

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2
MODUL V
REKURSIF



Disusun Oleh :

Rasyid Nafsyarie / 2311102011

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 dari masalah aslinya, hingga mencapai kondisi dasar (base case) yang menghentikan pemanggilan rekursif. Dengan menggunakan rekursi, masalah yang besar dan kompleks dapat dipecah menjadi sub-masalah yang lebih sederhana dan lebih mudah diselesaikan.

Dalam rekursi, setiap kali fungsi memanggil dirinya sendiri, keadaan saat ini disimpan di "stack" memori hingga base case tercapai. Setelah itu, fungsi akan kembali menyelesaikan permasalahan secara bertahap dari tumpukan stack (proses ini disebut "unwinding").

Rekursi terdiri dari dua komponen utama:

Base Case (Kondisi Dasar): Bagian dari fungsi yang menentukan kapan rekursi berhenti. Base case sangat penting karena jika tidak ada, rekursi akan berjalan tanpa henti, menyebabkan program mengalami "stack overflow".

Recursive Case (Kondisi Rekursif): Bagian dari fungsi yang memanggil fungsi itu sendiri dengan nilai yang lebih kecil atau lebih sederhana, secara bertahap menuju base case.

Sebagai contoh, berikut adalah implementasi rekursif untuk menghitung faktorial suatu bilangan dalam Golang:

Rumus Faktorial

Faktorial dari bilangan n (dilambangkan dengan $n!$) adalah hasil perkalian semua bilangan bulat positif hingga n . Secara rekursif, faktorial dapat didefinisikan sebagai:

$$n! = n \times (n-1)! \quad n! = n \times (n-1)! \text{ jika } n > 0$$

$$0! = 1 \text{ (base case)}$$

Kelebihan:

Mempermudah solusi untuk masalah yang dapat dipecah menjadi sub-masalah berulang, seperti perhitungan faktorial, Fibonacci, dan traversal pohon.

Membuat kode lebih bersih dan mudah dipahami untuk masalah tertentu.

Kekurangan:

Efisiensi: Rekursi sering memerlukan lebih banyak memori karena setiap panggilan fungsi disimpan dalam stack memori. Jika rekursi terlalu dalam, ini bisa menyebabkan **stack overflow**.

Overhead: Setiap pemanggilan fungsi memiliki overhead, sehingga rekursi sering lebih lambat dibandingkan solusi iteratif.

II. GUIDED

1. Guided 1

Sourcecode

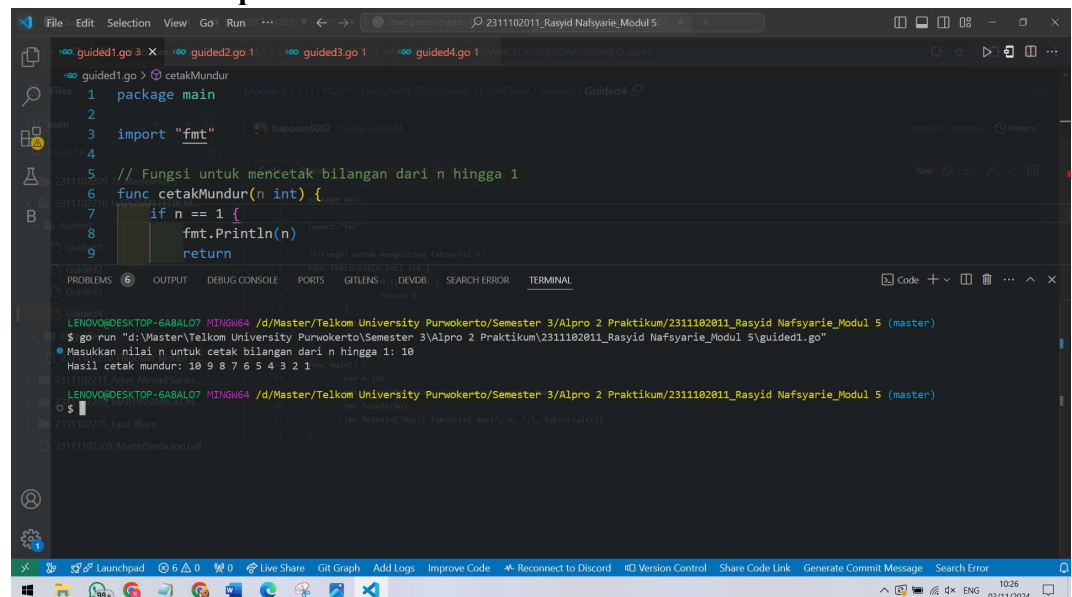
```
package main

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)
}
```

