

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 5  
REKURSIF**



**Disusun Oleh :**

**MAULANA GHANI ROLANDA | 2311102012**

**IF-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

Rekursif adalah teknik pemrograman di mana sebuah fungsi memanggil dirinya sendiri untuk menyelesaikan masalah yang lebih kecil, hingga mencapai kondisi tertentu yang disebut base case atau kasus dasar. Kasus dasar ini berfungsi untuk menghentikan pemanggilan rekursif agar tidak berjalan terus-menerus dan menyebabkan kesalahan seperti stack overflow. Struktur umum fungsi rekursif terdiri dari dua bagian, yaitu base case untuk menghentikan rekursi dan recursive case di mana fungsi memanggil dirinya sendiri.

Beberapa contoh umum dari algoritma rekursif adalah fungsi faktorial, deret Fibonacci, dan algoritma pembagian dan penaklukan seperti Merge Sort dan Quick Sort. Pada fungsi faktorial, nilai faktorial dihitung dengan memanggil fungsi secara rekursif hingga mencapai base case ( $1!$ ). Pada deret Fibonacci, setiap elemen adalah jumlah dua elemen sebelumnya dengan base case  $F(0) = 0$  dan  $F(1) = 1$ . Sementara pada algoritma pembagian dan penaklukan, data dipecah menjadi bagian-bagian lebih kecil untuk diselesaikan secara rekursif, kemudian digabungkan kembali.

Rekursi memiliki kelebihan dan kekurangan. Keunggulan utamanya adalah kode yang dihasilkan dapat lebih sederhana dan elegan, terutama pada masalah yang secara alami memiliki sifat rekursif seperti pohon biner atau graf. Namun, rekursi juga bisa menyebabkan penggunaan memori yang tinggi dan lambat jika tidak dikendalikan dengan baik, terutama tanpa kondisi dasar yang memadai. Untuk mengatasi kekurangan ini, optimisasi rekursi seperti memoization dan rekursi ekor sering digunakan. Memoization adalah teknik menyimpan hasil perhitungan yang telah dilakukan sebelumnya untuk menghindari perhitungan ulang, sementara rekursi ekor memungkinkan pemanggilan rekursif terjadi di akhir fungsi, sehingga kompiler atau interpreter dapat mengoptimalkan penggunaan memori.

## II. GUIDED

### 1. Sourcecode

```
//2311102012
package main

import "fmt"

// Fungsi untuk mencetak bilangan dari n hingga 1
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)
}
```

### Screenshot Output

```
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaP
rak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\guided1.go"
Masukkan nilai n untuk cetak bilangan dari n hingga 1: 5
Hasil cetak mundur: 5 4 3 2 1
```

### Deskripsi Program

Program ini adalah program Go yang menggunakan fungsi rekursif untuk mencetak bilangan mundur dari n hingga 1.

**Input:** Pengguna memasukkan nilai n.

**Fungsi cetakMundur:** Mencetak n dan memanggil dirinya sendiri dengan n-1 hingga mencapai 1, di mana rekursi berhenti.

**Output:** Menampilkan bilangan dari n hingga 1, dipisahkan oleh spasi.

Contoh: Jika n = 5, maka outputnya adalah 5 4 3 2 1.

## 2. Source Code

```
//2311102012
package main

import "fmt"

// Fungsi untuk menghitung penjumlahan 1 hingga n
func jumlahRekursi(n int) int {
    if n == 1 {
        return 1
    }
    return n + jumlahRekursi(n-1)
}

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

### Screenshot Output

