

**LAPORAN PRAKTIKUM
ALGORITMA PEMROGRAMAN 2**

**MODUL V
REKURSIF**



Oleh:

RIZKULLOH ALPRIYANSAH

2311102142

IF-11-08

**PROGRAM STUDI S1 INFORMATIKA
FAKULTAS INFORMATIKA
UNIVERSITAS TELKOM PURWOKERTO**

2024

I. DASAR TEORI

1. Pengertian Rekursif

Rekursif merupakan teknik dalam pemrograman di mana sebuah fungsi atau prosedur dapat memanggil dirinya sendiri untuk menyelesaikan suatu masalah. Metode ini membagi masalah menjadi sub-masalah yang lebih kecil dan serupa dengan masalah asal. Setiap pemanggilan fungsi rekursif akan mendekatkan masalah ke kondisi akhir atau kondisi dasar (base case) yang akan menghentikan proses rekursif.

2. Komponen Rekursif

Secara umum, setiap algoritma rekursif terdiri dari dua bagian utama:

1. Base Case

Ini adalah kondisi yang menentukan kapan proses rekursi harus berhenti. Base case sangat penting untuk mencegah rekursi berjalan terus menerus tanpa henti. Tanpa adanya base case, fungsi rekursif akan terus memanggil dirinya sendiri tanpa batas, yang dapat menyebabkan stack overflow atau menghentikan program secara tidak normal.

2. Recursive Case

Ini adalah bagian di mana fungsi memanggil dirinya sendiri dengan kondisi atau nilai yang lebih sederhana, mendekati base case. Bagian ini adalah inti dari rekursif karena di sini fungsi menyelesaikan masalah utama dengan memecahnya menjadi sub-masalah.

3. Cara Kerja Rekursif

Ini adalah bagian di mana fungsi memanggil dirinya sendiri dengan kondisi atau nilai yang lebih sederhana, mendekati base case. Bagian ini adalah inti dari rekursif karena di sini fungsi menyelesaikan masalah utama dengan memecahnya menjadi sub-masalah.

- **Forward**

Fungsi terus memanggil dirinya sendiri dengan nilai yang semakin kecil atau lebih sederhana hingga mencapai base case.

- **Backward**

Setelah mencapai base case, fungsi akan mengembalikan hasilnya ke pemanggilan sebelumnya, sehingga mulai

menyelesaikan dan menggabungkan hasil dari setiap pemanggilan hingga kembali ke pemanggilan awal.

4. Contoh Dasar Rekursif

1. Menampilkan Deret Bilangan dari n hingga 1

- Base case: Ketika bilangan == 1
- Pada setiap pemanggilan, bilangan akan berkurang satu hingga mencapai base case.

```
func deret(bilangan int) {  
    if bilangan == 1 {  
        fmt.Println(1)  
    } else {  
        fmt.Println(bilangan)  
        deret(bilangan - 1)  
    }  
}
```

2. Penjumlahan Bilangan dari 1 hingga n

- Base case: Ketika n = 1
- Fungsi akan menambahkan n ke hasil rekursif dari penjumlahan hingga n menjadi 1.

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

3. Menghitung Pangkat n dari 2 (2^n)

- Base case: Ketika n = 0, hasilnya adalah 1.
- Setiap pemanggilan rekursif mengalikan 2 dengan hasil rekursif dari 2^{n-1} .

```
func pangkat(n int) int {  
    if n == 0 {  
        return 1  
    } else {  
        return 2 * pangkat(n - 1)  
    }  
}
```

4. Faktorial dari n ($n!$)

- Base case: Ketika n = 0 atau n = 1, hasilnya adalah 1.

- Fungsi rekursif mengalikan n dengan hasil rekursif dari $(n-1)!$.

