

**LAPORAN PRAKTIKUM
ALGORITMA PROGRAM 2
MODUL 6
REKURSIF**



Oleh:

ADITHANA DHARMA PUTRA

2311102207

IF – 11- 02

**S1 TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

2024

I. DASAR TEORI

rekursif secara sederhana dapat diartikan sebagai cara menyelesaikan suatu masalah dengan cara menyelesaikan sub-masalah yang identik dari masalah utama. Sebagai contoh :

	Notasi Algoritma	Notasi dalam bahasa Go
1 2 3 4 5	procedure cetak(in x:integer) algoritma output(x) cetak(x+ 1) endprocedure	func cetak(x int){ fmt.Println(x) cetak(x+1) }

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 Control percabangan (if-then) untuk menghentikan proses rekursif ini. Kondisi ini disebut juga dengan base-case, artinya apabila kondisi base-case bernilai true maka proses rekursif akan berhenti. Sebagai contoh misalnya base case adalah ketika x bernilai 10 atau `x == 10`, maka tidak perlu dilakukan rekursif.

```

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

```

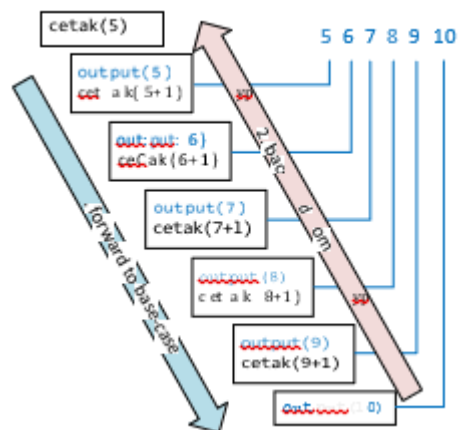
Apabila diperhatikan pada baris ke-3 di Program di atas, kita telah menambahkan base-case seperti penjelasan sebelumnya. Selanjutnya pada bagian aksi dari else di baris ke-6 dan ke-7 kita namakan recursive-case atau kasus pemanggilan dirinya sendiri tersebut terjadi. Kondisi dari **recursive-case** ini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



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 }
```

- 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 pedoman yang harus diketahui agar tidak 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 iteratif.

Komponen Rekursif

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

1. Guided 1

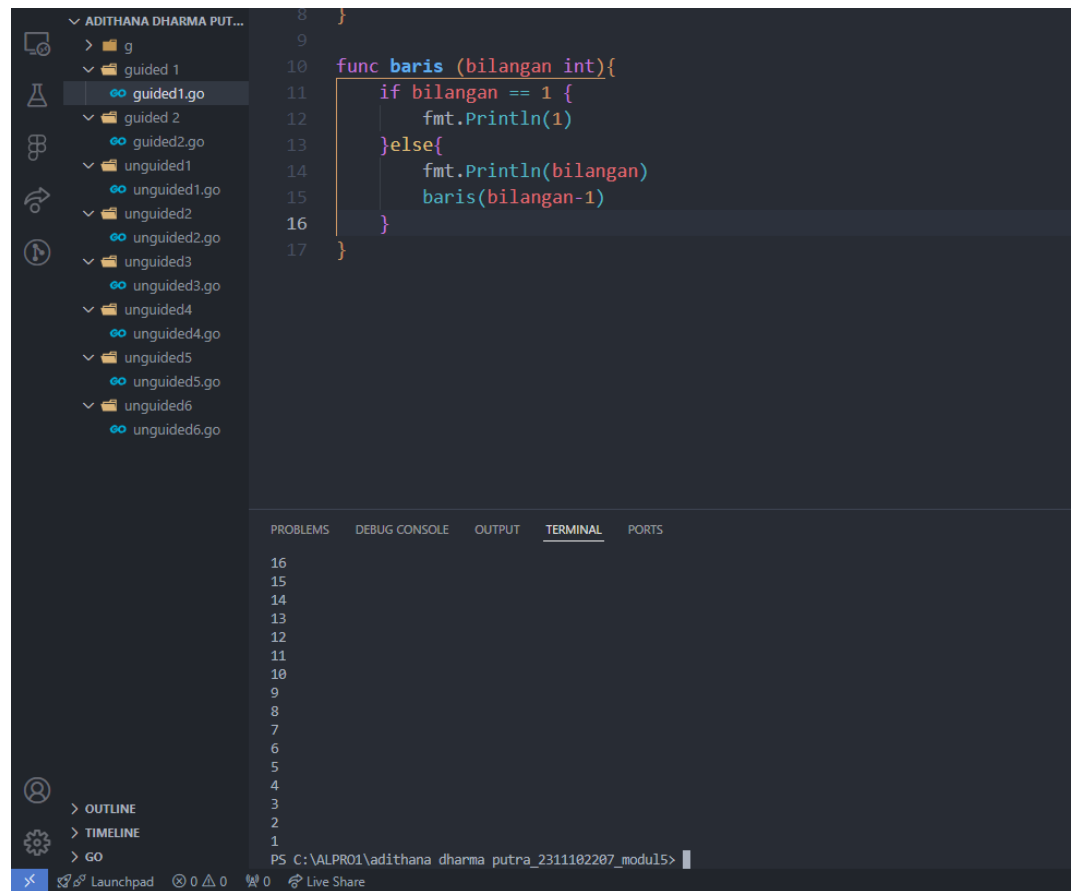
Source Code

