

**LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL 6
REKURSIF**



Disusun Oleh :

Zahra Tsurroya Poetri / 231110217

IF 11 - 05

Dosen Pengampu :

Arif Amrulloh

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. DASAR TEORI

1. Pengantar 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

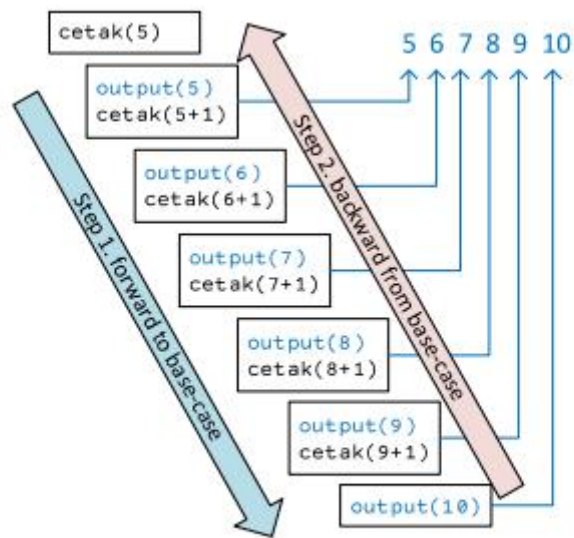
proses rekursif akan berhenti. Sebagai contoh misalnya base case adalah ketika x bernilai 10 atau 10, maka tidak perlu dilakukan rekursif.

```
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
```

Apabila diperhatikan pada baris ke-3 di Program di atas, kita telah menambahkan base-case seperti penjelasan sebelumnya. Selanjutnya pada bagian aksi dari else di baris ke 6 dan ke-7 kita namakan recursive-case atau kasus pemanggilan dirinya sendiri tersebut terjadi. Kondisi dari recursive-case ini adalah negasi dari kondisi base-case atau ketika nilai $x \neq 10$.

```
1 package main
2 import "fmt"
3 func main(){
4     cetak(5)
5 }
6 func cetak(x int){
7     if x == 10 {
8         fmt.Println(x)
9     }else{
10         fmt.Println(x)
11         cetak(x+1)
12     }
13 }
```

Apabila program di atas ini dijalankan maka akan tampil angka 5 6 7 8 9 10. Terlihat bahwa proses rekursif berhasil dihentikan ketika $x == 10$.



Gambar 1. Ilustrasi proses forward dan backward pada saat rekursif.

Pada Gambar 2 memperlihatkan saat subprogram dipanggil secara rekursif, maka subprogram akan terus melakukan pemanggilan (forward) hingga berhenti pada saat kondisi base-case terpenuhi atau true. Setelah itu akan terjadi proses backward atau kembali ke subprogram yang sebelumnya. Artinya setelah semua Instruksi cetak(10) selesai dieksekusi, maka program akan kembali ke cetak(9) yang memanggil cetak(10) tersebut. Begitu seterusnya hingga kembali ke cetak(5).

Perhatikan modifikasi program di atas dengan menukar posisi baris 10 dan 11, mengakibatkan ketika program dijalankan maka akan menampilkan 10 9 8 7 6 5. Kenapa bisa demikian? Pahami proses backward pada Gambar 2

```

1 package main
2 import "fmt"
3 func main(){
4     cetak(5)
5 }
6 func cetak(x int){
7     if x == 10 {
8         fmt.Println(x)
9     }else{
10        cetak(x+1)
11        fmt.Println(x)
12    }
13 }

```

Catatan:

- Teknik rekursif ini merupakan salah satu alternatif untuk mengganti struktur kontrol perulangan dengan memanfaatkan subprogram (bisa fungsi ataupun prosedur).
- Untuk menghentikan proses rekursif digunakan percabangan (if-then).
- Base-case adalah kondisi proses rekursif berhenti. Base-case merupakan hal terpenting dan pertama yang harus diketahui ketika akan membuat program rekursif. Mustahil membuat program rekursif tanpa mengetahui base-case terlebih dahulu.
- Recursive-case adalah kondisi dimana proses pemanggilan dirinya sendiri dilakukan. Kondisi recursive-case adalah komplemen atau negasi dari base-case.
- Setiap algoritma rekursif selalu memiliki padanan dalam bentuk algoritma liganatif.