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

I. GUIDED

Guided1

Source Code

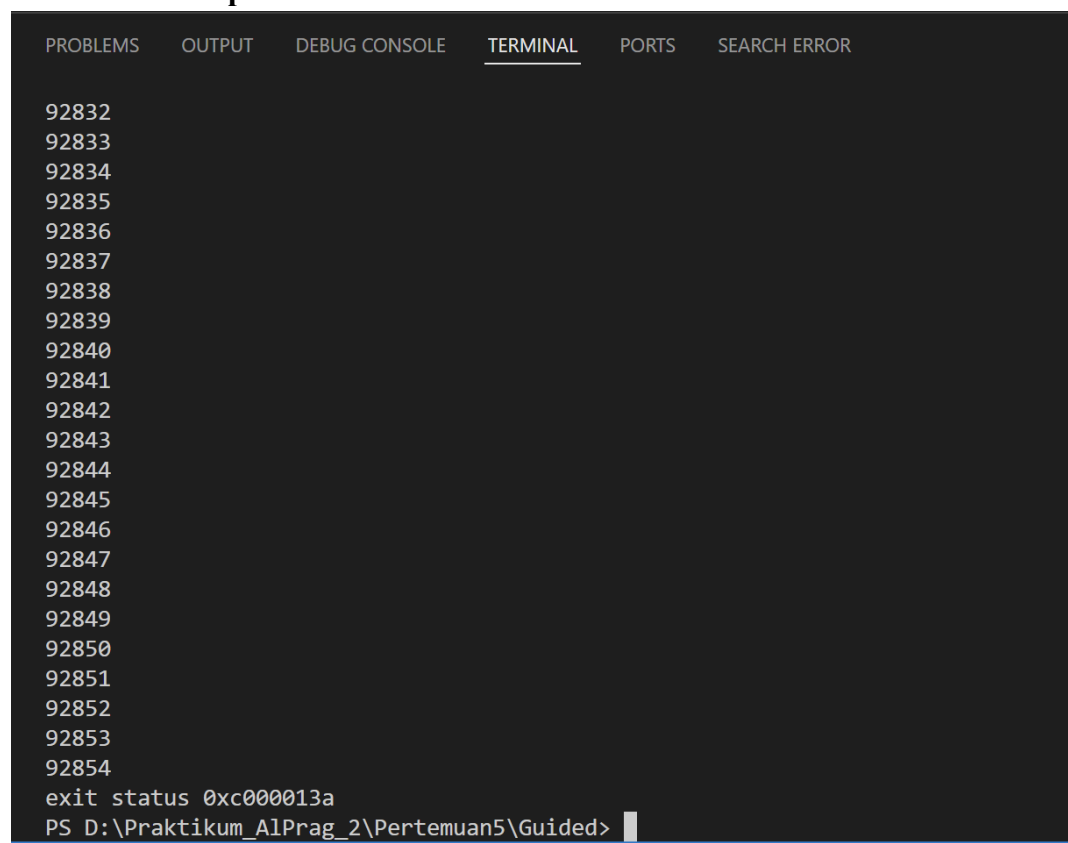
```
package main

import "fmt"

func main() {
    cetak2(5)
}

func cetak2(x int) {
    fmt.Println(x)
    cetak2(x + 1)
}
```

ScreenShot Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

92832
92833
92834
92835
92836
92837
92838
92839
92840
92841
92842
92843
92844
92845
92846
92847
92848
92849
92850
92851
92852
92853
92854
exit status 0xc000013a
PS D:\Praktikum_AlPrag_2\Pertemuan5\Guided>
```

Deskripsi

Kode ini punya fungsi utama “main” yang memanggil fungsi rekursif “cetak2” dengan nilai awal “5”. Fungsi “cetak2” nge-print nilai “x” terus-terusan sambil nambahin “x” satu per satu setiap kali dia manggil dirinya sendiri lagi. Karena

ga ada kondisi buat stop, program ini bakal jalan terus sampe akhirnya ngalami "stack overflow," alias error karena rekursinya ga pernah berhenti.

Guided2

Source Code

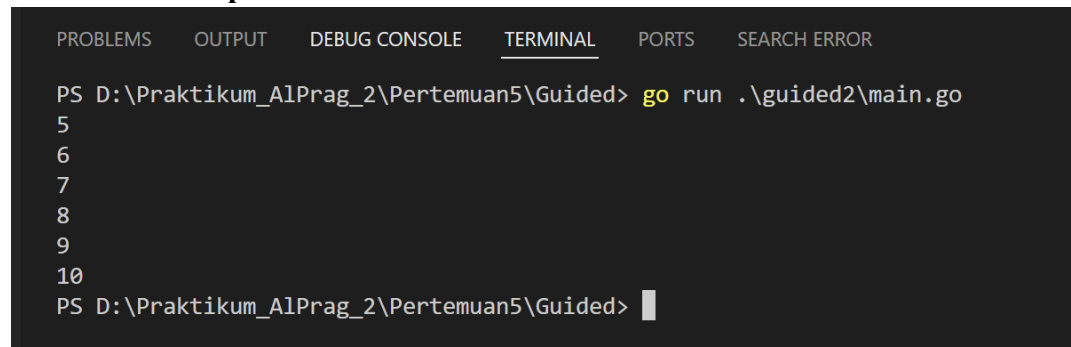
```
package main