```
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



The screenshot shows a Go IDE with a file explorer on the left, a code editor in the center, and a terminal at the bottom. The file explorer shows a project structure with folders 'guided' and 'unguided', each containing numbered Go files. The code editor displays a recursive function 'baris' that prints a sequence of numbers from a given integer down to 1. The terminal shows the output of the program, which is the sequence of numbers 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, printed on separate lines.

```
8 }
9
10 func baris (bilangan int){
11     if bilangan == 1 {
12         fmt.Println(1)
13     }else{
14         fmt.Println(bilangan)
15         baris(bilangan-1)
16     }
17 }
```

PROBLEMS DEBUG CONSOLE OUTPUT TERMINAL PORTS

16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1

PS C:\ALPR01\adithana dharma putra_2311102207_modul5>

Deskripsi:

Kode ini adalah program yang membaca sebuah bilangan bulat dari input pengguna dan mencetak bilangan tersebut secara berurutan dari bilangan itu sendiri hingga 1. Di dalam fungsi main, sebuah variabel `n` dideklarasikan untuk menyimpan bilangan bulat yang dibaca dari input pengguna menggunakan `fmt.Scan(&n)`. Setelah itu, fungsi `baris` dipanggil dengan argumen `n`. Fungsi `baris` sendiri adalah fungsi rekursif yang mencetak bilangan yang diterima sebagai argumen. Jika bilangan tersebut adalah 1, maka fungsi akan mencetak 1. Jika tidak, fungsi akan mencetak bilangan tersebut dan kemudian memanggil dirinya sendiri dengan argumen `bilangan - 1`. Dengan cara ini, program akan mencetak semua bilangan bulat positif dari bilangan yang dimasukkan pengguna hingga 1.

2. Guided 2

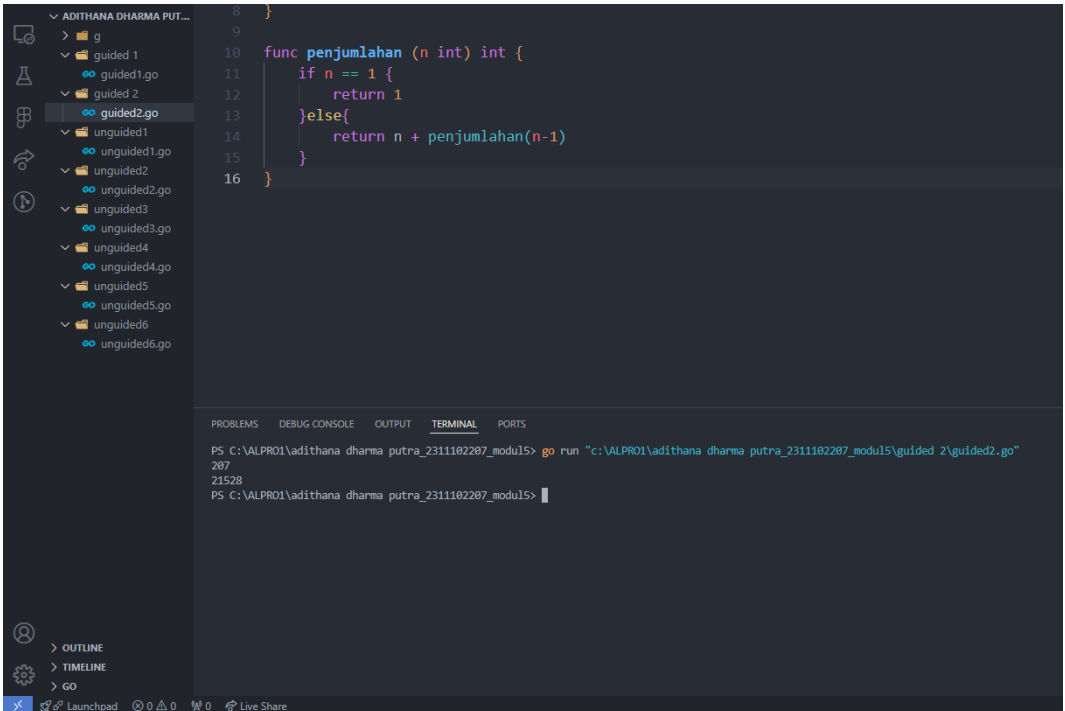
Source Code

```
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



The screenshot shows a Go IDE with a project named 'ADITHANA DHARMA PUTRA_2311102207'. The file explorer on the left shows a directory structure with 'guided1', 'guided2', and several 'unguided' files. The main editor displays a Go function named 'penjumlahan' which is a recursive function to calculate the sum of numbers from 1 to n. The function signature is 'func penjumlahan (n int) int'. The base case is 'if n == 1 { return 1 }'. The recursive case is 'return n + penjumlahan(n-1)'. The terminal at the bottom shows the command 'go run "c:\ALPRO1\adithana dharma putra_2311102207_modul5\guided 2\guided2.go"' and the output '207' and '21528'.

