

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN**

**2**

**MODUL III & IV  
FUNGSI & PROSEDUR**



**Oleh:**

**M. ARIF RACHMAN**

**2311102332**

**IF-11-07**

**S1 TEKNIK INFORMATIKA  
UNIVERSITAS TELKOM PURWOKERTO**

**2024**

## **I.DASAR    TEORI**

### **Modul 3**

Fungsi merupakan satu kesatuan rangkaian instruksi yang memberikan atau menghasilkan suatu nilai dan biasanya memetakan input ke suatu nilai yang lain. Oleh karena itu, fungsi selalu menghasilkan/mengembalikan nilai. Suatu subprogram dikatakan fungsi apabila:

1. Ada deklarasi tipe nilai yang dikembalikan, dan
2. Terdapat kata kunci return dalam badan subprogram.

Maka fungsi digunakan jika suatu nilai biasanya diperlukan, seperti:

- Assignment nilai ke suatu variabel
- Bagian dari ekspresi
- Bagian dari argumen suatu subprogram, dsb.

Karena itu selalu pilih nama fungsi yang menggambarkan nilai, seperti kata benda dan kata sifat.

## **Modul 4**

Prosedur dapat dianggap sebagai potongan beberapa instruksi program menjadi suatu instruksi baru yang dibuat untuk mengurangi kerumitan dari kode program yang kompleks pada suatu program yang besar. Prosedur akan menghasilkan suatu akibat atau efek langsung pada program ketika dipanggil pada program utama. Suatu subprogram dikatakan prosedur apabila:

1. Tidak ada deklarasi tipe nilai yang dikembalikan, dan
2. Tidak terdapat kata kunci return dalam badan subprogram.

Kedudukannya prosedur sama seperti instruksi dasar yang sudah ada sebelumnya (assignment) dan/atau instruksi yang berasal dari paket (fmt), seperti `fmt.Scan` dan `fmt.Print`. Karena itu selalu pilih nama prosedur yang berbentuk kata kerja atau sesuatu yang merepresentasikan proses sebagai nama dari prosedur.

## II. GUIDED

### Guided 1

#### Source Code

```
package main

import (
    "fmt"
)

func volumeTabung(jari_jari, tinggi int) float64
{
    var luasAlas, volume float64
    luasAlas = 3.14 *
    float64(jari_jari*jari_jari)
    volume =
    luasAlas * float64(tinggi)
    return volume
}

func main() {
    var r, t int
    var v1, v2 float64
    r = 5
    t = 10

    v1 = volumeTabung(r, t)
    v2 = volumeTabung(r, t) + volumeTabung(15, t)

    fmt.Println("Volume tabung 14 x 100: ",
    volumeTabung(14, 100))
    fmt.Println("v1: ", v1)
    fmt.Println("v2: ", v2)
}
```

## DESKRIPSI PROGRAM:

Program di atas merupakan implementasi sederhana dalam bahasa Go yang menghitung volume sebuah tabung. Fungsi `volumeTabung` menerima dua parameter, yaitu `jari_jari` dan `tinggi`, yang bertipe data integer, lalu mengembalikan volume tabung sebagai tipe data `float64`. Volume dihitung dengan cara mengalikan luas alas tabung (yang berbentuk lingkaran dengan rumus  $\pi \times \text{jari-jari}^2$ ) dengan tinggi tabung. Fungsi ini menggunakan nilai  $\pi$  sebagai 3.14 untuk mempermudah perhitungan.

## Guided 2

### Source Code

```
package main

import (
    "fmt"
)

func cetakNFibo(n int)
{
    var f1, f2, f3
    int f2 = 0;
    f3 = 1
    for i:=1; i<=n; i++){
        fmt.Println(f3)
        f1 = f2
        f2 = f3
        f3 =
        f1+f2
    }
}

func main(){
    var x int
    x = 5
    cetakNFibo(x)
    cetakNFibo(100)
}
```

**DESKRIPSI PROGRAM:**

Program di atas merupakan implementasi sederhana untuk mencetak deret Fibonacci hingga sejumlah elemen tertentu dalam bahasa Go. Deret Fibonacci adalah deret bilangan di mana setiap bilangan setelah dua bilangan pertama merupakan penjumlahan dari dua bilangan sebelumnya. Fungsi `cetakNFibo(n int)` bertugas mencetak `n` elemen pertama dari deret Fibonacci. Di dalam fungsi ini, tiga variabel digunakan: `f1`, `f2`, dan `f3` untuk menyimpan nilai-nilai dari dua bilangan sebelumnya dan hasil penjumlahan mereka.