II. GUIDED

Guided 1

Program Menampilkan Urutan Bilangan Bulat dari N

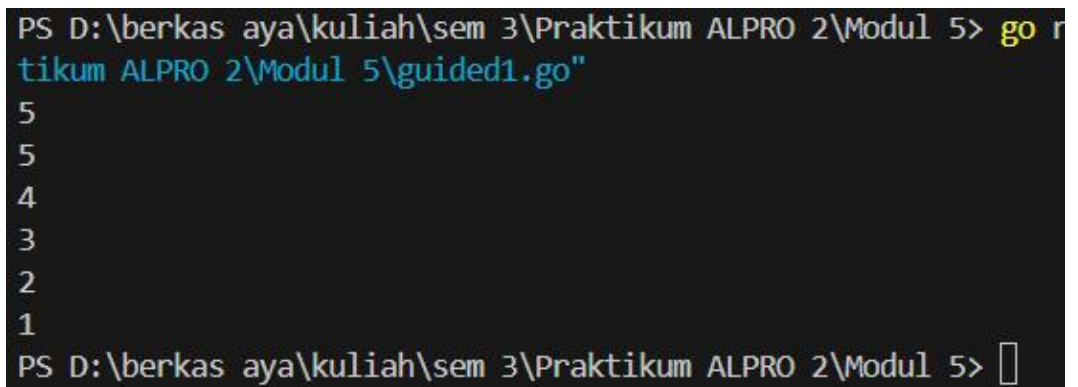
Sourcecode

```
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)  
    }  
}
```

Screenshoot Output



```
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run praktikum ALPRO 2\Modul 5\guided1.go  
5  
4  
3  
2  
1  
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> 
```

Deskripsi Program

Program di atas merupakan program untuk menampilkan urutan bilangan bulat dari **N** hingga 1. Saat pertama kali dijalankan, program akan meminta pengguna untuk memasukkan sebuah bilangan bulat positif **N**. Setelah input diterima oleh program, program akan memanggil fungsi rekursif yang bernama **baris**. Fungsi ini memiliki dua kondisi utama. Jika nilai bilangan yang diterima adalah 1, fungsi akan mencetak angka 1 dan proses akan berhenti. Namun, jika nilainya lebih besar dari 1, program akan mencetak nilai bilangan, kemudian memanggil fungsi **baris** lagi dengan mengurangi nilai bilangan sebanyak 1. Proses akan terus berlanjut sampai mencapai angka 1.

Program Menghitung Penjumlahan Menggunakan Fungsi Rekursif

Screenshot Output

Deskripsi Program

Program di atas merupakan program untuk menghitung jumlah semua bilangan bulat dari 1 hingga N menggunakan rekursif. Saat mulai

dijalankan, pengguna diminta untuk memasukkan bilangan bulat positif **N**, dan program akan memanggil fungsi rekursif bernama penjumlahan. Fungsi ini memiliki dua kondisi utama: jika **n** bernilai 1, maka fungsi mengembalikan nilai 1 sebagai hasil akhirnya. Namun, jika **n** lebih besar dari 1, fungsi akan mengembalikan hasil dari **n + penjumlahan(n-1)**, sehingga angka **n** terus berkurang hingga mencapai 1, lalu menghitung total keseluruhan bilangan tersebut. Setelah proses rekursif selesai, program akan mencetak hasil dari penjumlahan yang telah diinputkan.

Guided 3

Program Menghitung Pangkat (N) dari 2

Sourcecode

```
package main
import "fmt"

// Fungsi rekursif untuk menghitung 2^n
func pangkat(n int) int {
    if n == 0 {
        return 1
    } else {
        return 2 * pangkat(n-1)
    }
}

func main(){
    var n int
    fmt.Print("Masukkan nilai n:")
    fmt.Scan(&n)
    fmt.Println("Hasil dari 2 pangkat", n, "adalah: ",
pangkat(n))
}
```


Screenshoot Output

```
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go
aktikum ALPRO 2\Modul 5\tempCodeRunnerFile.go"
Masukkan nilai n: 8
Hasil dari 2 pangkat 8 adalah: 256
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> █
```

Deskripsi Program

Program di atas merupakan program untuk menghitung hasil dari 2^n menggunakan fungsi rekursif. Program ini meminta pengguna untuk memasukkan nilai bilangan bulat positif n , dan kemudian menghitung hasil 2^n dengan memanggil fungsi `pangkat`. Fungsi pangkat bekerja secara rekursif. Jika n bernilai 0, fungsi mengembalikan nilai 1, karena $2^0 = 1$. Jika n lebih besar dari 0, fungsi akan mengalikan 2 dengan hasil pemanggilan `pangkat(n-1)`, sehingga setiap kali pangkat dipanggil, nilai n berkurang 1. Program akan mencetak hasil dari 2 dipangkat nilai bilangan bulat positif yang sudah dimasukkan oleh pengguna.