```
8 }
9
10 func penjumlahan (n int) int {
11     if n == 1 {
12         return 1
13     }else{
14         return n + penjumlahan(n-1)
15     }
16 }
```

PROBLEMS DEBUG CONSOLE OUTPUT **TERMINAL** PORTS

```
PS C:\ALPRO1\adithana dharma putra_2311102207_modul5> go run "c:\ALPRO1\adithana dharma putra_2311102207_modul5\guided 2\guided2.go"
207
21528
PS C:\ALPRO1\adithana dharma putra_2311102207_modul5> |
```

Deskripsi:

Kode ini adalah program yang membaca sebuah bilangan bulat dari input pengguna dan menghitung jumlah semua bilangan dari 1 hingga bilangan tersebut menggunakan rekursi. Di dalam fungsi main, sebuah variabel n dideklarasikan untuk menyimpan bilangan bulat yang dibaca dari input pengguna menggunakan `fmt.Scan(&n)`. Setelah itu, fungsi `penjumlahan` dipanggil dengan argumen `n`, dan hasilnya dicetak menggunakan `fmt.Println`.

Fungsi `penjumlahan` adalah fungsi rekursif yang menghitung jumlah semua bilangan dari 1 hingga `n`. Jika `n` sama dengan 1, fungsi mengembalikan nilai 1. Jika tidak, fungsi mengembalikan nilai `n` ditambah hasil pemanggilan fungsi `penjumlahan` dengan argumen `n - 1`. Dengan cara ini, program akan menghitung dan mencetak jumlah semua bilangan dari 1 hingga bilangan yang dimasukkan pengguna.

II. UNGUIDED

1. Unguided 1

Source Code

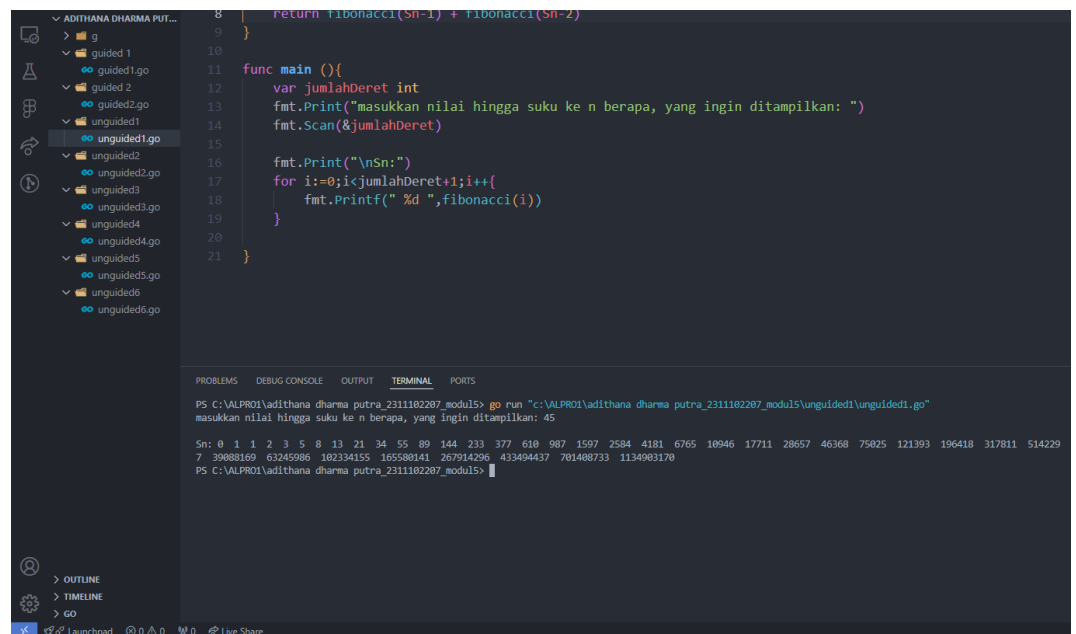
```
package main
import "fmt"

