

**LAPORAN PRAKTIKUM  
PEMROGRAMAN BERORIENTASI OBJEK**

**MODUL 4  
PROSEDUR**



Oleh:

DAMARA GALUH PEMBAYUN

2311102110

IF-11-02

**S1 TEKNIK INFORMATIKA  
INSTITUT TEKNOLOGI TELKOM PURWOKERTO  
2024**

## **I. DASAR TEORI**

Prosedur, atau kadang-kadang disebut sebagai fungsi, adalah sekumpulan petunjuk yang disusun secara sistematis dan dimaksudkan untuk menyelesaikan tugas tertentu. Prosedur dalam pemrograman adalah blok kode yang dapat dipanggil berulang kali dari berbagai komponen program.

Prosedur adalah konsep fundamental dalam pemrograman yang sangat penting untuk menghasilkan kode yang efisien, mudah dipelihara, dan mudah dibaca. Dengan memahami konsep prosedur, programmer dapat membangun program yang lebih kompleks dan berkualitas.

## II. GUIDED

### Guided 1

#### Source Code

```
package main

import "fmt"

// Fungsi buat menghitung faktorial
func factorial(n int) int {
    if n == 0 || n == 1 {
        return 1
    }
    result := 1
    for i := 2; i <= n; i++ {
        result *= i
    }
    return result
}

// Fungsi buat menghitung permutasi
func permutation(n, r int) int {
    return factorial(n) / factorial(n-r)
}

// Fungsi buat menghitung kombinasi
func combination(n, r int) int {
    return factorial(n) / (factorial(r) * factorial(n-r))
}

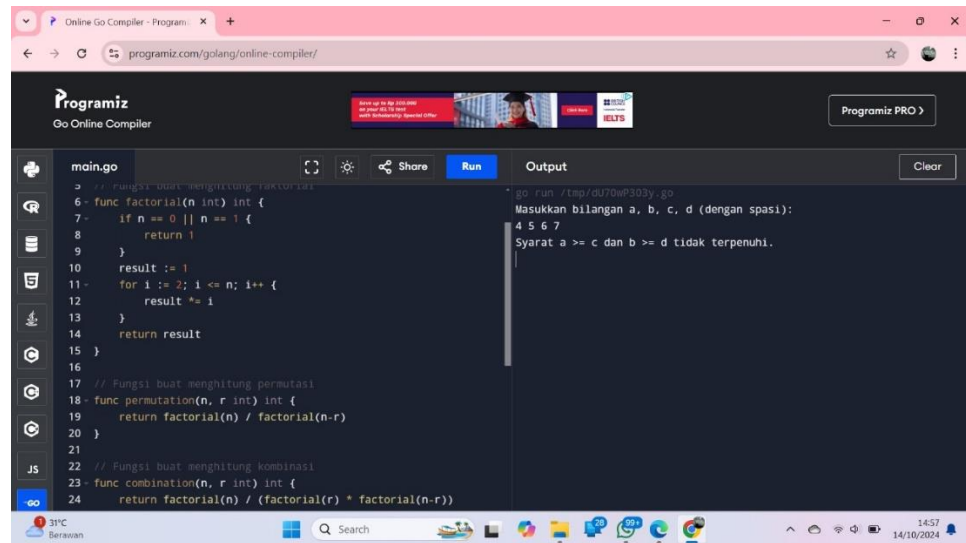
func main() {
    // Input 4 bilangan
    var a, b, c, d int
    fmt.Println("Masukkan bilangan a, b, c, d (dengan spasi): ")
    fmt.Scanf("%d %d %d %d", &a, &b, &c, &d)

    // Cek syarat a >= c dan b >= d
    if a >= c && b >= d {
        // Menghitung permutasi dan kombinasi a dan c
        permutasiAC := permutation(a, c)
        kombinasiAC := combination(a, c)

        // Menghitung permutasi dan kombinasi b dan d
        permutasiBD := permutation(b, d)
        kombinasiBD := combination(b, d)

        // Output hasil
        fmt.Println("Permutasi(a, c) dan Kombinasi(a, c):", permutasiAC, kombinasiAC)
        fmt.Println("Permutasi(b, d) dan Kombinasi(b, d):", permutasiBD, kombinasiBD)
    } else {
        fmt.Println("Syarat a >= c dan b >= d tidak terpenuhi.")
    }
}
```