```
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\guided2.go"
Masukkan nilai n untuk penjumlahan 1 hingga n: 7
Hasil penjumlahan: 28
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> █
```

### Deskripsi Program

Program yang ditulis dalam bahasa Go ini dirancang untuk menghitung penjumlahan dari angka 1 hingga n menggunakan pendekatan rekursif. Pertama-tama, program mengimpor paket "fmt" untuk keperluan input dan output.

Di dalam program, terdapat sebuah fungsi bernama jumlahRekursi yang menerima satu parameter bertipe integer, yaitu n. Fungsi ini bekerja dengan cara memeriksa apakah nilai n sama dengan 1. Jika ya, fungsi akan mengembalikan nilai 1, yang merupakan dasar dari rekursi. Jika tidak, fungsi akan mengembalikan hasil penjumlahan antara n dengan hasil pemanggilan fungsi jumlahRekursi itu sendiri dengan argumen n-1. Dengan cara ini, fungsi akan terus memanggil dirinya sendiri, mengurangi nilai n setiap kali, hingga mencapai 1, pada saat mana rekursi mulai kembali ke atas dengan mengakumulasi hasil penjumlahan.

Di dalam fungsi main, program meminta pengguna untuk memasukkan nilai n melalui konsol. Nilai ini kemudian dibaca dan disimpan dalam variabel n. Setelah itu, program memanggil fungsi jumlahRekursi dengan argumen n dan mencetak hasil penjumlahan ke layar. Dengan demikian, pengguna dapat melihat hasil penjumlahan dari 1 hingga n berdasarkan nilai yang mereka masukkan.

### 3. Source Code

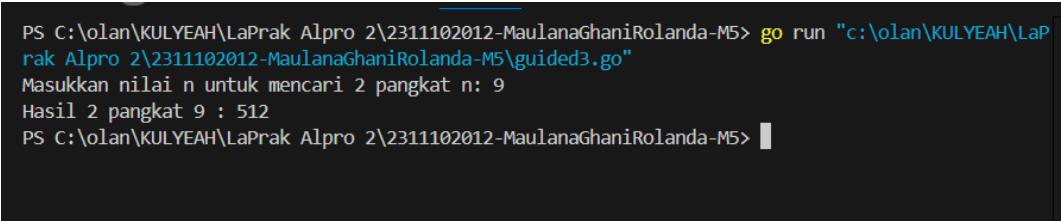
```
//2311102012
package main

import "fmt"

// Fungsi untuk mencari 2 pangkat n
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))
}
```

### Screenshot Output



```
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\guided3.go"
Masukkan nilai n untuk mencari 2 pangkat n: 9
Hasil 2 pangkat 9 : 512
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5>
```

### Deskripsi Program

Program dimulai dengan mengimpor paket "fmt" untuk menangani input dan output. Di dalam program, terdapat fungsi bernama pangkatDua yang menerima satu parameter bertipe integer, yaitu n. Fungsi ini bekerja dengan memeriksa

apakah  $n$  sama dengan 0. Jika  $n$  sama dengan 0, fungsi mengembalikan nilai 1, sesuai dengan definisi bahwa  $20=12^0 = 120=1$ . Jika  $n$  lebih besar dari 0, fungsi akan mengembalikan hasil perkalian 2 dengan hasil pemanggilan fungsi pangkatDua itu sendiri dengan argumen  $n-1$ . Ini memungkinkan fungsi untuk terus memanggil dirinya sendiri, mengurangi  $n$  hingga mencapai 0, dan kemudian melakukan perkalian pada setiap langkah kembali dalam proses rekursi.

Dalam fungsi main, program meminta pengguna untuk memasukkan nilai  $n$  melalui konsol. Setelah menerima input dari pengguna, nilai ini disimpan dalam variabel  $n$ . Program kemudian memanggil fungsi pangkatDua dengan argumen  $n$  dan mencetak hasil perhitungan  $2n2^n$  ke layar. Dengan demikian, pengguna dapat mengetahui hasil dari  $222$  yang dipangkatkan dengan nilai  $n$  yang mereka masukkan.

#### 4. Source Code

```
//2311102012
package main

import "fmt"

// Fungsi untuk menghitung faktorial n!
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))
}
```

#### ScreenShot Output