Screenshoot Output



Deskripsi Program

Fungsi cetakMundur Fungsi ini menerima parameter n yang merupakan bilangan bulat. Jika n sama dengan 1, fungsi mencetak 1 dan mengembalikan kontrol. Jika tidak, fungsi mencetak nilai n diikuti dengan spasi, kemudian memanggil dirinya sendiri dengan n-1.

2. Guided 2

Soal Studi Case

XXXXXXXXXXXXXXXXXX

Sourcecode

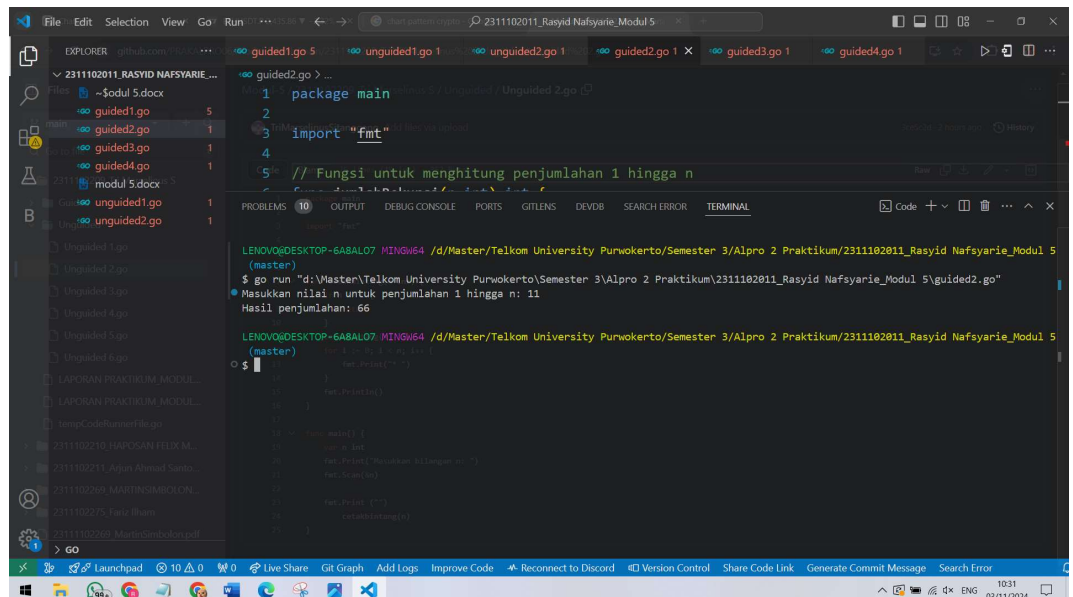
```
package main

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))
}
```

Screenshoot Output



Deskripsi Program

Fungsi `jumlahRekursi` Fungsi ini menerima satu parameter `n` yang merupakan angka hingga mana penjumlahan akan dilakukan. Jika `n` sama dengan 1, fungsi akan mengembalikan 1 (kasus dasar). Jika tidak, fungsi akan mengembalikan `n` ditambah hasil dari pemanggilan fungsi `jumlahRekursi` dengan parameter `n-1`. Ini adalah langkah rekursif yang akan terus berlanjut hingga mencapai kasus dasar.

