliki fungsi rekursif pada jika iput(**n**) != **1**, maka melakukan perulangan secara terus menerus hingga memenuhi base case yaitu **1**, fungsi rekursif terletak pada pemanggilan fungsi didalam sebuah fungsi baris (`bilangan - 1`). Kenapa terdapat pengurangan dalam pemanggilannya karena jika tidak dikurangi maka akan infinite loop dan akan mengeprint inputan(`bilangan`) dari user. Contoh jika user menginputkan **10**, dan kondisi rekursifnya baris (`bilangan`). Akan *infinite loop* mengeprint "**10**"

2. Soal Studi Case

Sourcecode

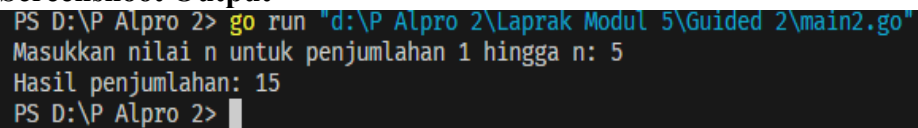
```
package main

import "fmt"

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

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

Screenshoot Output



```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laparak Modul 5\Guided 2\main2.go"
Masukkan nilai n untuk penjumlahan 1 hingga n: 5
Hasil penjumlahan: 15
PS D:\P Alpro 2>
```

Deskripsi Program

Program diatas dibuat untuk mencari hasil penjumlahan dari inputan user yang dikurang secara terus menerus(rekursif). Program diatas memiliki func `penjumlahan(n int)` yang dimana memiliki kondisi base case berupa `n == 1` maka program akan me-return inputan *user*, cara kerja program diatas adalah, *user* akan menginputkan sebuah bilangan bulat, pada contoh output diatas, *user* menginputkan "5" dan hasilnya "15". Fungsi rekursif diatas memiliki penyimpanan temporary yang dimana program akan menyimpan data sementara, pada kasus diatas, program akan menyimpan `n +`, kemudian melakukan perulangan rekursif dengan fungsi `penjumlahan(n-1)`, Jika dituliskan maka, `5 + penjumlahan(4)`, Program akan menyimpan data `n` yaitu "5" kemudian dijumlahkan dengan `n` yang baru yaitu "4", kemudian fungsi akan mengembalikannya lagi `penjumlahan(3)` dan seterusnya hingga memenuhi kondisi base case yaitu `n == 1`. Maka akan terjadi penjumlahan seperti $5+4+3+2+1 = 15$.

3. Soal Studi Case

Sourcecode

```
package main

import "fmt"

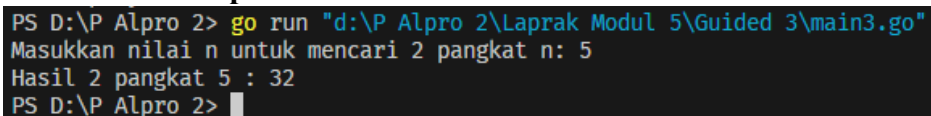
func pangkatDua(n int) int {
    if n == 0 {
        return 1
    }

    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



```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laparak Modul 5\Guided 3\main3.go"
Masukkan nilai n untuk mencari 2 pangkat n: 5
Hasil 2 pangkat 5 : 32
PS D:\P Alpro 2>
```

Deskripsi Program

Program diatas dibuat untuk mencari hasil perpangkatan dari 2 pangkat n, yang dimana program diatas melakukan perulangan sebanyak inputan n, Program diatas memiliki fungsi `func pangkatDua(n int) int` dengan kondisi base case yaitu n sama dengan 0, dan program ini akan melakukan perulangan 2 sebanyak n-kali, cara kerja programnya adalah *user* menginputkan nilai **n**, kemudian **n** akan di lakukan perulangan rekursif `2 * pangkatDua(n-1)`. Program ini sama dengan program guided ke 2 yang dimana yang membedahkan adalah operasinya dan variabel kontans pada nilai *return*. Jadi jika $n = 4$, maka program akan melakukan $2 * pangkat(4-1)$. Program akan menyimpan sementara nilai $2 *$, kemudian melakukan perulangan hingga $n == 0$, maka akan didapatkan nilai *return* yaitu $2 * 2 * 2 * 2 = 32$