Guided 4

Program Menghitung Pangkat (N) dari 2

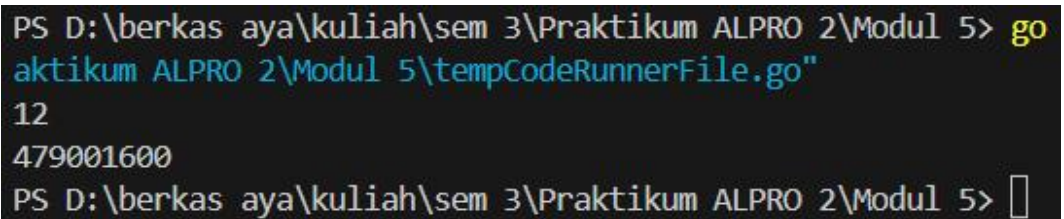
Sourcecode

```
package main
import "fmt"

var n int
func faktorial (n int) int {
    if n == 0 || n == 1 {
        return 1
    } else {
        return n * faktorial(n-1)
    }
}
```

```
func main() {  
    fmt.Scan(&n)  
    fmt.Println(faktorial(n))  
}
```

Screenshoot Output



```
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go  
aktikum ALPRO 2\Modul 5\tempCodeRunnerFile.go"  
12  
479001600  
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> █
```

Deskripsi Program

Program di atas merupakan program untuk menghitung nilai faktorial dari bilangan bulat positif `n` menggunakan pendekatan rekursif. Program ini mendeklarasikan variabel `n` secara global dan meminta input nilai `n` dari pengguna di dalam fungsi `main`. Setelah menerima input, program memanggil fungsi `faktorial` untuk menghitung faktorial dari nilai tersebut. Fungsi `faktorial` memiliki dua kondisi utama: base case dan rekursif. Jika `n` adalah 0 atau 1, fungsi mengembalikan `1`. Jika `n` lebih dari 1, fungsi mengalihkan `n` dengan hasil `faktorial(n-1)`, memanggil dirinya sendiri dengan nilai `n` yang terus berkurang.

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.

Sourcecode

```
package main

import "fmt"

// Fungsi rekursif
func S(n int) int {
    // Base case: Jika n adalah sama dengan 1, maka akan
    // mengembalikan nilai n
    if n <= 1 {
        return n
    }
    // Rekursif case:
    return S(n-1) + S(n-2) // Fungsi S melakukan rekursif
    // untuk menghitung S(n-1) dan S(n-2)
}

func main() {
    var n int = 10 // Inisialisasi variabel untuk
    // menyimpan jumlah suku

    fmt.Printf("Deret Fibonacci hingga suku ke-%d:\n", n)
    // Menampilkan pesan
    for i := 0; i <= n; i++
```

```

{
    // Loop untuk mencetak
    setiap suku dari 0 hingga n
    fmt.Printf("Suku ke-%d: %d\n", i, S(i)) //
    Mencetak suku ke-i dan nilai menggunakan fungsi S(i)
}
}

```

Screenshoot Output

```

PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run "d:\berkas aya\kuliah\sem
Deret Fibonacci hingga suku ke-10:
0      1      2      3      4      5      6      7      8      9      10
0      1      1      2      3      5      8      13     21     34     55
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5>