3. Guided 3

Soal Studi Case

XXXXXXXXXXXXXXXXXXXX

Sourcecode

```
package main

import "fmt"

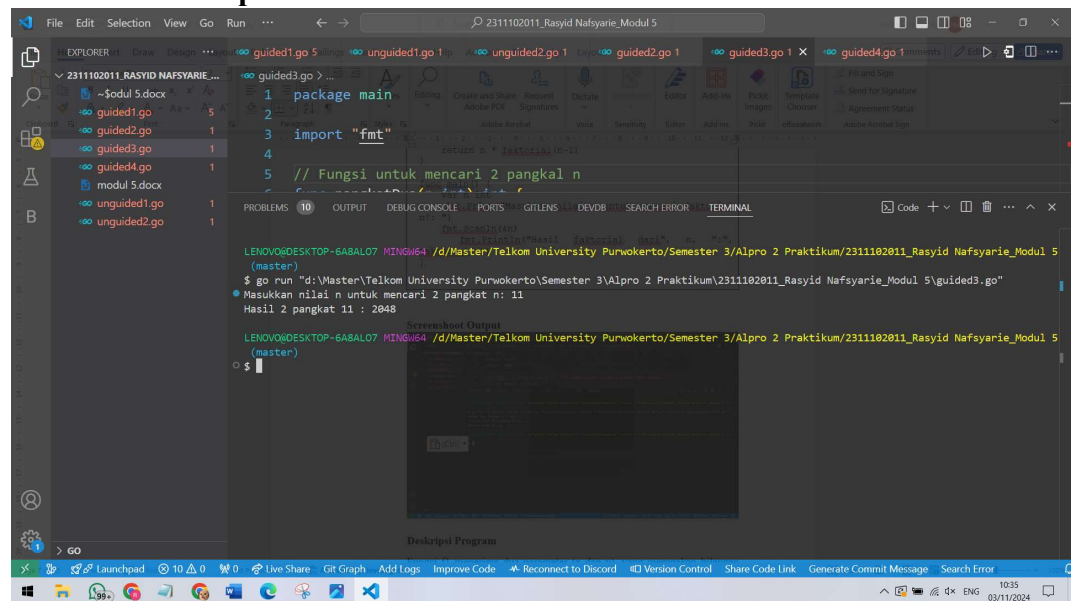
// Fungsi untuk mencari 2 pangkat n
func pangkatDua(n int) int {
    if n == 0 {
        return 1
    }
    return 2 * pangkatDua(n-1)
}
```

```

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))
}

```

Screenshoot Output



Deskripsi Program

Fungsi `pangkatDua` Basis Kasus: Jika `n` sama dengan 0, fungsi mengembalikan 1. Ini penting karena 2 pangkat 0 adalah 1. Rekursi: Jika `n` lebih besar dari 0, fungsi akan mengalikan 2 dengan hasil dari `pangkatDua(n-1)`. Ini berarti fungsi akan terus memanggil dirinya sendiri dengan nilai `n` yang berkurang hingga mencapai 0.

4. Guided 4

Soal Studi Case

XXXXXXXXXXXXXXXXXXXX

Sourcecode

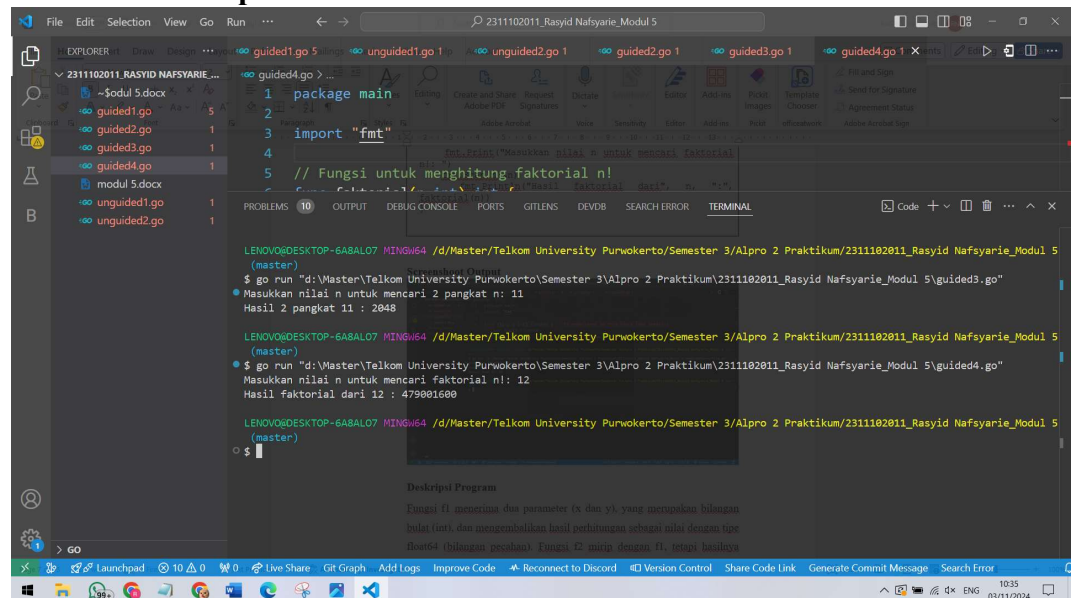
```
package main

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))
}
```