func fibonacci (Sn int) int {
    if Sn<=1{
        return Sn
    }
    return fibonacci(Sn-1) + fibonacci(Sn-2)
}

func main (){
    var jumlahDeret int
    fmt.Print("masukkan nilai hingga suku ke
n berapa, yang ingin ditampilkan: ")
    fmt.Scan(&jumlahDeret)

    fmt.Print("\nSn:")
    for i:=0;i<jumlahDeret+1;i++){
        fmt.Printf(" %d ",fibonacci(i))
    }
}
```

Screenshot



The screenshot shows a Go IDE with a project named 'ADITHANA DHARMA PUTRA'. The file explorer on the left shows a directory structure with files named 'guided1.go' through 'guided6.go' and 'unguided1.go' through 'unguided6.go'. The editor displays the code for 'unguided1.go'.

```
8      return fibonacci(Sn-1) + fibonacci(Sn-2)
9    }
10
11    func main () {
12        var jumlahDeret int
13        fmt.Print("masukkan nilai hingga suku ke n berapa, yang ingin ditampilkan: ")
14        fmt.Scan(&jumlahDeret)
15
16        fmt.Print("\nSn:")
17        for i:=0;i<jumlahDeret+1;i++){
18            fmt.Printf(" %d ",fibonacci(i))
19        }
20    }
21 }
```

The terminal output shows the command to run the program and the resulting Fibonacci sequence for n=45:

```
PS C:\VALPRO1\adithana dharm putra_2311182287_modul5> go run "c:\VALPRO1\adithana dharm putra_2311182287_modul5\unguided1\unguided1.go"
masukkan nilai hingga suku ke n berapa, yang ingin ditampilkan: 45
Sn: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657 46368 75025 121393 196418 317811 514229
7 39888169 63245986 102334155 165580141 267914296 433494437 701408733 1134963170
PS C:\VALPRO1\adithana dharm putra_2311182287_modul5>
```

Deskripsi

Kode ini adalah program yang menghitung dan menampilkan deret Fibonacci hingga suku ke-n yang dimasukkan oleh pengguna. Fungsi fibonacci adalah fungsi rekursif yang menghitung nilai suku ke-n dalam deret Fibonacci. Jika Sn kurang dari atau sama dengan 1, fungsi mengembalikan nilai Sn. Jika tidak, fungsi mengembalikan hasil penjumlahan dari dua pemanggilan rekursif fungsi fibonacci dengan argumen Sn-1 dan Sn-2.

Di dalam fungsi main, sebuah variabel jumlahDeret dideklarasikan untuk menyimpan bilangan bulat yang dibaca dari input pengguna menggunakan `fmt.Scan(&jumlahDeret)`. Program kemudian mencetak pesan untuk meminta pengguna memasukkan nilai hingga suku ke-n yang ingin ditampilkan. Setelah menerima input, program mencetak deret Fibonacci dari suku ke-0 hingga suku ke-n dengan menggunakan loop for dan memanggil fungsi fibonacci untuk setiap nilai i dari 0 hingga jumlahDeret.

2. Unguided 2

Source Code

```
package main
import "fmt"

func polaBintang(n int) {
    if n > 0 {
        polaBintang(n - 1)
        for i := 0; i < n; i++ {
            fmt.Print("*")
        }
        fmt.Println()
    }
}

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

Screenshot

```

8         fmt.Println("")
9     }
10    }
11    fmt.Println()
12    }
13
14    func main(){
15        var n int
16        fmt.Print("Masukkan jumlah baris: ")
17        fmt.Scan(&n)
18        polaBintang(n)
19    }

```

PROBLEMS DEBUG CONSOLE OUTPUT **TERMINAL** PORTS

```

PS C:\ALPRO01\adithana dharma putra_2311102207_moduls> go run "c:\ALPRO01\adithana dharma putra_2311102207_moduls\unguided2\unguided2.go"
Masukkan jumlah baris: 8
*
**
***
****
*****
*****
*****
*****
PS C:\ALPRO01\adithana dharma putra_2311102207_moduls>

```

Deskripsi

Kode ini adalah program yang mencetak pola bintang secara bertingkat berdasarkan jumlah baris yang dimasukkan oleh pengguna. Fungsi polaBintang adalah fungsi rekursif yang mencetak baris bintang. Jika n lebih besar dari 0, fungsi akan memanggil dirinya sendiri dengan argumen $n - 1$, kemudian mencetak n bintang pada baris baru.