```

Deskripsi Program

Program ini merupakan program untuk menghitung dan menampilkan deret Fibonacci hingga suku ke- n dengan pendekatan rekursif. Di dalam program ini, fungsi S didefinisikan untuk menghitung suku ke- n dalam deret Fibonacci dengan memanggil dirinya sendiri. Fungsi S memiliki dua bagian utama: base case dan rekursif case. Pada base case, jika nilai n adalah 0 atau 1, fungsi akan mengembalikan nilai n tersebut, karena dalam deret Fibonacci, suku pertama (0) dan kedua (1) adalah angka dasar yang tidak memerlukan perhitungan lebih lanjut. Pada rekursif case, fungsi S memanggil dirinya sendiri untuk menghitung nilai $S(n-1) + S(n-2)$, yaitu jumlah dari dua suku sebelumnya dalam deret Fibonacci. Pada bagian fungsi utama, variabel n diinisialisasi dengan nilai 10, yang berarti program akan menghitung dan menampilkan deret Fibonacci hingga suku ke-10. Dalam loop `for`, fungsi $S(i)$ dipanggil untuk setiap nilai i dari 0 hingga 10, dan hasilnya dicetak dengan urutan suku dan nilai masing-masing. Hasil akhirnya adalah deret Fibonacci hingga suku ke-10.

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
package main

import "fmt"

// Fungsi rekursif untuk mencetak bintang
func cetakBintang(n int) {
    // Base case: berhenti ketika n = 0
    if n == 0 {
        return
    }
    cetakBintang(n - 1) // Memanggil fungsi rekursif
    untuk mencetak baris sebelumnya
    for i := 0; i < n; i++ {
        fmt.Print("*") // Mencetak baris dengan n
        bintang, setelah baris sebelumnya tercetak
    }
}
```

```

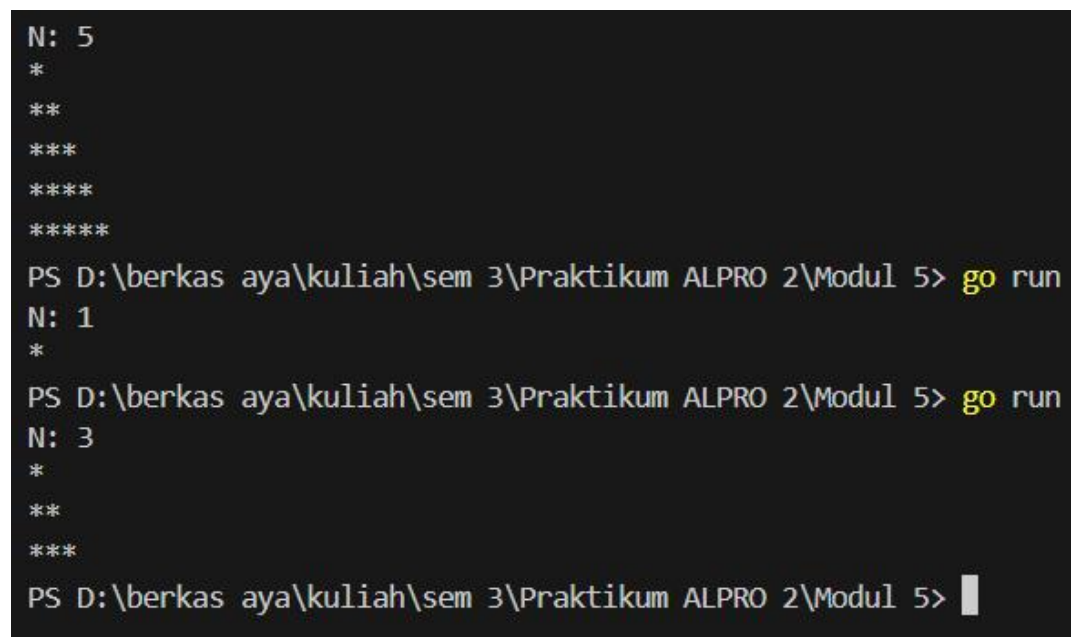
    }
    fmt.Println()
}

func main() {
    var n int // Mendeklarasikan variabel nilai N dengan
    tipe data integer
    fmt.Print("N: ") // Pengguna menginputkan nilai N
    fmt.Scan(&n)