Screenshoot Output



Deskripsi Program

Fungsi Rekursif: Kode ini menggunakan fungsi rekursif untuk menghitung faktorial. Fungsi rekursif adalah fungsi yang memanggil dirinya sendiri untuk menyelesaikan masalah yang lebih kecil.

III. UNGUIDED

Unguided 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 $S_2 = S_{n-1} + S_{n-2}$. Berikut ini adalah contoh nilai deret fibonacci hingga suku ke-10. Buatlah program yang mengimplementasikan fungsi rekursif pada deret fibonacci tersebut.

Sourcecode

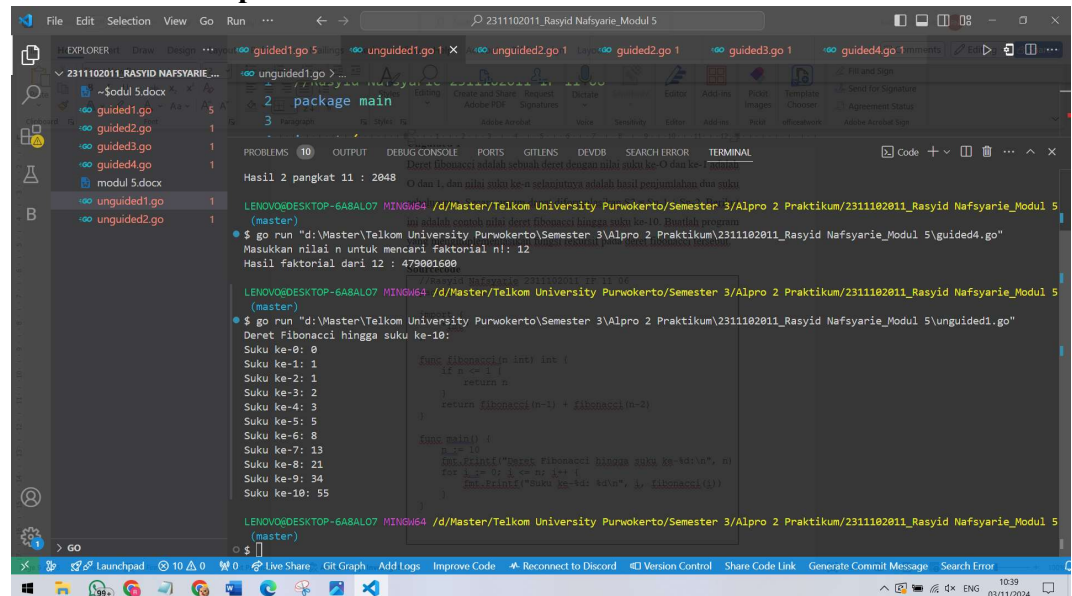
```
//Rasyid Nafsyarie 2311102011 IF 11 06
package main

import (
    "fmt"
)

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

func main() {
    n := 10
    fmt.Printf("Deret Fibonacci hingga suku ke-%d:\n", n)
    for i := 0; i <= n; i++ {
        fmt.Printf("Suku ke-%d: %d\n", i, fibonacci(i))
    }
}
```

Screenshoot Output



The screenshot shows a VS Code editor with a Go project. The Explorer view on the left shows files like `guided1.go`, `guided2.go`, `guided3.go`, `guided4.go`, and `modul 5.docx`. The main editor shows the `package main` file. The terminal window at the bottom displays the following output:

```
Hasil 2 pangkat 11 : 2048
O dan 1, dan nilai suku ke-n selanjutnya adalah hasil penjumlahan dua suku
itu adalah suku ke-10 dari Fibonacci hingga suku ke-10. Berikut perintah
$ go run "d:\Master\Telkom University Purwokerto\Semester 3\Alpro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 5\guided4.go"
Masukkan nilai n untuk mencari faktorial ni: 12
Hasil faktorial dari 12 : 479001600

$ go run "d:\Master\Telkom University Purwokerto\Semester 3\Alpro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 5\unguided1.go"
Deret Fibonacci hingga suku ke-10:
Suku ke-0: 0
Suku ke-1: 1
Suku ke-2: 1
Suku ke-3: 2
Suku ke-4: 3
Suku ke-5: 5
Suku ke-6: 8
Suku ke-7: 13
Suku ke-8: 21
Suku ke-9: 34
Suku ke-10: 55
```