import "fmt"

func main() {
    cetak(5)
}

func cetak(x int) {
    if x == 10 {
        fmt.Println(x) //base case
    } else {
        fmt.Println(x)
        cetak(x + 1)
    }
}
```

ScreenShot Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\Praktikum_AlPrag_2\Pertemuan5\Guided> go run .\guided2\main.go
5
6
7
8
9
10
PS D:\Praktikum_AlPrag_2\Pertemuan5\Guided> █
```

Deskripsi

Kode ini punya fungsi utama “main” yang manggil fungsi “cetak” dengan nilai awal “5”. Di dalam fungsi “cetak”, ada kondisi “base case” buat ngecek apakah nilai “x” udah sampe “10”. Kalau “x” sama dengan “10”, program nge-print “x” dan berhenti di situ. Tapi kalau “x” masih di bawah “10”, program bakal nge-print nilai “x” dan manggil lagi fungsi “cetak” dengan “x” ditambah “1”. Jadi, fungsi ini jalan rekursif sampe nilai “x” mencapai “10”, baru programnya stop.

Guided3

Source Code

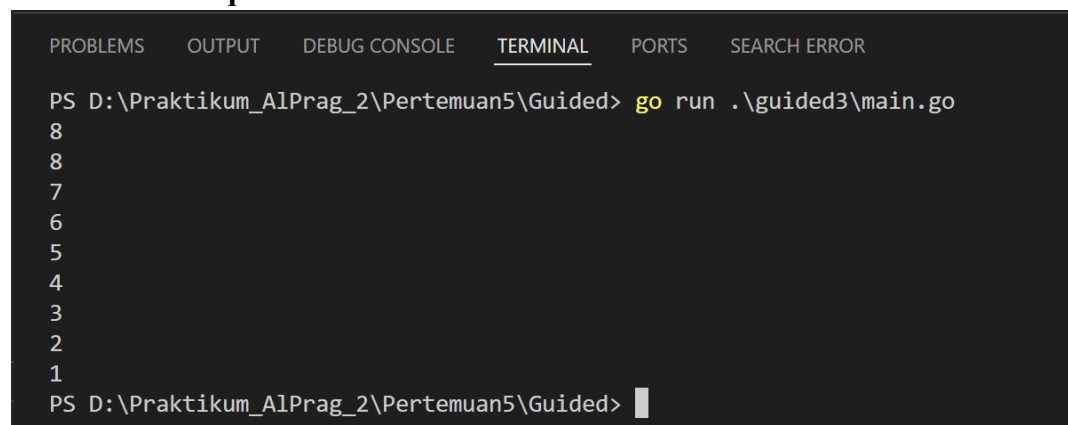
```
package main

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)
    }
}
```

ScreenShot Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\Praktikum_AIPrag_2\Pertemuan5\Guided> go run .\guided3\main.go
8
8
7
6
5
4
3
2
1
PS D:\Praktikum_AIPrag_2\Pertemuan5\Guided> █
```

Deskripsi

Kode ini dimulai di fungsi utama "main" yang meminta input angka "n" dari user menggunakan "fmt.Scan(&n)", lalu memanggil fungsi "baris" dengan "n" sebagai argumen. Di fungsi "baris", ada kondisi untuk mengecek apakah "bilangan" udah sama dengan "1"; kalau iya, program nge-print "1" dan berhenti. Tapi kalau "bilangan" lebih dari "1", program nge-print nilai "bilangan" sekarang, terus panggil lagi fungsi "baris" dengan "bilangan - 1", jadi fungsi ini bakal nge-print angka dari "n" turun sampe "1" dan berhenti.

Guided4

Source Code

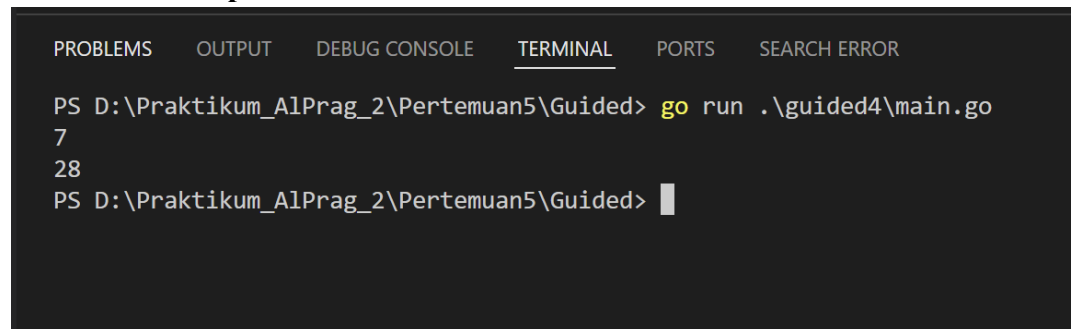
```
package main

import "fmt"