### III. UNGUIDED

#### Modul 3

##### Unguided 1

1. Minggu ini, mahasiswa Fakultas Informatika mendapatkan tugas dari mata kuliah matematika diskrit untuk mempelajari kombinasi dan permutasi. Jonas salah seorang mahasiswa, iseng untuk mengimplementasikannya ke dalam suatu program. Oleh karena itu bersediakah kalian membantu Jonas? (tidak tentunya ya :p)

Masukan terdiri dari empat buah bilangan asli a, b, c, dan d yang dipisahkan oleh spasi, dengan syarat  $a \leq c$  dan  $b \geq d$ .

Keluaran terdiri dari dua baris. Baris pertama adalah hasil permutasi dan kombinasi a terhadap c, sedangkan baris kedua adalah hasil permutasi dan kombinasi b terhadap d.

Catatan: permutasi (P) dan kombinasi (C) dari n terhadap r ( $n \geq r$ ) dapat dihitung dengan menggunakan persamaan berikut!

$$P(n, r) = \frac{n!}{(n-r)!}, \text{ sedangkan } C(n, r) = \frac{n!}{r!(n-r)!}$$

Contoh

No	Masukan	Keluaran	Penjelasan
1	5 10 3 10	60 10 3628800 1	$P(5, 3) = 5! / 2! = 120 / 2 = 60$ $C(5, 3) = 5! / (3! \times 2!) = 120 / 12 = 10$ $P(10, 10) = 10! / 0! = 3628800 / 1 = 3628800$ $C(10, 10) = 10! / (10! \times 0!) = 10! / 10! = 1$
2	8 0 2 0	56 28 1 1	

Selesaikan program tersebut dengan memanfaatkan subprogram yang diberikan berikut ini!

```
function factorial(n: integer) → integer
{mengembalikan nilai faktorial dari n}

function permutation(n,r : integer) → integer
{Mengembalikan hasil n permutasi r, dan n >= r}

function combination(n,r : integer) → integer
{Mengembalikan hasil n kombinasi r, dan n >= r}
```



### Source Code:

```
package main
import "fmt"
//M.Arif Rachman
func factorial(n int) int {
if n == 0 {
return 1
}
return n * factorial(n-1)
}

func permutation(n, r int) int {
return factorial(n) / factorial(n-r)
}

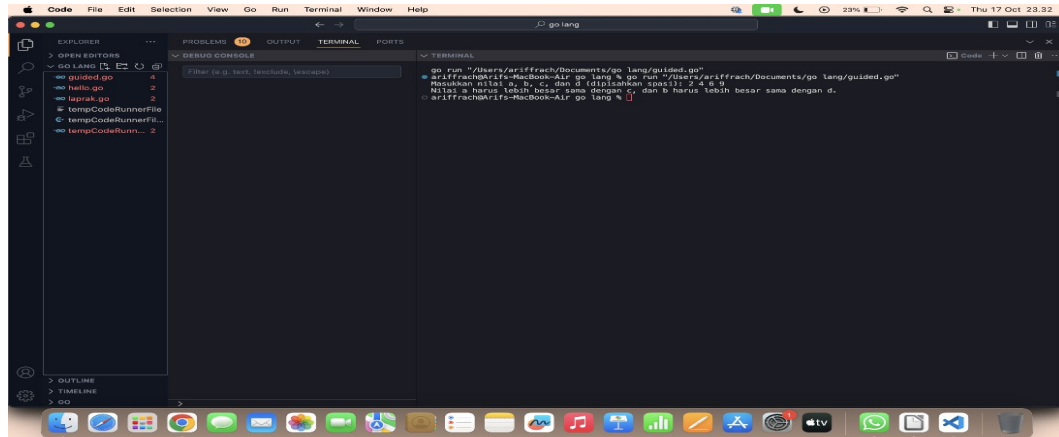
func combination(n, r int) int {
return factorial(n) / (factorial(r) * factorial(n-r))
}

func main() {
var a, b, c, d int
fmt.Print("Masukkan nilai a, b, c, dan d (dipisahkan spasi): ")
fmt.Scan(&a, &b, &c, &d)

if a < c || b < d {
fmt.Println("Nilai a harus lebih besar sama dengan c, dan b harus lebih besar sama
dengan d.")
return
}

fmt.Printf("P(%d,%d) C(%d,%d)\n", a, c, a, c)
fmt.Printf("%d %d\n", permutation(a, c), combination(a, c))
fmt.Printf("P(%d,%d) C(%d,%d)\n", b, d, b, d)
fmt.Printf("%d %d\n", permutation(b, d), combination(b, d))
}
```