Deskripsi Program

Fungsi Fibonacci Fungsi ini menerima parameter n yang merupakan indeks suku Fibonacci yang ingin dihitung. Jika n kurang dari atau sama dengan 1, fungsi mengembalikan n itu sendiri (0 atau 1). Jika tidak, fungsi memanggil dirinya sendiri untuk menghitung dua suku sebelumnya dan menjumlahkannya.

Unguided 2

Buatlah sebuah program yang digunakan untuk menampilkan pola bintang berikut ini dengan menggunakan fungsi rekursif. N adalah masukan dari user.

Sourcecode

```
//Rasyid Nafsyarie 2311102011 IF 11 06
package main
import "fmt"

func cetakbintang(n int) {
    if n <= 0 {
        return
    }
    cetakbintang(n - 1)
    for i := 0; i < n; i++ {
```

```

        fmt.Print("* ")
    }
    fmt.Println()
}

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

    fmt.Print ("")
    cetakbintang(n)
}

```

Screenshoot Output

The screenshot shows a Go IDE with the following code in `unguided2.go`:

```

5 func cetakbintang(n int) {
13     fmt.Println()
14 }
15
16 func main() {
17     var n int

```

The output window shows the execution results:

```

Suku ke-8: 21
Suku ke-9: 34
Suku ke-10: 55

```

The terminal window shows the command to run the program and the resulting output:

```

$ go run "d:\Master\Telkom University Purwokerto\Semester 3\Alpro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 5\unguided2.go"
Masukkan bilangan n: 11
*
**
***
****
*****
*****
*****
*****
*****
*****
*****
*****
*****

```

Deskripsi Program

Fungsi cetakbintang Fungsi ini menerima satu parameter `n` yang merupakan bilangan bulat. Jika `n` kurang dari atau sama dengan 0, fungsi akan berhenti (return). Fungsi memanggil dirinya sendiri dengan `n - 1`, sehingga menciptakan efek rekursif. Setelah panggilan rekursif, loop for digunakan untuk mencetak bintang sebanyak `n` kali, diikuti dengan `fmt.Println()` untuk pindah ke baris berikutnya.

Unguided 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).

Sourcecode

```
//Rasyid Nafsyarie 2311102011 IF 11 06

package main
import (
    "fmt"
)

func findFactors(N, divisor int) {

    if divisor > N {
        return
    }

    if N%divisor == 0 {
        fmt.Printf("%d ", divisor)
    }

    findFactors(N, divisor+1)
}

func main() {
    var N int

    fmt.Print("Masukkan bilangan bulat positif N: ")
    fmt.Scan(&N)

    fmt.Printf("Faktor-faktor dari %d adalah: ", N)
    findFactors(N, 1)
    fmt.Println()
}
```

The screenshot shows a Windows desktop with a VS Code editor open. The terminal window is active, showing the following commands and output:

```

LENOVODESKTOP-GABAL07 MINGW64 /d/Master/Telkom University Purwokerto/Semester 3/Alpro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 5
(master)
$ go run "d:\Master\Telkom University Purwokerto\Semester 3\Alpro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 5\unguided2.go"
Masukkan bilangan n: 12
Faktor-faktor dari 12 adalah: 1 2 3 4 6 12
  
```

Fungsi `findFactors` Fungsi ini menerima dua parameter: `N` (bilangan bulat positif) dan `divisor` (bilangan yang digunakan untuk mencari faktor). Jika `divisor` lebih besar dari `N`, fungsi akan berhenti (base case). Jika `N` dapat dibagi oleh `divisor` tanpa sisa ($N \% \text{divisor} == 0$), maka `divisor` dicetak sebagai faktor. Fungsi kemudian memanggil dirinya sendiri dengan `divisor` yang ditingkatkan satu untuk mencari faktor berikutnya.