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

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

ScreenShot Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\Praktikum_AIPrag_2\Pertemuan5\Guided> go run .\guided4\main.go
7
28
PS D:\Praktikum_AIPrag_2\Pertemuan5\Guided> 
```

Deskripsi

Kode ini dimulai di fungsi utama "main" yang minta input angka "n" dari user pake "fmt.Scan(&n)", lalu panggil fungsi "penjumlahan" buat menghitung jumlah total dari angka "n" sampai "1". Di fungsi "penjumlahan", ada kondisi base case buat ngecek apakah "n" udah sama dengan "1"; kalau iya, fungsi bakal return "1". Kalau belum, fungsi bakal return hasil dari "n + penjumlahan(n-1)", artinya nambahin "n" sekarang sama hasil penjumlahan dari "n-1". Ini jalan terus rekursif sampai ketemu base case, ngembaliin total penjumlahan semua angka dari "n" sampai "1".

Guided5

Source Code

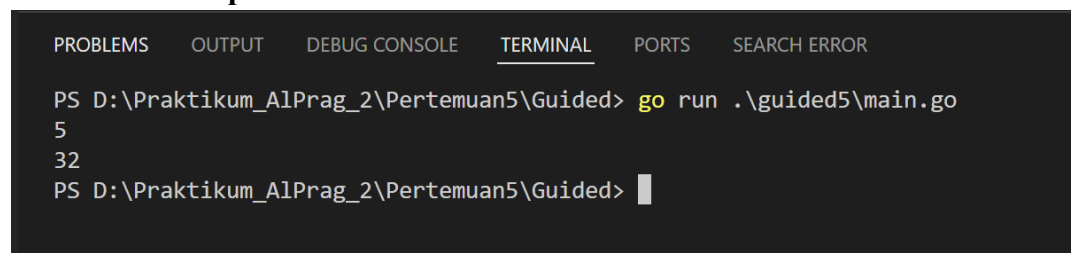
```
package main

import "fmt"

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

func pangkat(n int) int {
    if n == 0 {
        return 1
    } else {
        return 2 * pangkat(n-1)
    }
}
```

ScreenShot Output



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\Praktikum_AlPrag_2\Pertemuan5\Guided> go run .\guided5\main.go
5
32
PS D:\Praktikum_AlPrag_2\Pertemuan5\Guided> 
```

Deskripsi

Kode ini dimulai di fungsi utama "main" yang meminta input angka "n" dari user pake "fmt.Scan(&n)", lalu panggil fungsi "pangkat" buat ngitung hasil dari 2 pangkat "n". Di fungsi "pangkat", ada kondisi base case buat ngecek apakah "n" udah sama dengan "0"; kalau iya, fungsi bakal return "1" (karena 2 pangkat 0 itu 1). Kalau belum, fungsi bakal return hasil dari "2 * pangkat(n-1)", artinya ngaliin 2 dengan hasil pangkat dari "n-1". Ini jalan terus secara rekursif sampai "n" mencapai 0, dan hasil akhirnya jadi nilai 2 pangkat "n".

Guided6

Source Code

```
package main
```

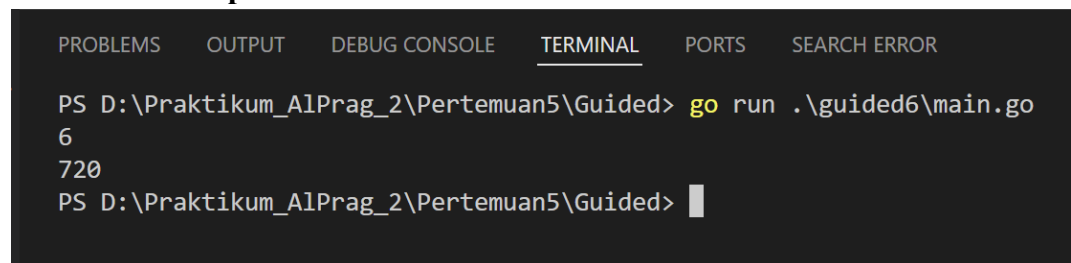
```
import "fmt"

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

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

}
```