```
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\guided4.go"
Masukkan nilai n untuk mencari faktorial n!: 5
Hasil faktorial dari 5 : 120
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> |
```

### **Deskripsi Program**

Program dimulai dengan mengimpor paket "fmt" untuk memfasilitasi input dan output. Di dalam program, terdapat sebuah fungsi bernama faktorial yang menerima satu parameter bertipe integer, yaitu  $n$ . Fungsi ini menggunakan pendekatan rekursif untuk menghitung faktorial. Pertama, fungsi memeriksa apakah  $n$  sama dengan 0 atau 1. Jika ya, fungsi akan mengembalikan nilai 1, karena faktorial dari 0 dan 1 adalah 1. Jika  $n$  lebih besar dari 1, fungsi akan mengembalikan hasil perkalian  $n$  dengan hasil pemanggilan fungsi faktorial dengan argumen  $n-1$ . Proses ini berlanjut hingga  $n$  mencapai 0 atau 1, di mana fungsi mulai kembali ke atas dalam tumpukan rekursi dan mengalikan semua nilai.

Dalam fungsi main, program meminta pengguna untuk memasukkan nilai  $n$  melalui konsol. Setelah nilai dimasukkan, program menyimpannya dalam variabel  $n$  dan kemudian memanggil fungsi faktorial dengan argumen  $n$ . Hasil faktorial kemudian dicetak ke layar, sehingga pengguna dapat melihat hasil perhitungan dari  $n!$ . Dengan demikian, program ini memungkinkan pengguna untuk menghitung faktorial dari nilai yang mereka masukkan secara interaktif.

### III. UNGUIDED

#### Study Case 1

- 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.

$n$	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

```
//2311102012
package main

import "fmt"

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

func main() {

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



## Screenshoot Output

```
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\tempCodeRunnerFile.go"
Fibonacci(0) = 0
Fibonacci(1) = 1
Fibonacci(2) = 1
Fibonacci(3) = 2
Fibonacci(4) = 3
Fibonacci(5) = 5
Fibonacci(6) = 8
Fibonacci(7) = 13
Fibonacci(8) = 21
Fibonacci(9) = 34
Fibonacci(10) = 55
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5>
```

## Deskripsi Program

Program dimulai dengan mengimpor paket "fmt" yang digunakan untuk output ke konsol. Terdapat sebuah fungsi bernama fibonacci yang mengambil satu parameter bertipe integer, yaitu n. Fungsi ini menggunakan pendekatan rekursif untuk menghitung angka Fibonacci. Jika n sama dengan 0, fungsi mengembalikan 0; jika n sama dengan 1, fungsi mengembalikan 1. Untuk nilai n yang lebih besar dari 1, fungsi akan mengembalikan hasil penjumlahan dari pemanggilan fungsi fibonacci dengan argumen n-1 dan n-2. Ini mengikuti definisi dari deret Fibonacci, di mana setiap angka adalah hasil penjumlahan dua angka sebelumnya.

Di dalam fungsi main, program menggunakan sebuah loop for untuk menghitung dan mencetak nilai Fibonacci dari 0 hingga 10. Dalam setiap iterasi, fungsi fibonacci dipanggil dengan argumen i, dan hasilnya dicetak ke konsol menggunakan fmt.Printf, yang memformat output dengan cara yang terstruktur. Dengan demikian, pengguna dapat melihat setiap angka Fibonacci dari F(0) hingga F(10) ditampilkan di layar, memberikan gambaran yang jelas tentang deret Fibonacci untuk nilai-nilai tersebut.

## Study Case 2

- 2) 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	* ** ***

## Source Code

```
//2311102012
package main

import (
    "fmt"
)

// Fungsi rekursif untuk mencetak bintang dalam satu baris
func cetakBintang(jumlah int) {
    if jumlah > 0 {
        fmt.Print("*")
        cetakBintang(jumlah - 1)
    }
}

// Fungsi rekursif untuk mencetak pola bintang dari 1 hingga jumlah
func cetakPola(jumlah, current int) {
    if current <= jumlah {
        cetakBintang(current)
        fmt.Println()
        cetakPola(jumlah, current+1)
    }
}
```

```
func main() {
    var jumlah int
    fmt.Print("Masukkan jumlah baris: ")
    fmt.Scan(&jumlah)

    fmt.Println("Pola bintang:")
    cetakPola(jumlah, 1)
}
```

### Screenshot Output

```
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\unguided2.go"
Masukkan jumlah baris: 5
Pola bintang:
*
**
***
****
*****
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> |
```

### Deskripsi Program

Program diawali dengan mengimpor paket "fmt" untuk mendukung operasi input dan output. Terdapat dua fungsi rekursif dalam program ini: cetakBintang dan cetakPola.

Fungsi cetakBintang bertugas untuk mencetak bintang (\*) dalam satu baris. Fungsi ini menerima satu parameter, yaitu jumlah, yang menunjukkan berapa banyak bintang yang akan dicetak. Jika jumlah lebih besar dari 0, fungsi akan mencetak satu bintang dan kemudian memanggil dirinya sendiri dengan mengurangi nilai jumlah sebanyak 1. Proses ini berlanjut hingga jumlah menjadi 0, pada titik mana tidak ada lagi bintang yang dicetak.