Buatlah program yang mengimplementasikan rekursif untuk menampilkan barisan bilangan tertentu

Sourcecode

```
// Rasyid Nafsyarie 2311102011 IF 11 06
package main

import (
    "fmt"
)
```

```

func printSeries(N, current int) {
    if current == 1 {
        fmt.Printf("%d ", current)
        return
    }

    fmt.Printf("%d ", current)
    printSeries(N, current-1)
    fmt.Printf("%d ", current)
}

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

    fmt.Printf("Barisan bilangan dari %d hingga 1 dan
kembali ke %d:\n", N, N)
    printSeries(N, N)
    fmt.Println()
}

```

Screenshoot Output

The screenshot shows a VS Code editor with a Go project. The Explorer view on the left shows a file named `2311102011_RASYID NAFSARIYE_Modul 5`. The Editor view shows the `printSeries` function and the `main` function. The Output view at the bottom shows the execution of the program.

```

LENOVO\DESKTOP-GABAL07 MINGW64 /d/Master/Telkom University Purwokerto/Semester 3/AIpro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 5
(master)
$ go run "d:\Master\Telkom University Purwokerto\Semester 3\AIpro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 5\unguided3.go"
Masukkan bilangan bulat positif N: 12
Faktor-faktor dari 12 adalah: 1 2 3 4 6 12

LENOVO\DESKTOP-GABAL07 MINGW64 /d/Master/Telkom University Purwokerto/Semester 3/AIpro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 5
(master)
$ go run "d:\Master\Telkom University Purwokerto\Semester 3\AIpro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 5\unguided4.go"
Masukkan bilangan bulat positif N: 12
Barisan bilangan dari 12 hingga 1 dan kembali ke 12:
12 11 10 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9 10 11 12

```

Deskripsi Program

Fungsi printSeries Parameter N adalah bilangan bulat positif yang dimasukkan oleh pengguna. Parameter current adalah bilangan yang sedang diproses. Jika current sama dengan 1, fungsi mencetak 1 dan mengembalikan kontrol. Jika tidak, fungsi mencetak current, memanggil dirinya sendiri dengan current-1, dan kemudian mencetak current lagi setelah pemanggilan rekursif.

Unguided 5

Buatlah program yang menimplementasikan 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

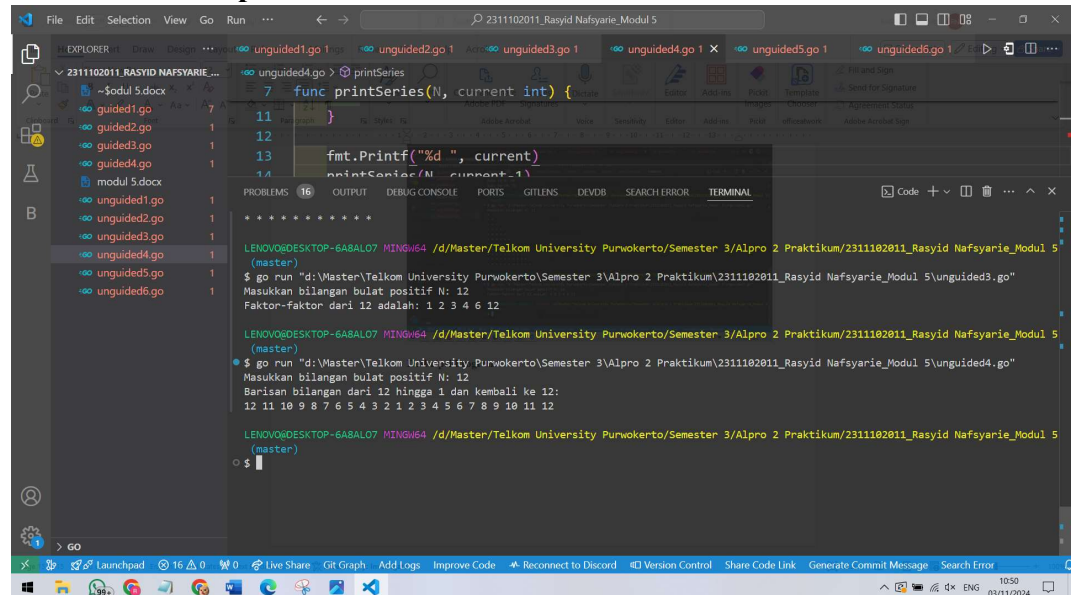
```
//Rasyid Nafsyarie 2311102011 IF 11 06
package main
import "fmt"

func ganjil(n int) {
    if n < 1 {
        return
    }

    if n%2 != 0 {
        ganjil(n - 2)
        fmt.Print(n, " ")
    } else {
        ganjil(n - 1)
    }
}

func main() {
    var n int
    fmt.Print("Masukkan n: ")
    fmt.Scanln(&n)
    ganjil(n)
}
```