ScreenShot Output

A screenshot of a Go IDE's terminal window. The terminal has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is active), PORTS, and SEARCH ERROR. The command prompt shows the user is in the directory D:\Praktikum_AIPrag_2\Pertemuan5\Guided. They run the command 'go run .\guided6\main.go'. The program outputs the number '6' on the first line and '720' on the second line. The prompt then returns to 'PS D:\Praktikum_AIPrag_2\Pertemuan5\Guided>' with a cursor.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\Praktikum_AIPrag_2\Pertemuan5\Guided> go run .\guided6\main.go
6
720
PS D:\Praktikum_AIPrag_2\Pertemuan5\Guided> 
```

Deskripsi

Kode ini dimulai di fungsi utama "main" yang minta input angka "n" dari user pake "fmt.Scan(&n)", lalu panggil fungsi "faktorial" buat ngitung nilai faktorial dari "n". Di fungsi "faktorial", ada kondisi base case buat ngecek apakah "n" sama dengan "0" atau "1"; kalau iya, fungsi bakal return "1" (karena faktorial dari 0 dan 1 itu 1). Kalau belum, fungsi bakal return hasil dari "n * faktorial(n - 1)", artinya ngaliin "n" dengan hasil faktorial dari "n-1". Proses ini berjalan secara rekursif sampai mencapai base case, dan akhirnya ngasilin nilai faktorial dari "n".

II. UNGUIDED

Unguided1

Source Code

```
package main

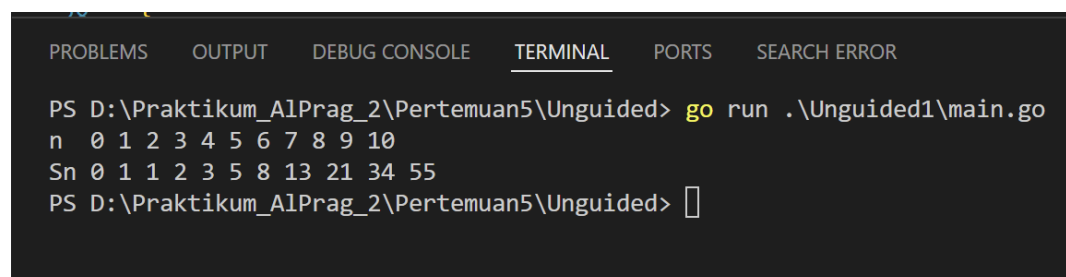
import (
    "fmt"
)

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() {
    fmt.Print("n ")
    for i := 0; i <= 10; i++ {
        fmt.Printf("%d ", i)
    }
    fmt.Println()

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

Screenshot



The screenshot shows a terminal window with the following content:

```
PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided1\main.go
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:\Praktikum_AlPrag_2\Pertemuan5\Unguided> 
```

The terminal window has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (selected), PORTS, and SEARCH ERROR.

Deskripsi

Kode ini menghitung deret Fibonacci dengan fungsi rekursif. Fungsi "fibonacci" menerima parameter "n" dan mengecek kondisi base case; kalau "n" sama dengan "0", fungsi bakal return "0", dan kalau "n" sama dengan "1", fungsi bakal return "1". Kalau belum, fungsi bakal return hasil penjumlahan dari "fibonacci(n-1)" dan "fibonacci(n-2)", yang berarti ngitung nilai Fibonacci untuk dua angka sebelumnya. Di fungsi "main", program nge-print angka dari "0" sampai "10" untuk nilai "n", terus nge-print nilai Fibonacci dari "0" sampai "10" dengan memanggil fungsi "fibonacci" untuk setiap angka. Outputnya bakal nunjukkin angka Fibonacci yang sesuai dengan urutan deretnya.