Di dalam fungsi main, sebuah variabel `n` dideklarasikan untuk menyimpan bilangan bulat yang dibaca dari input pengguna menggunakan `fmt.Scan(&n)`. Program kemudian mencetak pesan untuk meminta pengguna memasukkan jumlah baris yang ingin ditampilkan. Setelah menerima input, program memanggil fungsi `polaBintang` dengan argumen `n`, yang akan mencetak pola bintang dari 1 hingga `n` baris.

3. Unguided 3

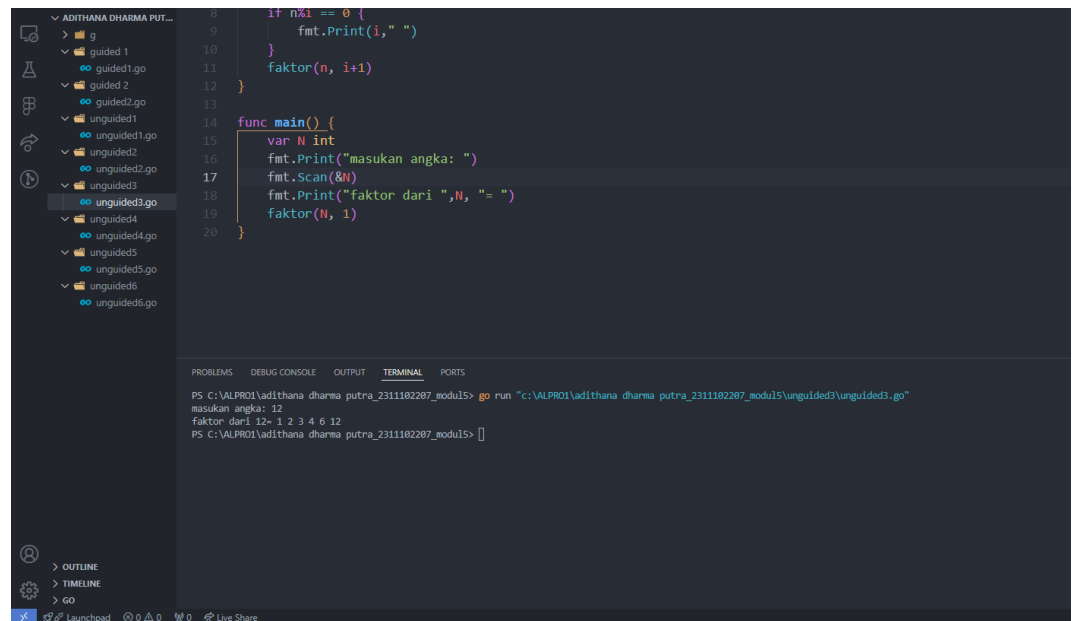
Source Code

```
package main
import "fmt"

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

func main() {
    var N int
    fmt.Print("masukan angka: ")
    fmt.Scan(&N)
    fmt.Print("faktor dari ", N, "= ")
    faktor(N, 1)
}
```

Screenshot



```
8  if n%1 == 0 {
9      fmt.Print(i, " ")
10 }
11 faktor(n, i+1)
12 }
13
14 func main() {
15     var N int
16     fmt.Print("masukan angka: ")
17     fmt.Scan(&N)
18     fmt.Print("faktor dari ", N, " = ")
19     faktor(N, 1)
20 }
```

PROBLEMS DEBUG CONSOLE OUTPUT **TERMINAL** PORTS

```
PS C:\VALPRO1\adithana dharm putra_2311182287_modul5> go run "c:\VALPRO1\adithana dharm putra_2311182287_modul5\unguided3\unguided3.go"
masukan angka: 12
faktor dari 12= 1 2 3 4 6 12
PS C:\VALPRO1\adithana dharm putra_2311182287_modul5> []
```

Deskripsi

Kode ini adalah program yang mencari dan mencetak semua faktor dari sebuah bilangan yang dimasukkan oleh pengguna. Fungsi faktor adalah fungsi rekursif yang mencari faktor dari bilangan n. Fungsi ini menerima dua argumen: n (bilangan yang akan dicari faktornya) dan i (bilangan yang digunakan untuk memeriksa apakah i adalah faktor dari n). Di dalam fungsi faktor, jika i lebih besar dari n, fungsi akan berhenti. Jika n habis dibagi i, maka i dicetak sebagai faktor. Fungsi kemudian memanggil dirinya sendiri dengan argumen i yang ditambah 1. Di dalam fungsi main, sebuah variabel N dideklarasikan untuk menyimpan bilangan bulat yang dibaca dari input pengguna menggunakan `fmt.Scan(&N)`. Program kemudian mencetak pesan untuk meminta pengguna memasukkan sebuah angka. Setelah menerima input, program mencetak pesan yang menunjukkan bahwa faktor dari N akan ditampilkan, dan memanggil fungsi faktor dengan argumen N dan 1 untuk memulai pencarian faktor.