4. Soal Studi Case

Sourcecode

```
package main

import "fmt"

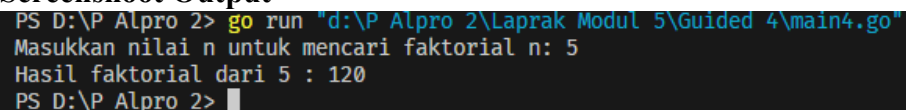
func faktorial(n int) int {
    if n == 0 || n == 1 {
        return 1
    }

    return n * faktorial(n-1)
}

func main() {
    var n int

    fmt.Print("Masukkan nilai n untuk mencari faktorial
n: ")
    fmt.Scanln(&n)
    fmt.Println("Hasil faktorial dari", n, ":",
faktorial(n))
}
```

Screenshoot Output



```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laprak Modul 5\Guided 4\main4.go"
Masukkan nilai n untuk mencari faktorial n: 5
Hasil faktorial dari 5 : 120
PS D:\P Alpro 2>
```

Deskripsi Program

Program diatas dibuat untuk mencari sebuah faktorial dari **n**, Program diatas memiliki fungsi rekursif berupa `func faktorial(n int)` yang dimana memiliki kondisi base case yaitu jika `n == 0` atau `n == 1`, program ini melakukan perkalian **n**, dengan **(n-1)** sehingga akan menampilkan hasil faktorial dari **n**, Cara kerja program ini adalah *user* akan menginputkan nilai **n** yang berupa (5) kemudian akan melakukan perulangan rekursif `return 5 * faktorial(5-1)`, maka akan menjadi seperti ini $5 * 4 * 3 * 2 * 1 = 120$, dan jika `n == 1` maka akan berhenti atau bisa disebut juga sebagai base case.

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

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

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

    fmt.Print("\nSn : ")
    for j := 0; j < n; j++ {
        fmt.Print(fibonacci(j), " ")
    }
}
```

Screenshoot Output

```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laparak Modul 5\Unguided 1\main.go"
Masukkan Panjang n : 10
n : 0 1 2 3 4 5 6 7 8 9
Sn : 0 1 1 2 3 5 8 13 21 34
PS D:\P Alpro 2> █
```

Deskripsi Program

Program di atas adalah implementasi deret Fibonacci, yang terdiri dari dua fungsi utama. Fungsi pertama adalah `fibonacci(n int)` yang menghitung nilai Fibonacci ke- n secara rekursif dengan base case jika $n = 0$ maka return 0, jika $n = 1$ maka return 1, dan untuk rekursif $n > 1$ nilai Fibonacci adalah jumlah dari 2 nilai sebelumnya. Output program akan menampilkan dua baris sesuai dengan inputan *user*, pada output diatas *user* menginputkan “10” maka baris pertama menunjukkan indeks 0 sampai 9 dan baris kedua menunjukkan nilai Fibonacci untuk setiap indeks tersebut (contoh: 0 1 1 2 3 5 8 13 21 34), sehingga program ini efektif mendemonstrasikan deret Fibonacci dimana setiap angka merupakan hasil penjumlahan dari dua angka sebelumnya. $Fibonacci(n) = (S_{n-1}) + (S_{n-2})$. Contoh sederhana :

- $fibonacci(0) = 0$
- $fibonacci(1) = 1$
- $fibonacci(2) = 1 (1 + 0)$
- $fibonacci(3) = 2 (1 + 1)$
- $fibonacci(4) = 3 (2 + 1)$
- $fibonacci(5) = 5 (3 + 2)$

2. Soal Studi Case

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	* ** ***

Sourcecode