Unguided2

Source Code

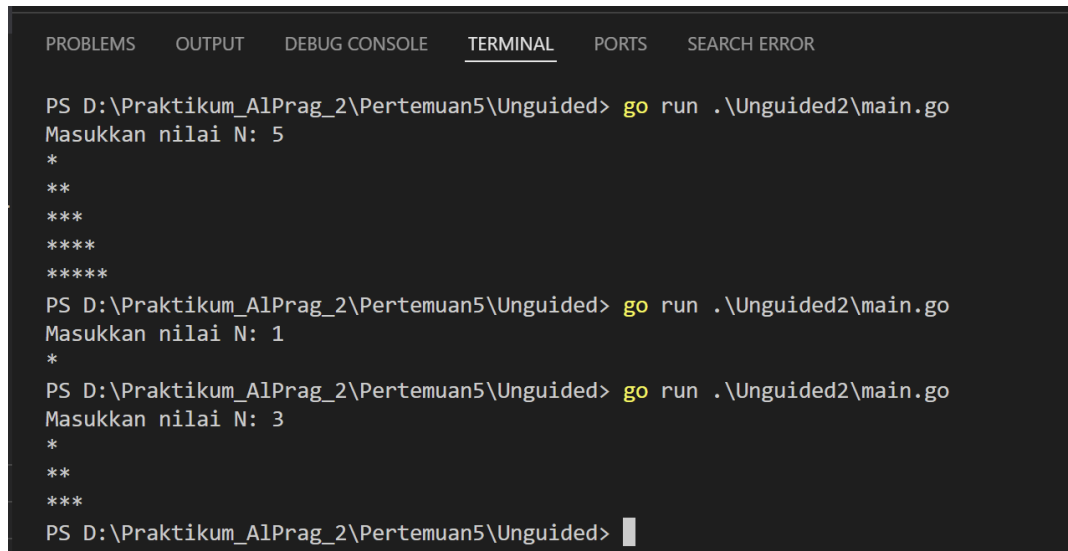
```
package main

import (
    "fmt"
)

func printStars(n_142, current_142 int) {
    if current_142 > n_142 {
        return
    }
    for i := 0; i < current_142; i++ {
        fmt.Print("*")
    }
    fmt.Println()
    printStars(n_142, current_142+1)
}

func main() {
    var n_142 int
    fmt.Print("Masukkan nilai N: ")
    fmt.Scan(&n_142)
    printStars(n_142, 1)
}
```

Screenshot



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided2\main.go
Masukkan nilai N: 5
*
**
***
****
*****

PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided2\main.go
Masukkan nilai N: 1
*

PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided2\main.go
Masukkan nilai N: 3
*
**
***

PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> 
```

Deskripsi

Kode ini mencetak pola bintang dengan jumlah bintang yang meningkat di setiap baris. Fungsi "printStars" menerima dua parameter: "n_142" (jumlah maksimum baris) dan "current_142" (baris saat ini). Pertama, fungsi mengecek apakah "current_142" lebih besar dari "n_142"; kalau iya, fungsi bakal return dan berhenti. Jika belum, fungsi nge-print bintang sebanyak "current_142" di baris itu dengan loop "for", terus pindah ke baris baru. Setelah itu, fungsi memanggil dirinya sendiri dengan "current_142" yang ditambah satu, sehingga pola bintang bakal terus dicetak sampai "current_142" mencapai "n_142". Di fungsi "main", program minta input dari user buat nilai "N", lalu panggil fungsi "printStars" dengan nilai "N" dan "1" sebagai argumen awal.

Unguided3

Source Code

```
package main

import (
    "fmt"
)

func printFactors(n_142, current_142 int) {
```

```

    if current_142 > n_142 {
        return
    }
    if n_142%current_142 == 0 {
        fmt.Print(current_142, " ")
    }
    printFactors(n_142, current_142+1)
}

func main() {
    var n_142 int
    fmt.Print("Masukkan nilai N: ")
    fmt.Scan(&n_142)

    fmt.Println("Faktor dari", n_142, ":")
    printFactors(n_142, 1)
    fmt.Println()
}

```

Screenshot

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

PS D:\Praktikum_AIPrag_2\Pertemuan5\Unguided> go run .\Unguided3\main.go
Masukkan nilai N: 5
Faktor dari 5 :
1 5
PS D:\Praktikum_AIPrag_2\Pertemuan5\Unguided> go run .\Unguided3\main.go
Masukkan nilai N: 12
Faktor dari 12 :
1 2 3 4 6 12
PS D:\Praktikum_AIPrag_2\Pertemuan5\Unguided>