    cetakBintang(n)
}

```

Screenshoot Output



```

N: 5
*
**
***
****
*****

PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
N: 1
*

PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
N: 3
*
**
***

PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5>

```

Deskripsi Program

Program di atas merupakan program untuk mencetak pola segitiga bintang menggunakan fungsi rekursif. Program ini meminta input integer **n** dari pengguna. Base case terjadi ketika nilai **n** mencapai 0, yang mengakhiri

proses pencetakan. Sebelum mencetak bintang untuk nilai `n` saat ini, `cetakBintang` pertama-tama memanggil dirinya sendiri dengan nilai `n-1`, sehingga setiap baris dengan jumlah bintang lebih sedikit tercetak lebih dulu. Setelah kembali dari pemanggilan rekursif, loop for mencetak `n` bintang dalam satu baris. Dengan demikian, setiap pemanggilan rekursif mencetak membentuk pola segitiga, di mana jumlah bintang bertambah dengan bertambahnya baris.

3. Soal Studi Case

Buatlah program yang mengimplementasikan rekursif untuk menampilkan faktor bilangan dan 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`).

Sourcecode

```
package main

import "fmt"

// Fungsi rekursif untuk mencari dan mencetak faktor dari
// N
func cariFaktor(n, habisBagi int) {
    // Base case: jika habisBagi melebihi N, maka proses
    // berhenti
    if habisBagi > n {
        return
    }
    // Cek apakah habisBagi adalah faktor dari N
    if n % habisBagi == 0 {
        fmt.Print(habisBagi, " ")
    }
}
```

```

        // Memanggil rekursif dengan menambah habisBagi
        cariFaktor(n, habisBagi+1)
    }

func main() {
    var N int // Mendeklarasikan variabel nilai N dengan
    tipe data integer
    fmt.Print("Masukkan bilangan bulat positif N: ") //
    Pengguna memasukkan bilangan bulat positif N
    fmt.Scan(&N)

    fmt.Print("Hasil: ") // Cetak hasil
    cariFaktor(N, 1)
    fmt.Println()
}

```

Screenshoot Output

```

PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
Masukkan bilangan bulat positif N: 5
Hasil: 1 5
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
Masukkan bilangan bulat positif N: 12
Hasil: 1 2 3 4 6 12
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5>

```

Deskripsi Program

Program di atas merupakan program untuk mencari dan mencetak faktor dari suatu bilangan bulat positif menggunakan fungsi rekursif. Pada program ini, terdapat fungsi `cariFaktor` yang menerima dua parameter: `n`, sebagai bilangan yang akan dicari faktornya, dan `habisBagi`, yang berfungsi sebagai penanda apakah bilangan tersebut adalah faktor dari `N`. Program pertama kali memeriksa base case, yaitu jika `habisBagi` melebihi nilai `N`, yang menandakan bahwa faktor sudah ditemukan, dan rekursif

berhenti. Jika `habisBagi` adalah faktor dari `N` (dengan sisa bagi nol), nilai tersebut akan dicetak ke layar. Selanjutnya, fungsi rekursif `cariFaktor` dipanggil kembali dengan menambah nilai `habisBagi` sebesar 1 untuk memeriksa angka berikutnya hingga faktor ditemukan. Proses ini dilakukan ketika pengguna telah memasukkan bilangan `n` yang diminya oleh program. Kemudian program akan mencetak seluruh faktor dalam satu baris.

4. 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`.

Sourcecode

```
package main

import "fmt"

func baris(n int) {
    // Base case: jika nilai n adalah 1, maka proses
    berhenti
    if n == 1 {
        fmt.Print(n, " ")
        return // Mengembalikan fungsi ke pemanggilan
        sebelumnya
    }

    // Menampilkan bilangan menurun
    fmt.Print(n, " ")
    baris(n - 1) // Melakukan iterasi menurun (nilai n
    berkurang 1 setiap kali)

    // Menampilkan bilangan meningkat
    fmt.Print(n, " ") // Mencetak nilai n setiap tahap
    meningkat
}
```

```

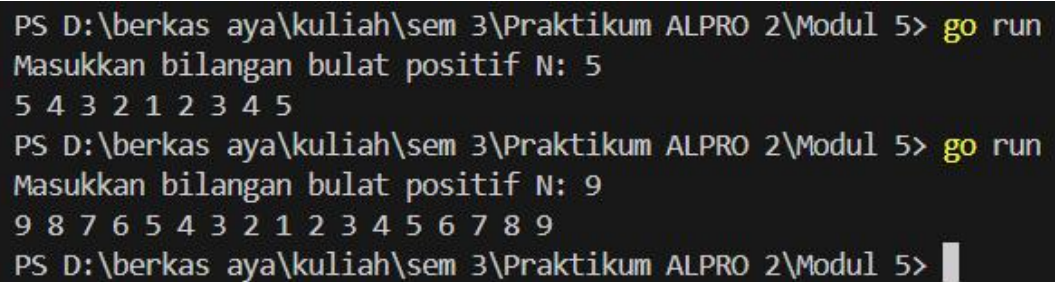
}

func main() {
    var N int // Mendeklarasikan variabel nilai N dengan
    tipe data integer
    fmt.Print("Masukkan bilangan bulat positif N: ")
    fmt.Scan(&N)