```
package main

import "fmt"

func PolaAsterisk(n int) {
    if n == 0 {
        return
    }
    PolaAsterisk(n - 1)

    for i := 0; i < n; i++ {
        fmt.Print("*")
    }

    fmt.Println()
}

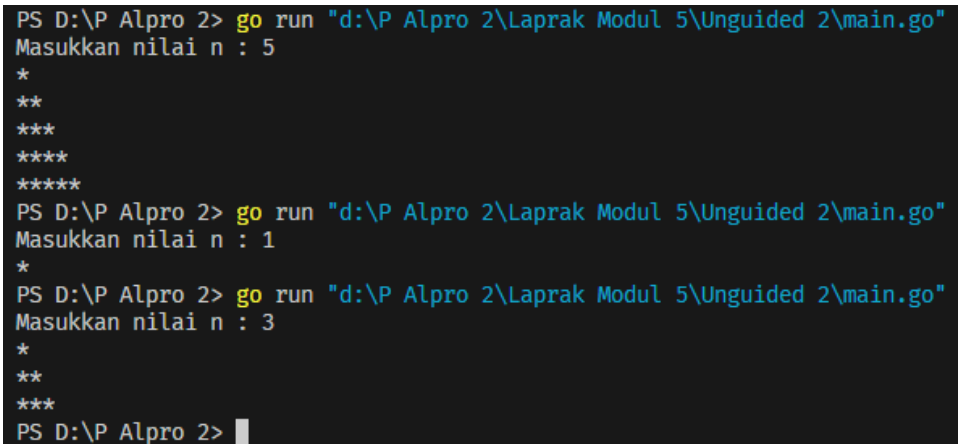
func main() {

    var n int

    fmt.Print("Masukkan nilai n : ")
    fmt.Scan(&n)
```

```
PolaAsterisk(n)
}
```

Screenshoot Output



```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laprak Modul 5\Unguided 2\main.go"
Masukkan nilai n : 5
*
**
***
****
*****
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laprak Modul 5\Unguided 2\main.go"
Masukkan nilai n : 1
*
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laprak Modul 5\Unguided 2\main.go"
Masukkan nilai n : 3
*
**
***
PS D:\P Alpro 2> █
```

Deskripsi Program

Program ini dirancang untuk menghasilkan pola bintang yang menurun berbentuk seperti tangga atau segitiga. Program memiliki fungsi `PolaAsterisk(n int)` yang bekerja secara rekursif untuk mencetak bintang (*) dengan pola meningkat. Fungsi ini memiliki base case ketika $n = 0$ maka fungsi akan berhenti (return). Setiap pemanggilan fungsi akan mengurangi nilai n sebanyak 1 ($n-1$) dan mencetak bintang sebanyak n pada setiap barisnya. Proses kerjanya adalah sebagai berikut:

- Masukkan nilai (**n**)
- Panggil fungsi `PolaAsterisk(n int)`
- Di dalam fungsi `PolaAsterisk(n int)` terdapat pemanggilan `PolaAsterisk(n - 1)`
- Nilai **n** akan terus berkurang hingga mencapai base case yaitu **n = 0**, setelah itu program akan berhenti.

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).

Contoh **masukan** dan **keluaran**:

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

Sourcecode

```
package main

import "fmt"

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

func main() {
    var n int

    fmt.Print("Masukkan bilangan bulat : ")
    fmt.Scan(&n)
    fmt.Print("Faktor dari ", n, ": ")
    faktorBilangan(n, 1)
}
```

Screenshoot Output