```

Deskripsi

Kode ini mencari dan mencetak faktor dari bilangan yang dimasukkan oleh user. Fungsi "printFactors" adalah fungsi rekursif yang menerima dua parameter: "n_142" (bilangan yang akan dicari faktornya) dan "current_142" (bilangan yang sedang diperiksa untuk menjadi faktor). Pertama, fungsi mengecek apakah "current_142" lebih besar dari "n_142"; jika iya, fungsi bakal return dan berhenti. Selanjutnya, fungsi mengecek apakah "current_142" adalah faktor dari "n_142" dengan mengecek apakah sisa pembagian "n_142" dengan "current_142" sama dengan nol. Jika iya, fungsi bakal nge-print "current_142"

sebagai faktor. Setelah itu, fungsi memanggil dirinya sendiri dengan "current_142" yang ditambah satu, sehingga pemeriksaan faktor bakal terus berjalan sampai semua bilangan dari 1 sampai "n_142" diperiksa. Di fungsi "main", program minta input dari user untuk nilai "N", nge-print judul untuk hasil, dan memanggil fungsi "printFactors" untuk menampilkan faktor dari bilangan tersebut.

Unguided4

Source Code

```
package main

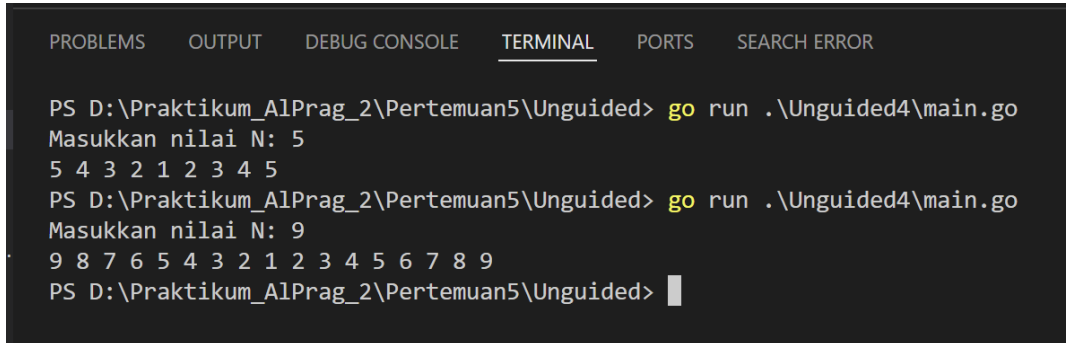
import (
    "fmt"
)

func printDescending(n_142 int) {
    if n_142 < 1 {
        return
    }
    fmt.Print(n_142, " ")
    printDescending(n_142 - 1)
}

func printAscending(current_142, n_142 int) {
    if current_142 > n_142 {
        return
    }
    fmt.Print(current_142, " ")
    printAscending(current_142 + 1, n_142)
}

func main() {
    var n_142 int
    fmt.Print("Masukkan nilai N: ")
    fmt.Scan(&n_142)
    printDescending(n_142)
    printAscending(2, n_142)
    fmt.Println()
}
```

Screenshot



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided4\main.go
Masukkan nilai N: 5
5 4 3 2 1 2 3 4 5
PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided4\main.go
Masukkan nilai N: 9
9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9
PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> 
```

Deskripsi

Kode ini mencetak dua urutan angka: satu secara menurun dan satu secara menaik. Fungsi "printDescending" menerima satu parameter "n_142" dan mengecek apakah "n_142" kurang dari 1; jika iya, fungsi bakal berhenti. Jika tidak, fungsi nge-print nilai "n_142" dan memanggil dirinya sendiri dengan "n_142" dikurangi 1, sehingga mencetak angka dari "n_142" sampai 1 secara menurun. Fungsi "printAscending" menerima dua parameter: "current_142" (angka yang sedang dicetak) dan "n_142" (batas atas). Fungsi ini mengecek apakah "current_142" lebih besar dari "n_142"; jika iya, fungsi bakal berhenti. Jika tidak, fungsi nge-print "current_142" dan memanggil dirinya sendiri dengan "current_142" ditambah 1, sehingga mencetak angka dari 2 sampai "n_142" secara menaik. Di fungsi "main", program minta input dari user untuk nilai "N", kemudian memanggil fungsi "printDescending" untuk mencetak angka secara menurun, diikuti dengan fungsi "printAscending" untuk mencetak angka secara menaik, lalu nge-print newline di akhir.

Unguided5

Source Code