## Screenshot Output Program:

A screenshot of a macOS desktop showing a Visual Studio Code editor window. The editor is open to a file named 'guided.go'. The terminal window at the bottom shows the output of running the program. The output text is as follows:

```
go run "/Users/ariffnach/Documents/go_lang/guided.go"  
ariffnach@ariffnach-MacBook-Air go_lang % go run "/Users/ariffnach/Documents/go_lang/guided.go"  
Masukkan nilai a, b, c dan d (dibedakan spasi): 2 4 8 6  
Nilai a harus lebih besar sama dengan c, dan b harus lebih besar sama dengan d.  
ariffnach@ariffnach-MacBook-Air go_lang %
```

## Deskripsi program:

Program ini menghitung nilai permutasi dan kombinasi dari dua set angka. Pengguna diminta memasukkan empat angka: `a`, `b`, `c`, dan `d`. Program kemudian memeriksa apakah `a` lebih besar atau sama dengan `c`, dan `b` lebih besar atau sama dengan `d`. Jika syarat ini terpenuhi, program menghitung dan menampilkan hasil permutasi dan kombinasi dari pasangan `(a, c)` serta `(b, d)`. Permutasi menghitung jumlah susunan berbeda tanpa pengulangan, sedangkan kombinasi menghitung jumlah cara memilih elemen tanpa memperhatikan urutan.

## Unguided 2

2. Diberikan tiga buah fungsi matematika yaitu  $f(x) = x^2$ ,  $g(x) = x - 2$  dan  $h(x) = x + 1$ .

Fungsi komposisi  $(f \circ g \circ h)(x)$  artinya adalah  $f(g(h(x)))$ . Tuliskan  $f(x)$ ,  $g(x)$  dan  $h(x)$  dalam bentuk function.

**Masukan** terdiri dari sebuah bilangan bulat  $\alpha$ ,  $b$  dan  $c$  yang dipisahkan oleh spasi.

**Keluaran** terdiri dari tiga baris. Baris pertama adalah  $(f \circ g \circ h)(\alpha)$ , baris kedua  $(g \circ h \circ f)(b)$ , dan baris ketiga adalah  $(h \circ f \circ g)(c)$ !

### Contoh

No	Masukan	Keluaran	Penjelasan
1	7 2 10	36 3 65	$(f \circ g \circ h)(7) = 36$ $(g \circ h \circ f)(2) = 3$ $(h \circ f \circ g)(10) = 65$
2	5 5 5	16 24 10	$(f \circ g \circ h)(5) = 16$ $(g \circ h \circ f)(5) = 24$ $(h \circ f \circ g)(5) = 10$
3	3 8 4	4 63 5	$(f \circ g \circ h)(5) = 4$ $(g \circ h \circ f)(5) = 63$ $(h \circ f \circ g)(5) = 5$

## Source Code

```
package main
//M. Arif Rachman
import "fmt"

func f(x int) int {
    return x * x
}

func g(x int) int {
    return x - 2
}

func h(x int) int {
    return x + 1
}

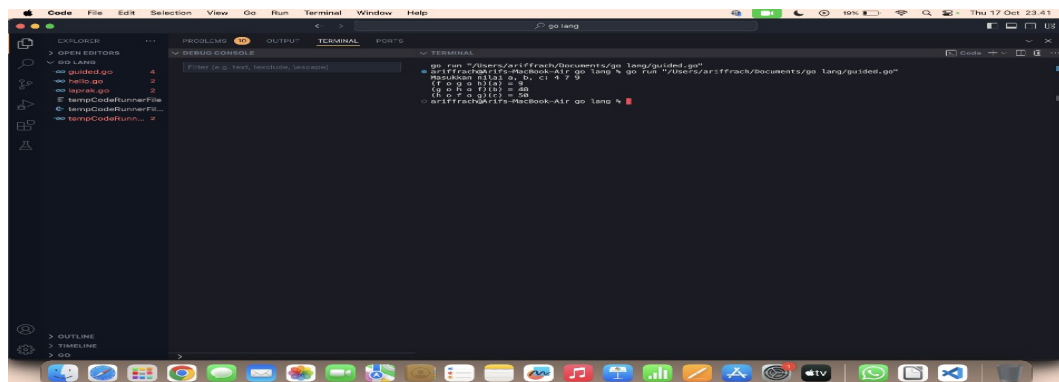
func main() {
    var a, b, c int

    fmt.Print("Masukkan nilai a, b, c: ")
    fmt.Scan(&a, &b, &c)

    fogoh := f(g(h(a)))
    gohof := g(h(f(b)))
    hofog := h(f(g(c)))

    fmt.Println("(f o g o h)(a) =", fogoh)
    fmt.Println("(g o h o f)(b) =", gohof)
    fmt.Println("(h o f o g)(c) =", hofog)
}
```