## Hasil ScreenShoot



## Deskripsi

Program Golang ini dibuat untuk menghitung nilai permutasi dan kombinasi dari dua set bilangan bulat yang diberikan pengguna. Banyak metode untuk menyusun sejumlah objek dalam urutan yang berbeda disebut mutasi, sedangkan kombinasi adalah banyak metode untuk memilih sejumlah objek dari suatu himpunan tanpa memperhatikan urutan objek tersebut.

## Guided 2

### SourchCode

```
package main

import "fmt"

// Fungsi untuk menghitung faktorial
func factorial(n int) int {
    if n == 0 {
        return 1
    }
    result := 1
    for i := 1; i <= n; i++ {
        result *= i
    }
    return result
}

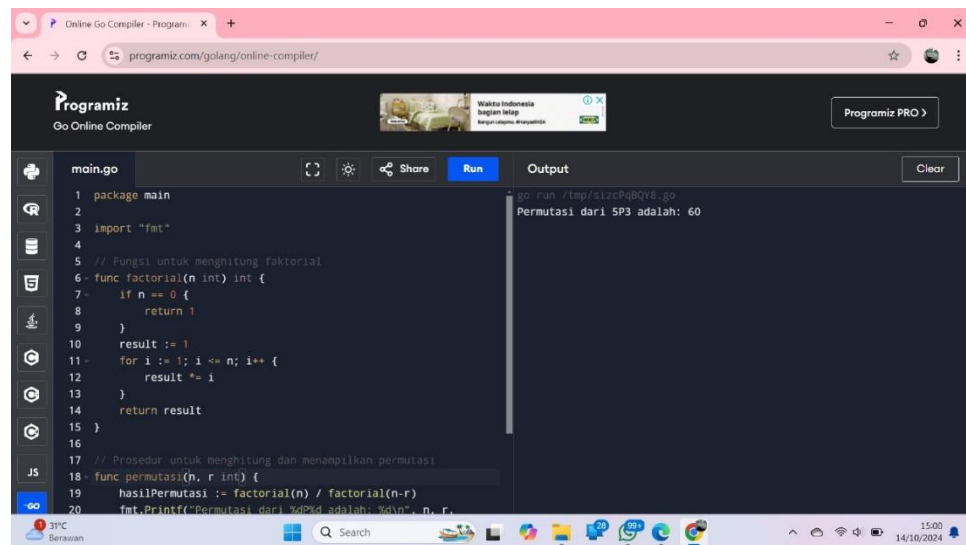
// Prosedur untuk menghitung dan menampilkan permutasi
func permutasi(n, r int) {
    hasilPermutasi := factorial(n) / factorial(n-r)
    fmt.Printf("Permutasi dari %dP%d adalah: %d\n", n, r,
    hasilPermutasi)
}

func main() {
```

```
// Memanggil prosedur untuk menghitung dan menampilkan
permutasi
n, r := 5, 3
permutasi(n, r)
}

//buat soal nomer 1
```

## Hasil ScreenShoot



The screenshot shows the Programiz Online Go Compiler interface. The code editor on the left contains a Go program with the following code:

```
1 package main
2
3 import "fmt"
4
5 // Fungsi untuk menghitung faktorial
6 func factorial(n int) int {
7     if n == 0 {
8         return 1
9     }
10    result := 1
11    for i := 1; i <= n; i++ {
12        result *= i
13    }
14    return result
15 }
16
17 // Prosedur untuk menghitung dan menampilkan permutasi
18 func permutasi(n, r int) {
19     hasilPermutasi := factorial(n) / factorial(n-r)
20     fmt.Printf("Permutasi dari %dPd adalah: %d\n", n, r,
```

The output panel on the right shows the result of running the program:

```
go run /tmp/sizcPg8QY8.go
Permutasi dari 5P3 adalah: 60
```

## Deskripsi

Program ini menyediakan implementasi hitung permutasi sederhana. Fungsi permutasi menghitung permutasi dengan menggunakan hasil dari fungsi factorial, sedangkan fungsi permutasi menghitung nilai permutasi dengan menggunakan hasil dari fungsi factorial. Dalam fungsi permutasi, argumen n dan r menunjukkan jumlah objek total dan jumlah objek yang diambil dalam permutasi. Oleh karena itu, argumen ini sangat penting.