4. Unguided 4

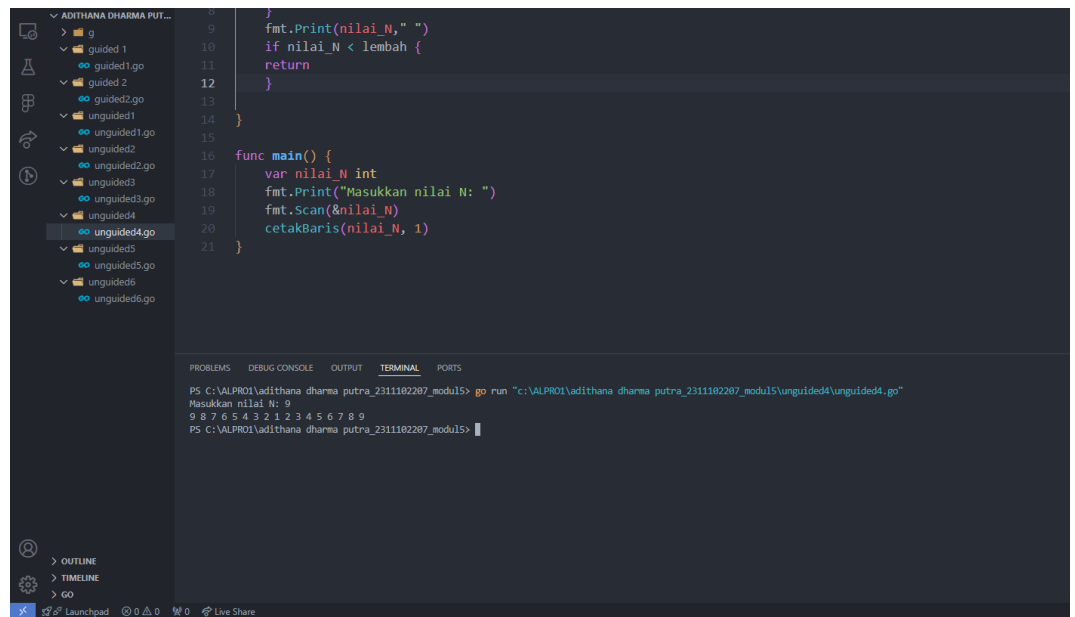
Source Code

```
package main
import "fmt"

func cetakBaris(nilai_N,lembah int) {
    if nilai_N > lembah {
        fmt.Print(nilai_N," ")
        cetakBaris(nilai_N-1,lembah)
    }
    fmt.Print(nilai_N," ")
    if nilai_N < lembah {
        return
    }
}

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


Screenshot



```
8      }
9      fmt.Print(nilai_N, " ")
10     if nilai_N < lembah {
11     return
12     }
13
14 }
15
16 func main() {
17     var nilai_N int
18     fmt.Print("Masukkan nilai N: ")
19     fmt.Scan(&nilai_N)
20     cetakBaris(nilai_N, 1)
21 }
```

PROBLEMS DEBUG CONSOLE OUTPUT TERMINAL PORTS