```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laparak Modul 5\Unguided 3\main.go"
Masukkan bilangan bulat : 5
Faktor dari 5: 1 5
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laparak Modul 5\Unguided 3\main.go"
Masukkan bilangan bulat : 12
Faktor dari 12: 1 2 3 4 6 12
PS D:\P Alpro 2> █
```

Deskripsi Program

Program ini dirancang untuk menghitung faktorisasi dari bilangan bulat n , yang berfungsi untuk mencari faktor-faktor dari sebuah bilangan bulat menggunakan pendekatan rekursif. Program memiliki fungsi `faktorBilangan(n, hasil int)` yang bekerja dengan cara memeriksa setiap bilangan dari 1(hasil) hingga n untuk menentukan apakah bilangan tersebut merupakan faktor dari n . Fungsi akan berhenti (return) jika nilai hasil sudah melebihi n , dan akan mencetak nilai hasil jika n modulus hasil sama dengan 0 (artinya hasil adalah faktor dari n), Jika inputan *user* berupa "12" hasilnya akan ditampilkan sebagai berikut:

- 12 % 1 (*true*)
- 12 % 2 (*true*)
- 12 % 3 (*true*)
- 12 % 4 (*true*)
- 12 % 5 (*false*)
- 12 % 6 (*true*)
- 12 % 7 (*false*)
- 12 % 8 (*false*)
- 12 % 9 (*false*)
- 12 % 10 (*false*)
- 12 % 11 (*false*)
- 12 % 12 (*true*)

Jika kondisi $n \% i == 0$ tidak terpenuhi, maka program berhenti dan menampilkan semua hasil(*true*) yang diperoleh.

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.

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

Sourcecode

```
package main

import "fmt"

func baris(n, hasil int) {
    if n == hasil {
        fmt.Print(1, " ")
        return
    }
    fmt.Print(n, " ")
    baris(n-1, hasil)
    fmt.Print(n, " ")
}

func main() {
    var n int
    fmt.Print("Masukkan bilangan bulat n : ")
    fmt.Scan(&n)
    fmt.Print("Hasil : ")
    baris(n, 1)
}
```

Screenshoot Output

```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laprak Modul 5\Unguided 4\main.go"
Masukkan bilangan bulat n : 5
Hasil : 5 4 3 2 1 2 3 4 5
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laprak Modul 5\Unguided 4\main.go"
Masukkan bilangan bulat n : 9
Hasil : 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9
PS D:\P Alpro 2> █
```

Deskripsi Program

Program ini dirancang untuk menggunakan rekursif untuk mencetak pola bilangan dengan format menurun kemudian naik. Program memiliki fungsi `func_baris(n, hasil int)` memiliki kegunaan untuk mencetak bilangan dari `n` hingga 1, kemudian kembali mencetak dari 1 hingga `n`. Fungsi ini memiliki base case ketika `n` sama dengan `hasil` (`n == hasil`), maka akan mencetak angka 1 dan berhenti. Cara kerja pemanggilan fungsi rekursif yaitu akan mencetak nilai `n` saat ini, kemudian memanggil fungsi dengan nilai `n` akan dikurangi 1 (`n-1`), dan setelah pemanggilan rekursif selesai akan mencetak nilai `n` lagi. Berikut adalah langkah kerja dari fungsi ini:

- Fungsi dipanggil pertama kali sebagai `baris(5, 1)`
- Memasuki rekursi untuk menurunkan nilai '`n`' (`baris (n-1, i)`)
- Mencetak 5, lanjut ke `baris(4, 1)`
- Mencetak 4, lanjut ke `baris(3, 1)`
- Mencetak 3, lanjut ke `baris(2, 1)`
- Mencetak 2, lanjut ke `baris(1, 1)`
- Ketika mencapai base case '`n == i`', angka 1 dicetak, dan rekursi berhenti menurun, kemudian memulai pengembalian nilai.
- Kembali ke `baris(2, 5)` dan mencetak 2 lagi.
- Kembali ke `baris(3, 5)` dan mencetak 3 lagi.
- Kembali ke `baris(4, 5)` dan mencetak 4 lagi.
- Kembali ke `baris(5, 5)` dan mencetak 5 lagi. Output akhir untuk `N = 5` adalah: '`5 4 3 2 1 2 3 4 5`'.

Dapat disimpulkan bahwa fungsi ini mencetak urutan angka menurun dari `N` ke 1, lalu meningkat kembali dari 1 ke `N`.

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.

Contoh **masukan** dan **keluaran**:

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 Ganjil(n, hasil int) {
    if hasil > n {
        return
    }
    if hasil%2 != 0 {
        fmt.Print(hasil, " ")
    }

    Ganjil(n, hasil+1)
}

func main() {
    var n int

    fmt.Print("Masukkan bilangan bulat n : ")
    fmt.Scan(&n)
    fmt.Print("Hasil bilangan ganjil dari ", n, " : ")
    Ganjil(n, 1)
}
```