Screenshoot Output



The screenshot shows a Go IDE with a file explorer on the left containing files like `modul 5.docx`, `guided1.go`, `guided2.go`, `guided3.go`, `guided4.go`, `guided5.go`, and `guided6.go`. The main editor displays a Go function `printSeries` defined as follows:

```
func printSeries(N, current int) {  
    if current > N {  
        return  
    }  
    fmt.Printf("%d ", current)  
    printSeries(N, current+1)  
}
```

The terminal output shows the execution of the program for `N=12`:

```
*****  
LENVOBDESKTOP-6A8A07 MINGW64 /d/Master/Telkom University Purwokerto/Semester 3/Apro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 5  
(master)  
$ go run "d:\Master\Telkom University Purwokerto\Semester 3\Apro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 5\unguided3.go"  
Masukkan bilangan bulat positif N: 12  
Faktor-faktor dari 12 adalah: 1 2 3 4 6 12  
  
LENVOBDESKTOP-6A8A07 MINGW64 /d/Master/Telkom University Purwokerto/Semester 3/Apro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 5  
(master)  
$ go run "d:\Master\Telkom University Purwokerto\Semester 3\Apro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 5\unguided4.go"  
Masukkan bilangan bulat positif N: 12  
Barisan bilangan dari 12 hingga 1 dan kembali ke 12:  
12 11 10 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9 10 11 12  
  
LENVOBDESKTOP-6A8A07 MINGW64 /d/Master/Telkom University Purwokerto/Semester 3/Apro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 5  
(master)  
$
```

Deskripsi Program

Fungsi `printOddNumbers` Fungsi ini menerima dua parameter: `N` (batas atas) dan `current` (bilangan saat ini yang sedang diperiksa). Jika `current` lebih besar dari `N`, fungsi akan berhenti (base case). Jika `current` adalah bilangan ganjil (diperiksa dengan `current%2 != 0`), maka bilangan tersebut dicetak. Fungsi kemudian memanggil dirinya sendiri dengan `current+1` untuk memeriksa bilangan berikutnya.

Unguided 6

Buatlah program yang menimplementasikan 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

```
//Rasyid Nafsyarie 2311102011 IF 11 06  
package main
```

```
import "fmt"

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

func main() {
    var x, y int
    fmt.Print("Masukkan x: ")
    fmt.Scanln(&x)
    fmt.Print("Masukkan y: ")
    fmt.Scanln(&y)

    fmt.Println(pangkat(x, y))
}
```

Screenshoot Output

The screenshot shows a VS Code editor with a Go file named `unguided6.go` open. The code defines a recursive function `pangkat` and a `main` function. The terminal output shows the program being run, with input values `x=11` and `y=12`, and the resulting output `3138428376721`.

```
1 //Rasyid Nafsyarie 2311102011 IF 11 06
2 package main
3 import "fmt"

LENOVO@DESKTOP-6ABAL07 MINGW64 /d/Master/Telkom University Purwokerto/Semester 3/AIpro 2 Praktikum/2311102011_Rasyid Nafsyarie_Modul 5
$ go run "d:\Master\Telkom University Purwokerto\Semester 3\AIpro 2 Praktikum\2311102011_Rasyid Nafsyarie_Modul 5\unguided6.go"
Masukkan x: 11
Masukkan y: 12
3138428376721
```

Deskripsi Program

Fungsi pangkat Memeriksa nilai y. Jika y sama dengan 0, maka hasilnya adalah 1 (karena setiap bilangan pangkat 0 adalah 1). Jika y sama dengan 1,

maka hasilnya adalah x . Jika y lebih besar dari 1, fungsi memanggil dirinya sendiri dengan y dikurangi 1, dan mengalikan hasilnya dengan x .