```
PS C:\VALPRO1\adithana_dharma_putra_2311182287_modul5> go run "c:\VALPRO1\adithana_dharma_putra_2311182287_modul5\unguided4\unguided4.go"
Masukkan nilai N: 9
9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9
PS C:\VALPRO1\adithana_dharma_putra_2311182287_modul5>
```

Deskripsi

Kode ini adalah program yang mencetak deret angka dari nilai_N hingga 1, kemudian kembali mencetak angka dari 1 hingga nilai_N. Fungsi cetakBaris adalah fungsi rekursif yang mencetak deret angka berdasarkan dua argumen: nilai_N (nilai awal) dan lembah (nilai akhir).

Di dalam fungsi cetakBaris, jika nilai_N lebih besar dari lembah, fungsi mencetak nilai_N dan memanggil dirinya sendiri dengan argumen nilai_N - 1 dan lembah. Setelah pemanggilan rekursif, fungsi mencetak nilai_N lagi. Jika nilai_N kurang dari lembah, fungsi akan berhenti.

Di dalam fungsi main, sebuah variabel nilai_N dideklarasikan untuk menyimpan bilangan bulat yang dibaca dari input pengguna menggunakan `fmt.Scan(&nilai_N)`. Program kemudian mencetak pesan untuk meminta pengguna memasukkan nilai N. Setelah menerima input, program memanggil fungsi cetakBaris dengan argumen nilai_N dan 1 untuk memulai pencetakan deret angka.

5. Unguided 5

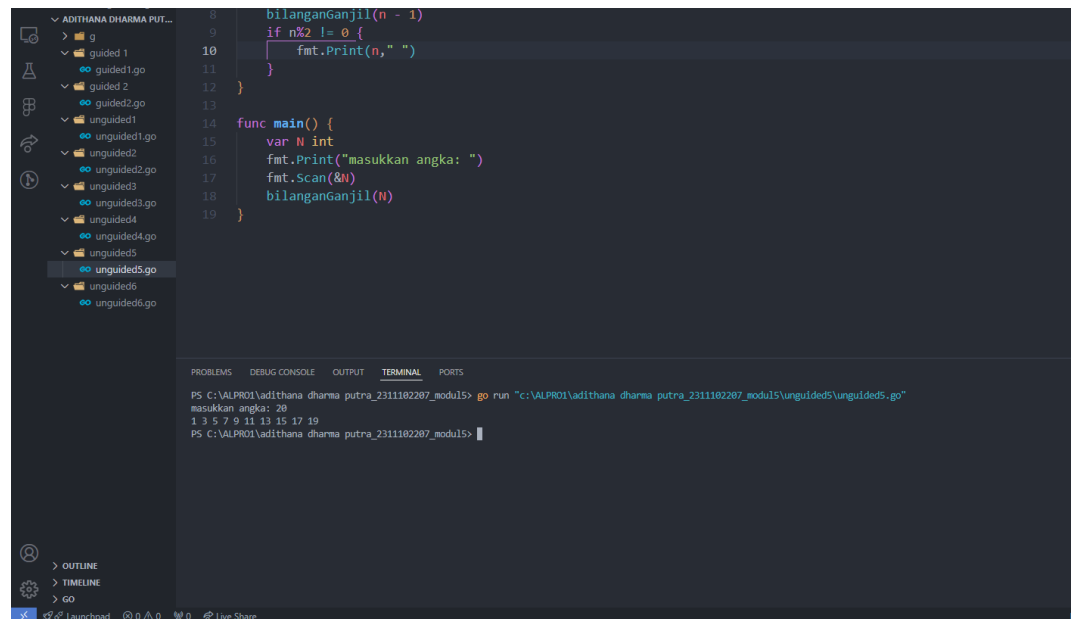
Source Code

```
package main
import "fmt"

func bilanganGanjil(n int) {
    if n <= 0 {
        return
    }
    bilanganGanjil(n - 1)
    if n%2 != 0 {
        fmt.Print(n, " ")
    }
}

func main() {
    var N int
    fmt.Print("masukkan angka: ")
    fmt.Scan(&N)
    bilanganGanjil(N)
}
```

Screenshot



```
8  bilanganGanjil(n - 1)
9
10 if n%2 != 0 {
11     fmt.Println(n, " ")
12 }
13
14 func main() {
15     var N int
16     fmt.Print("masukkan angka: ")
17     fmt.Scan(&N)
18     bilanganGanjil(N)
19 }
```

PROBLEMS DEBUG CONSOLE OUTPUT **TERMINAL** PORTS

```
PS C:\ALPRO1\adithana_dharma_putra_2311182287_modul5> go run "c:\ALPRO1\adithana_dharma_putra_2311182287_modul5\unguided5\unguided5.go"
masukkan angka: 20
1 3 5 7 9 11 13 15 17 19
PS C:\ALPRO1\adithana_dharma_putra_2311182287_modul5>
```

Deskripsi

Kode ini adalah program yang mencetak semua bilangan ganjil dari 1 hingga N, di mana N adalah bilangan yang dimasukkan oleh pengguna. Fungsi `bilanganGanjil` adalah fungsi rekursif yang mencetak bilangan ganjil dari 1 hingga n. Di dalam fungsi `bilanganGanjil`, jika n kurang dari atau sama dengan 0, fungsi akan berhenti. Jika tidak, fungsi akan memanggil dirinya sendiri dengan argumen `n - 1`. Setelah pemanggilan rekursif, fungsi memeriksa apakah n adalah bilangan ganjil dengan menggunakan kondisi `n%2 != 0`. Jika benar, n dicetak.

Di dalam fungsi `main`, sebuah variabel `N` dideklarasikan untuk menyimpan bilangan bulat yang dibaca dari input pengguna menggunakan `fmt.Scan(&N)`. Program kemudian mencetak pesan untuk meminta pengguna memasukkan sebuah angka. Setelah menerima input, program memanggil fungsi `bilanganGanjil` dengan argumen `N` untuk memulai pencetakan bilangan ganjil dari 1 hingga N.

6. Unguided 6

Source Code

```
package main
import "fmt"