Fungsi kedua, cetakPola, digunakan untuk mencetak pola bintang dari 1 hingga jumlah baris yang diminta. Fungsi ini memiliki dua parameter: jumlah, yang merupakan jumlah maksimum baris yang akan dicetak, dan current, yang menunjukkan baris saat ini yang sedang dicetak. Jika current kurang dari atau sama dengan jumlah, fungsi akan memanggil cetakBintang dengan argumen current untuk mencetak jumlah bintang yang sesuai untuk baris tersebut. Setelah itu, fungsi akan mencetak newline dan memanggil dirinya sendiri dengan menambah nilai current

sebesar 1. Dengan cara ini, pola bintang akan bertambah satu bintang di setiap barisnya.

Dalam fungsi main, program meminta pengguna untuk memasukkan jumlah baris melalui konsol. Nilai yang dimasukkan disimpan dalam variabel jumlah, dan kemudian fungsi cetakPola dipanggil dengan argumen jumlah dan current yang diinisialisasi ke 1. Dengan demikian, pengguna akan melihat pola bintang yang terbentuk sesuai dengan jumlah baris yang mereka tentukan, di mana setiap baris berisi satu bintang lebih banyak dari baris sebelumnya.

### Study Case 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).

Contoh masukan dan keluaran:

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

### Source Code

```
//2311102012
package main

import (
    "fmt"
)

// Fungsi rekursif untuk mencetak faktor bilangan
func cetakFaktor(n, i int) {
    if i > n {
        return
    }
    if n%i == 0 {
        fmt.Print(i, " ")
    }
    cetakFaktor(n, i+1)
}
```

```

}

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

    fmt.Printf("Faktor dari %d adalah: ", n)
    cetakFaktor(n, 1)
    fmt.Println()
}

```

### Screenshot Program

```

PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\unguided3.go"
Masukkan bilangan bulat positif: 12
Faktor dari 12 adalah: 1 2 3 4 6 12
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5>

```

### Deskripsi Program

Program dimulai dengan mengimpor paket "fmt" untuk menangani input dan output. Terdapat fungsi rekursif bernama cetakFaktor, yang bertugas untuk menemukan dan mencetak faktor dari bilangan n. Fungsi ini memiliki dua parameter: n, yang merupakan bilangan yang faktornya ingin dicetak, dan i, yang merupakan bilangan yang sedang diperiksa untuk menjadi faktor.

Di dalam fungsi cetakFaktor, pertama-tama ada kondisi untuk memeriksa apakah nilai i lebih besar dari n. Jika ya, fungsi akan mengembalikan kontrol tanpa melakukan apa pun, yang mengakhiri proses rekursi. Selanjutnya, fungsi memeriksa apakah n dapat dibagi habis oleh i dengan menggunakan operator modulus (%). Jika  $n \% i == 0$ , berarti i adalah faktor dari n, dan program mencetak nilai i diikuti dengan spasi.

Setelah memeriksa dan mencetak, fungsi cetakFaktor kemudian dipanggil kembali dengan parameter i yang ditambah satu. Proses ini berulang hingga semua angka dari 1 hingga n diperiksa.

Di dalam fungsi main, program meminta pengguna untuk memasukkan bilangan bulat positif melalui konsol. Nilai ini disimpan dalam variabel n. Setelah itu, program mencetak pesan yang menunjukkan bilangan yang sedang diperiksa, kemudian memanggil fungsi cetakFaktor dengan argumen n dan 1. Setelah semua faktor dicetak, program menambahkan newline untuk membuat output lebih rapi.

Dengan demikian, pengguna dapat melihat semua faktor dari bilangan yang mereka masukkan dengan mudah.

#### Study Case 4

- 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.