Screenshoot Output

```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laprak Modul 5\Unguided 5\main.go"
Masukkan bilangan bulat n : 5
Hasil bilangan ganjil dari 5 : 1 3 5
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laprak Modul 5\Unguided 5\main.go"
Masukkan bilangan bulat n : 20
Hasil bilangan ganjil dari 20 : 1 3 5 7 9 11 13 15 17 19
PS D:\P Alpro 2>
```

Deskripsi Program

Program ini dibuat untuk mencari dan mencetak bilangan ganjil berdasarkan input dari pengguna. Program memiliki fungsi `Ganjil(n, hasil int)` yang bekerja secara rekursif dengan base case ketika nilai `hasil > n` maka fungsi akan berhenti (return). Dalam setiap iterasi, fungsi akan memeriksa apakah nilai `hasil` adalah bilangan ganjil dengan menggunakan operasi modulus (`hasil%2 != 0`), jika benar maka nilai tersebut akan dicetak. Fungsi kemudian akan memanggil dirinya sendiri dengan menambahkan 1 pada parameter `hasil` (`hasil+1`). i. Berikut adalah tahapan kerja dari fungsi ini:

- Rekursif pada `Ganjil(n, hasil + 1)`
- Input dimulai dengan (5, 1), yang akan diproses sebagai berikut:
- `Ganjil(5, 1)` mencetak angka karena 1 adalah bilangan ganjil.
- Fungsi terus berlanjut, mencetak bilangan ganjil berikutnya hingga mencapai kondisi base case di mana `hasil > n`, misalnya pada `Ganjil(5, 6)`, di mana fungsi tidak akan melanjutkan rekursif.

Program akan berhenti setelah semua bilangan ganjil dalam rentang yang diberikan dicetak.

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".

Contoh **masukan** dan **keluaran**:

No	Masukan	Keluaran
1	2 2	4
2	5 3	125

Sourcecode

```
package main

import "fmt"

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

func main() {
    var x, y int
    fmt.Print("Masukkan Bilangan(x, y) : ")
    fmt.Scan(&x, &y)
    fmt.Print("Hasil ", x, " pangkat ", y, " : ",
    pangkat(x, y))
}
```

Screenshoot Output

```
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laparak Modul 5\Unguided 6\main.go"
Masukkan Bilangan(x, y) : 2 2
Hasil 2 pangkat 2 : 4
PS D:\P Alpro 2> go run "d:\P Alpro 2\Laparak Modul 5\Unguided 6\main.go"
Masukkan Bilangan(x, y) : 5 3
Hasil 5 pangkat 3 : 125
PS D:\P Alpro 2> █
```

Deskripsi Program

Program ini dirancang untuk menghitung perpangkatan x^y menggunakan fungsi rekursif. Program ini memiliki fungsi `pangkat(x, y int)` yang bekerja secara rekursif untuk menghitung nilai x pangkat y . Fungsi ini memiliki base case ketika $y = 0$ maka akan mengembalikan nilai 1 (1 disini dimaksudkan nilai x sebanyak 1 kali). Jika y tidak sama dengan 0, maka fungsi akan mengalikan x dengan hasil pemanggilan rekursif `pangkat(x, y-1)`, dimana y dikurangi 1 pada setiap pemanggilan sampai mencapai kondisi base case

IV. REFERENSI

- [1] Digimensia, "Understanding Recursive Function in Golang," *Digimensia*, 2023. <https://digimensia.com/golang-recursive-function/>. Diakses pada 03 Nopember 2024