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

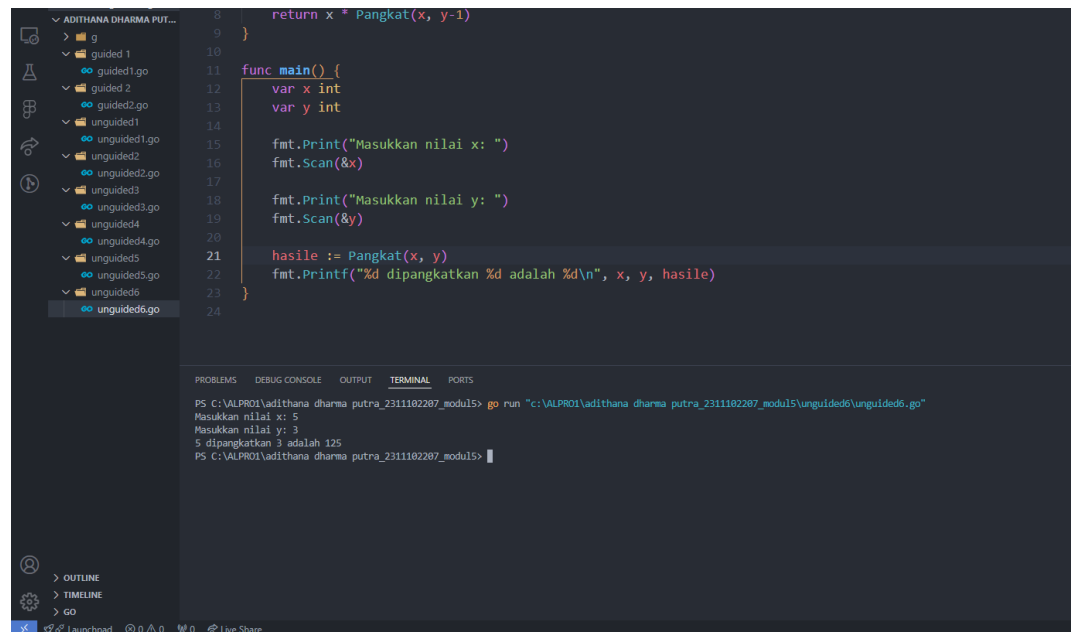
func main() {
    var x int
    var y int

    fmt.Print("Masukkan nilai x: ")
    fmt.Scan(&x)

    fmt.Print("Masukkan nilai y: ")
    fmt.Scan(&y)

    hasile := Pangkat(x, y)
    fmt.Printf("%d dipangkatkan %d adalah %d\n", x, y, hasile)
}
```

Screenshot



```
8      return x * Pangkat(x, y-1)
9    }
10
11    func main() {
12        var x int
13        var y int
14
15        fmt.Print("Masukkan nilai x: ")
16        fmt.Scan(&x)
17
18        fmt.Print("Masukkan nilai y: ")
19        fmt.Scan(&y)
20
21        hasile := Pangkat(x, y)
22        fmt.Printf("%d dipangkatkan %d adalah %d\n", x, y, hasile)
23    }
24
```

PROBLEMS DEBUG CONSOLE OUTPUT TERMINAL PORTS

```
PS C:\VALPRO1\adithana dhama putra_2311182287_modul5> go run "c:\VALPRO1\adithana dhama putra_2311182287_modul5\unguided6\unguided6.go"
Masukkan nilai x: 5
Masukkan nilai y: 3
5 dipangkatkan 3 adalah 125
PS C:\VALPRO1\adithana dhama putra_2311182287_modul5>
```

Deskripsi

Kode ini adalah program dalam bahasa Go yang menghitung hasil perpangkatan dari dua bilangan yang dimasukkan oleh pengguna. Fungsi Pangkat adalah fungsi rekursif yang menghitung hasil perpangkatan dari x dipangkatkan y. Jika y sama dengan 0, fungsi mengembalikan nilai 1 (karena setiap bilangan yang dipangkatkan 0 adalah 1). Jika tidak, fungsi mengembalikan hasil perkalian x dengan hasil pemanggilan rekursif fungsi Pangkat dengan argumen x dan y-1.

Di dalam fungsi main, dua variabel x dan y dideklarasikan untuk menyimpan bilangan bulat yang dibaca dari input pengguna menggunakan `fmt.Scan(&x)` dan `fmt.Scan(&y)`. Program kemudian mencetak pesan untuk meminta pengguna memasukkan nilai x dan y. Setelah menerima input, program memanggil fungsi Pangkat dengan argumen x dan y, dan hasilnya disimpan dalam variabel `hasile`. Program kemudian mencetak hasil perpangkatan tersebut menggunakan `fmt.Printf`