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

#### Source Code

```
//2311102012
package main

import (
    "fmt"
)

// Fungsi rekursif untuk menghasilkan barisan angka
func buatBarisan(angkaAwal, angkaSekarang int) []int {
    // Jika sudah mencapai angka 1, mulai membangun barisan balik
    if angkaSekarang == 1 {
        return []int{1}
    }
    // Panggil fungsi secara rekursif untuk menghasilkan barisan
    angka
    barisan := append([]int{angkaSekarang}, buatBarisan(angkaAwal,
angkaSekarang-1)...)
    return append(barisan, angkaSekarang)
}

// Fungsi pembantu untuk mengubah slice angka menjadi string
func ubahBarisanKeString(barisan []int) string {
    hasil := ""
```

```

    for _, angka := range barisan {
        hasil += fmt.Sprintf("%d ", angka)
    }
    return hasil
}

func main() {
    var angka int
    // Meminta input dari pengguna
    fmt.Print("Masukkan sebuah angka positif: ")
    fmt.Scanln(&angka)

    // Memeriksa jika input valid
    if angka <= 0 {
        fmt.Println("Masukkan angka positif yang lebih besar dari 0.")
        return
    }

    // Memanggil fungsi dan menampilkan hasil
    fmt.Println("Hasil:", ubahBarisanKeString(buatBarisan(angka,
angka)))
}

```

## Screenshot Output

```

PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\unguided4.go"
Masukkan sebuah angka positif: 10
Hasil: 10 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9 10
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5>

```

## Deskripsi Program

Program dimulai dengan mengimpor paket "fmt" untuk mendukung operasi input dan output. Terdapat dua fungsi utama: `buatBarisan` dan `ubahBarisanKeString`.

Fungsi `buatBarisan` bertugas untuk menghasilkan barisan angka secara rekursif. Fungsi ini menerima dua parameter: `angkaAwal`, yang merupakan angka positif yang dimasukkan oleh pengguna, dan `angkaSekarang`, yang menunjukkan angka saat ini yang sedang diproses. Jika `angkaSekarang` mencapai 1, fungsi akan mengembalikan slice berisi angka 1. Jika tidak, fungsi akan memanggil dirinya sendiri dengan mengurangi `angkaSekarang`.

sebesar 1 dan menyimpan hasilnya dalam slice barisan. Dengan menggunakan append, fungsi akan menambahkan angkaSekarang ke depan dari hasil barisan yang dihasilkan. Kemudian, fungsi juga akan menambahkan angkaSekarang di akhir barisan sebelum mengembalikannya. Ini menghasilkan barisan yang mencetak angka dari angkaSekarang ke 1 dan kembali ke angkaSekarang.

Fungsi kedua, ubahBarisanKeString, berfungsi untuk mengubah slice angka menjadi sebuah string. Fungsi ini mengambil slice barisan sebagai parameter, dan untuk setiap angka dalam slice, ia menambahkan angka tersebut ke string hasil, dipisahkan oleh spasi. Setelah semua angka ditambahkan, string hasil dikembalikan.

Di dalam fungsi main, program meminta pengguna untuk memasukkan angka positif. Jika angka yang dimasukkan kurang dari atau sama dengan 0, program akan mencetak pesan peringatan dan menghentikan eksekusi. Jika input valid, program memanggil fungsi buatBarisan dan mengubah hasilnya menjadi string menggunakan ubahBarisanKeString. Akhirnya, hasil barisan dicetak ke layar.

Dengan demikian, pengguna dapat melihat barisan angka yang dimulai dari angka yang mereka masukkan, berkurang hingga 1, dan kemudian kembali ke angka yang dimasukkan.

## Study Case 5

- 5) 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



Fakultas Informatika  
School of Computing





## Source Code

```
//2311102012
package main

import (
    "fmt"
)

// Fungsi rekursif untuk mencetak bilangan ganjil dari 1 hingga n
func cetakGanjil(n, i int) {
    if i > n {
        return
    }
    if i%2 != 0 {
        fmt.Print(i, " ")
    }
    cetakGanjil(n, i+1)
}

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

    fmt.Printf("Bilangan ganjil dari 1 hingga %d adalah: ", n)
    cetakGanjil(n, 1)
    fmt.Println()
}
```

## Screenshot Output

```
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaP
rak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\unguided5.go"
Masukkan bilangan bulat positif: 20
Bilangan ganjil dari 1 hingga 20 adalah: 1 3 5 7 9 11 13 15 17 19
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> █
```

## Deskripsi Program

Program dimulai dengan mengimpor paket "fmt", yang digunakan untuk menangani input dan output. Terdapat fungsi rekursif bernama cetakGanjil yang bertugas untuk menemukan dan mencetak bilangan ganjil. Fungsi ini menerima dua parameter: n, yang merupakan bilangan bulat positif hingga mana bilangan ganjil akan dicetak, dan i, yang menunjukkan bilangan saat ini yang sedang diperiksa.