    // Memastikan input N adalah bilangan bulat positif
    if N > 0 {
        baris(N)
    } else {
        fmt.Println("Mohon masukkan bilangan bulat
        positif.") // Jika input N bukan bilangan bulat positif
    }
}

```

Screenshoot Output



```

PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
Masukkan bilangan bulat positif N: 5
5 4 3 2 1 2 3 4 5
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
Masukkan bilangan bulat positif N: 9
9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5>

```

Deskripsi Program

Program di atas merupakan kode program untuk mencetak baris bilangan secara menurun dan kemudian kembali meningkat, menggunakan rekursif. Fungsi `baris` menerima parameter `n`, yang mewakili angka awal yang ingin dicetak. Pada awalnya, fungsi memeriksa base case: jika nilai `n` adalah 1, maka fungsi mencetak angka 1 dan proses rekursif berhenti. Selanjutnya, bilangan `n` dicetak terlebih dahulu untuk membentuk urutan menurun, dan

`baris(n - 1)` dipanggil secara rekursif, mengurangi `n` hingga mencapai 1. Setelah proses penurunan selesai, setiap tahap rekursif kembali ke tahap sebelumnya, mencetak nilai `n` lagi untuk membentuk urutan meningkat. Di dalam fungsi utama, pengguna diminta memasukkan bilangan `N`, dan program hanya menjalankan fungsi `baris` jika `N` adalah bilangan bulat positif, memastikan input valid. Output akhir akan menampilkan bilangan menurun dari `N` hingga 1, lalu meningkat kembali hingga `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`.

Sourcecode

```
package main

import "fmt"

func bilGanjil(n int) {
    // Base case: jika n lebih dari 1, maka proses
    berhenti
    if n < 1 {
        return
    }

    // Jika n adalah genap, kurangi n dengan 1, maka
    bilangan menjadi ganjil terbesar di bawah n
    if n % 2 == 0 {
        n -= 1
    }

    bilGanjil(n - 2) // Memanggil rekursif untuk bilangan
    ganjil dari 1 hingga n-2
}
```

```

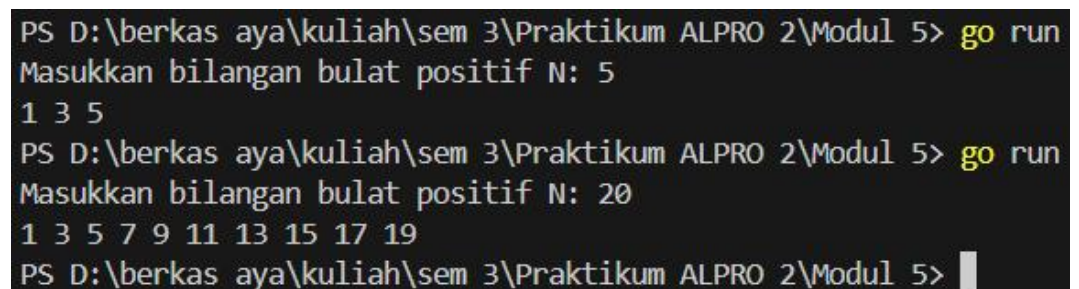
        fmt.Print(n, " ") // Cetak
    }

func main() {
    var N int // Mendeklarasikan variabel nilai N dengan
    tipe data integer
    fmt.Print("Masukkan bilangan bulat positif N: ")
    fmt.Scan(&N)

    // Memastikan input N adalah bilangan bulat positif
    if N > 0 {
        bilGanjil(N)
    } else {
        fmt.Println("Mohon masukkan bilangan bulat
        positif.") // Jika input N bukan bilangan bulat positif
    }
}

```

Screenshoot Output