### III. UNGUIDED

#### Unguided 1 ( no.3)

##### SourchCode

```
package main

//Damara Galuh Pembayun 23111202110

import "fmt"

func cetakDeret(n int) {

    for n != 1 {

        fmt.Print(n, " ")

        if n%2 == 0 {

            n /= 2

        } else {

            n = 3*n + 1

        }

    }

    fmt.Println(1)

}

func main() {

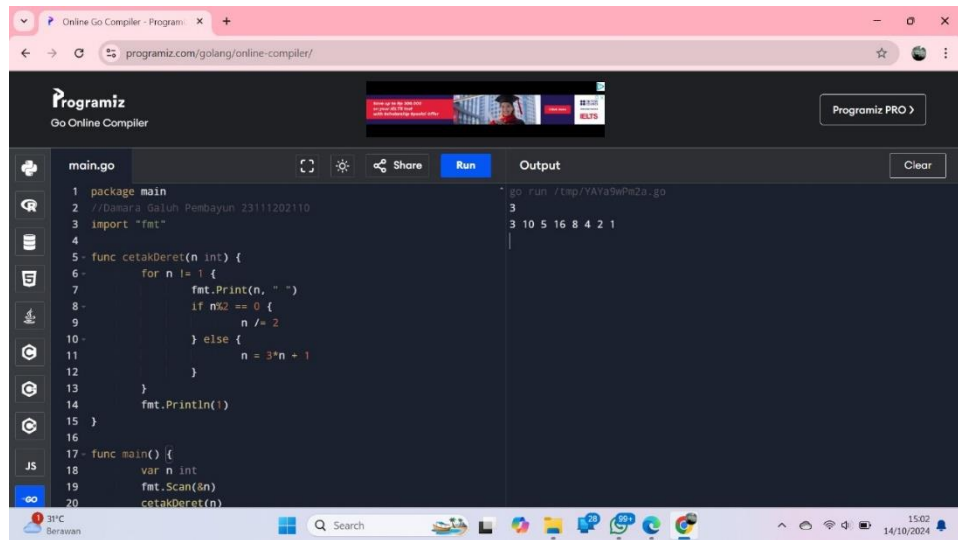
    var n int

    fmt.Scan(&n)

    cetakDeret(n)

}
```

## Hasil ScreenShoot



The screenshot shows the Programiz online Go compiler interface. The code editor contains a Go program named `main.go` that implements a function `cetakDeret` to generate a sequence of numbers based on the Skiena and Revilla rules. The program takes an input of 3 and outputs the sequence 3 10 5 16 8 4 2 1.

```
1 package main
2 //Gocara Galuh Pembayun 23111202110
3 import "fmt"
4
5 func cetakDeret(n int) {
6     for n != 1 {
7         fmt.Print(n, " ")
8         if n%2 == 0 {
9             n /= 2
10        } else {
11            n = 3*n + 1
12        }
13    }
14    fmt.Println()
15 }
16
17 func main() {
18     var n int
19     fmt.Scan(&n)
20     cetakDeret(n)
21 }
```

The output of the program is displayed in the Output panel:

```
go run /tmp/YAYa9wPw2a.go
3
3 10 5 16 8 4 2 1
```

## Deskripsi

Program Golang ini dirancang untuk menghasilkan deret bilangan yang mengikuti aturan khusus yang dikenal sebagai deret Skiena dan Revilla. Deret ini memiliki sifat unik di mana setiap suku berikutnya ditentukan oleh suku sebelumnya berdasarkan aturan tertentu:

- Jika suku saat ini genap, maka suku berikutnya adalah setengah dari suku saat ini.
- Jika suku saat ini ganjil, maka suku berikutnya adalah tiga kali suku saat ini ditambah satu.

Deret ini selalu berakhir pada angka 1.