Di dalam fungsi cetakGanjil, pertama-tama ada kondisi untuk memeriksa apakah nilai i lebih besar dari n. Jika i lebih besar, fungsi akan mengembalikan kontrol tanpa melakukan apa pun, yang menandakan akhir dari proses rekursi. Jika i tidak lebih besar dari n, fungsi kemudian memeriksa apakah i adalah bilangan ganjil dengan menggunakan operator modulus (%). Jika i adalah ganjil (yaitu  $i \% 2 \neq 0$ ), maka fungsi akan mencetak nilai i diikuti oleh spasi.

Setelah memeriksa dan mencetak bilangan ganjil (jika ada), fungsi cetakGanjil kemudian dipanggil kembali dengan nilai i yang ditambah satu. Proses ini berlanjut hingga semua angka dari 1 hingga n telah diperiksa.

Di dalam fungsi main, program meminta pengguna untuk memasukkan bilangan bulat positif melalui konsol. Nilai ini disimpan dalam variabel n. Setelah itu, program mencetak pesan yang menunjukkan bilangan yang sedang diproses, dan kemudian memanggil fungsi cetakGanjil dengan argumen n dan 1. Setelah semua bilangan ganjil dicetak, program menambahkan newline untuk membuat output lebih rapi.

Dengan demikian, pengguna dapat melihat semua bilangan ganjil dari 1 hingga n yang mereka masukkan, dengan cara yang mudah dan interaktif.

## Study Case 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".

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	2 2	4
2	5 3	125

## Source Code

```
//2311102012
package main

import (
    "fmt"
)

// Fungsi rekursif untuk menghitung x pangkat y
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 bilangan basis (x): ")
    fmt.Scan(&x)
    fmt.Print("Masukkan bilangan pangkat (y): ")
    fmt.Scan(&y)

    hasil := pangkat(x, y)
    fmt.Printf("Hasil %d pangkat %d adalah: %d\n", x, y, hasil)
}
```

## ScreenShot Output

```
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5> go run "c:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5\unguided6.go"
Masukkan bilangan basis (x): 2
Masukkan bilangan pangkat (y): 5
Hasil 2 pangkat 5 adalah: 32
PS C:\olan\KULYEAH\LaPrak Alpro 2\2311102012-MaulanaGhaniRolanda-M5>
```

## Deskripsi Program

Program dimulai dengan mengimpor paket "fmt" yang digunakan untuk operasi input dan output. Di dalam program, terdapat fungsi rekursif bernama pangkat, yang bertugas untuk menghitung hasil pangkat dari dua bilangan bulat. Fungsi ini menerima dua parameter: x, yang merupakan basis, dan y, yang merupakan eksponen.

Di dalam fungsi pangkat, terdapat pengecekan untuk melihat apakah nilai y sama dengan 0. Jika ya, fungsi akan mengembalikan nilai 1, sesuai dengan hukum bahwa setiap bilangan yang dipangkatkan dengan 0 adalah 1. Jika tidak, fungsi akan mengembalikan hasil perkalian antara x dan pemanggilan fungsi pangkat dengan argumen yang sama untuk x dan y dikurangi 1. Proses ini akan terus berulang, dengan nilai y berkurang hingga mencapai 0, pada titik mana fungsi mulai kembali ke atas dalam tumpukan rekursi dan menghitung hasilnya.

Di dalam fungsi main, program meminta pengguna untuk memasukkan nilai untuk basis x dan eksponen y melalui konsol. Nilai yang dimasukkan disimpan dalam variabel x dan y. Setelah kedua nilai diperoleh, program memanggil fungsi pangkat untuk menghitung hasil  $xyx^yxy$  dan menyimpannya dalam variabel hasil. Akhirnya, program mencetak hasil pangkat tersebut ke konsol dalam format yang mudah dibaca.

Dengan demikian, pengguna dapat menghitung nilai pangkat dari bilangan yang mereka masukkan secara interaktif.