```

PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
Masukkan bilangan bulat positif N: 5
1 3 5
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
Masukkan bilangan bulat positif N: 20
1 3 5 7 9 11 13 15 17 19
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5>

```

Deskripsi Program

Program di atas merupakan program untuk mencetak semua bilangan ganjil positif yang lebih kecil dari bilangan bulat positif **N** yang dimasukkan oleh pengguna. Fungsi **bilGanjil** menerima parameter **n** yang akan diproses oleh program. Di dalam fungsi, pertama-tama terdapat base

case yang memeriksa apakah n kurang dari 1; jika iya, fungsi akan berhenti. Selanjutnya, jika n adalah bilangan genap, n akan dikurangi 1 untuk memastikan kita bekerja dengan bilangan ganjil. Kemudian, fungsi dipanggil secara rekursif dengan $n - 2$, yang akan mencetak semua bilangan ganjil dari n hingga 1. Setelah panggilan rekursif selesai, bilangan n dicetak. Dengan demikian, output dari program ini adalah deretan bilangan ganjil yang lebih kecil dari N dan ditampilkan dalam urutan dari yang terbesar hingga yang terkecil.

6. Soal Studi Case

Buatlah program yang mengimplementasikan rekursif untuk mencari hasil pangkat dari dua buah bilangan.

Masukan terdiri dari bilangan bulat dan y .

Keluaran terdiri dari hasil x dipangkatkan y .

Catatan: diperbolehkan menggunakan asterik "*", tapi dilarang menggunakan Import "math".

Sourcecode

```
package main

import "fmt"

// Fungsi rekursif untuk menghitung x pangkat y
func pangkat(x int, y int) int {
    // Base case: jika y adalah 0, maka hasilnya 1 ( $x^0 = 1$ )
    if y == 0 {
        return 1
    }

    // Base case: jika y adalah 1, maka hasilnya x ( $x^1 = x$ )
}
```

```
    if y == 1{
        return x
    }

    // Jika y kurang dari 0 atau negatif, maka hitug
    pangkat positif dan menjadi kebalikannya
    if y < 0 {
        return 1 / pangkat(x, -y) // Menggunakan
    pembagian untuk menangani pangkat negatif
    }

    return x * pangkat(x, y-1) // Memanggil rekursif
    untuk menghitung x pangkat (y - 1) dan kalikan dengan x
}

func main() {
    var x, y int // Mendeklarasikan nilai x dan y dengan
    tipe data integer
    fmt.Print("Masukkan bilangan bulat x: ")
    fmt.Scan(&x)
    fmt.Print("Masukkan bilangan bulat y: ")
    fmt.Scan(&y)

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

Screenshoot Output

```
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
Masukkan bilangan bulat x: 2
Masukkan bilangan bulat y: 2
2 pangkat 2 adalah 4
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> go run
Masukkan bilangan bulat x: 5
Masukkan bilangan bulat y: 3
5 pangkat 3 adalah 125
PS D:\berkas aya\kuliah\sem 3\Praktikum ALPRO 2\Modul 5> |
```

Deskripsi Program

Program di atas merupakan kode program untuk menghitung nilai x pangkat y menggunakan pendekatan rekursif. Fungsi utama yang digunakan adalah pangkat, yang menerima dua parameter, yaitu x dan y . Di dalam fungsi ini, terdapat beberapa base case untuk menangani situasi tertentu: jika y sama dengan 0, fungsi akan mengembalikan 1 karena $x^0 = 1$. jika y sama dengan 1, fungsi akan mengembalikan x itu sendiri karena $x^1 = x$. Jika y adalah bilangan negatif, fungsi akan menghitung nilai pangkat positif dari x dengan cara membalikkan hasilnya, yaitu menggunakan pembagian $1/\text{pangkat}(x, -y)$. Jika tidak ada kondisi dasar yang terpenuhi, fungsi akan melanjutkan dengan memanggil dirinya sendiri secara rekursif, mengurangi nilai y sebanyak 1, dan mengalikan hasilnya dengan x . Kemudian hasilnya akan dicetak, menyatakan hasil dari x pangkat y .

Daftar Pustaka

- [1] Susilowati, A. Zahara, A. (2024). *Modul 5 Praktikum Algoritma Pemrograman 2*. Program Studi Teknik Informatika, Telkom University Purwokerto.