```
package main

import (
    "fmt"
)
```



```

func printOdd(current_142, n_142 int) {
    if current_142 > n_142 {
        return
    }
    fmt.Print(current_142, " ")
    printOdd(current_142+2, n_142)
}

func main() {
    var n_142 int
    fmt.Print("Masukkan nilai N: ")
    fmt.Scan(&n_142)

    printOdd(1, n_142)
    fmt.Println()
}

```

Screenshot

The screenshot shows a Go IDE interface with a terminal window. The terminal displays the following commands and output:

```

PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided5\main.go
Masukkan nilai N: 5
1 3 5
PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided5\main.go
Masukkan nilai N: 20
1 3 5 7 9 11 13 15 17 19
PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided>

```

Deskripsi

Kode ini mencetak angka ganjil mulai dari 1 hingga nilai yang dimasukkan oleh user. Fungsi "printOdd" adalah fungsi rekursif yang menerima dua parameter: "current_142" (angka ganjil saat ini) dan "n_142" (batas atas). Fungsi ini mengecek apakah "current_142" lebih besar dari "n_142"; jika iya, fungsi bakal berhenti. Jika tidak, fungsi nge-print "current_142" dan kemudian manggil dirinya sendiri dengan "current_142" ditambah 2, sehingga mencetak angka ganjil secara berurutan. Di fungsi "main", program minta input dari user untuk nilai "N" dan kemudian manggil fungsi "printOdd" dengan 1 sebagai angka awal, lalu nge-print newline di akhir untuk format yang rapi.

Unguided6

Source Code

```
package main

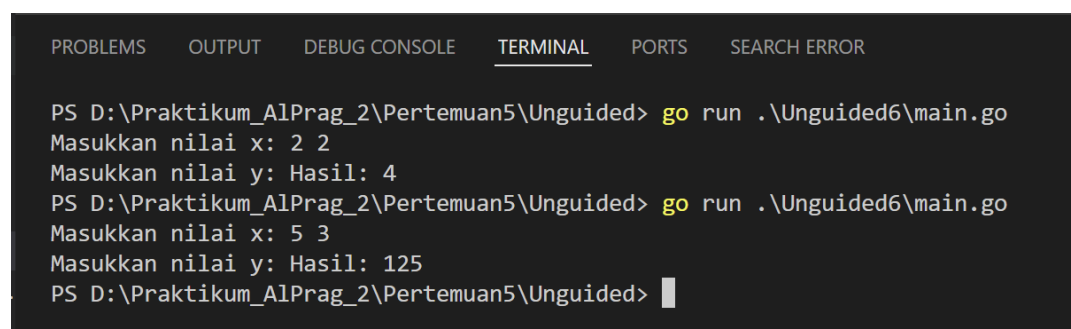
import (
    "fmt"
)

func power(x_142, y_142 int) int {
    if y_142 == 0 {
        return 1
    }
    return x_142 * power(x_142, y_142-1)
}

func main() {
    var x_142, y_142 int
    fmt.Print("Masukkan nilai x: ")
    fmt.Scan(&x_142)
    fmt.Print("Masukkan nilai y: ")
    fmt.Scan(&y_142)

    result_142 := power(x_142, y_142)
    fmt.Println("Hasil:", result_142)
}
```

Screenshot



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided6\main.go
Masukkan nilai x: 2 2
Masukkan nilai y: Hasil: 4
PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> go run .\Unguided6\main.go
Masukkan nilai x: 5 3
Masukkan nilai y: Hasil: 125
PS D:\Praktikum_AlPrag_2\Pertemuan5\Unguided> █
```

Deskripsi

Kode ini menghitung nilai pangkat dari bilangan x dengan eksponen y menggunakan fungsi rekursif. Fungsi "power" menerima dua parameter: "x_142" (bilangan yang akan dipangkatkan) dan "y_142" (eksponen). Di dalam fungsi, ada kondisi base case yang mengecek apakah "y_142" sama dengan 0; jika

iya, fungsi bakal return 1 (karena bilangan apa pun yang dipangkatkan 0 itu 1). Jika tidak, fungsi bakal return hasil dari "x_142" dikali dengan hasil pangkat dari "x_142" dengan eksponen "y_142" yang dikurangi 1. Di fungsi "main", program minta input dari user untuk nilai x dan y, kemudian manggil fungsi "power" untuk menghitung hasil pangkat dan nge-print hasilnya.

III. DAFTAR PUSTAKA

- 1) Asisten praktikum, Akmelia Zahara dan Kyla Azzahra Kinan “Modul V Rekursif” Learning Management System, 2024