## Screenshot Program



**Deskripsi Program:**

Program ini melakukan komposisi fungsi matematika  $f$ ,  $g$ , dan  $h$  terhadap tiga nilai yang dimasukkan oleh pengguna:  $a$ ,  $b$ , dan  $c$ . Fungsi  $f$  mengkuadratkan nilai,  $g$  mengurangi nilai dengan 2, dan  $h$  menambahkan 1. Program kemudian menghitung hasil dari komposisi fungsi-fungsi tersebut, yaitu  $(f \circ g \circ h)(a)$ ,  $(g \circ h \circ f)(b)$ , dan  $(h \circ f \circ g)(c)$ , lalu menampilkan hasilnya.

**Unguided 3**

3. [**Lingkaran**] Suatu lingkaran didefinisikan dengan koordinat titik pusat  $(cx, cy)$  dengan radius  $r$ . Apabila diberikan dua buah lingkaran, maka tentukan posisi sebuah titik sembarang  $(x, y)$  berdasarkan dua lingkaran tersebut.

Masukan terdiri dari beberapa tiga baris. Baris pertama dan kedua adalah koordinat titik pusat dan radius dari lingkaran 1 dan lingkaran 2, sedangkan baris ketiga adalah koordinat titik sembarang. Asumsi sumbu  $x$  dan  $y$  dari semua titik dan juga radius direpresentasikan dengan bilangan bulat.

Keluaran berupa string yang menyatakan posisi titik "**Titik di dalam lingkaran 1 dan 2**", "**Titik di dalam lingkaran 1**", "**Titik di dalam lingkaran 2**", atau "**Titik di luar lingkaran 1 dan 2**".

**Contoh**

No	Masukan	Keluaran
1	1 1 5 8 8 4 2 2	Titik di dalam lingkaran 1
2	1 2 3	Titik di dalam lingkaran 2

	4 5 6 7 8	
3	5 10 15 -15 4 20 0 0	Titik di dalam lingkaran 1 dan 2
4	1 1 5 8 8 4 15 20	Titik di luar lingkaran 1 dan 2

Fungsi untuk menghitung jarak titik (a, b) dan (c, d) dimana rumus jarak adalah:

$$jarak = \sqrt{(a - c)^2 + (b - d)^2}$$

dan juga fungsi untuk menentukan posisi sebuah titik sembarang berada di dalam suatu lingkaran atau tidak.

```
function jarak(a,b,c,d : real) -> real
{Mengembalikan jarak antara titik (a,b) dan titik (c,d)}

function didalam(cx,cy,r,x,y : real) -> boolean
{Mengembalikan true apabila titik (x,y) berada di dalam lingkaran yang
memiliki titik pusat (cx,cy) dan radius r}
```

**Catatan:** Lihat paket **math** dalam lampiran untuk menggunakan fungsi **math.Sqrt()** untuk menghitung akar kuadrat.

### Source Code:

```
package main
//M.Arif Rachman
import (
    "fmt"
    "math"
)

func jarak(x1, y1, x2, y2 float64) float64 {
    return math.Sqrt(math.Pow(x2-x1, 2) + math.Pow(y2-y1, 2))
}

func dalamLingkaran(cx, cy, r, x, y float64) bool {
    return jarak(cx, cy, x, y) <= r
}

func main() {
    var cx1, cy1, r1 float64
    fmt.Println("Masukkan cx1, cy1, r1 (Lingkaran 1): ")
    fmt.Scan(&cx1, &cy1, &r1)

    var cx2, cy2, r2 float64
    fmt.Println("Masukkan cx2, cy2, r2 (Lingkaran 2): ")
    fmt.Scan(&cx2, &cy2, &r2)

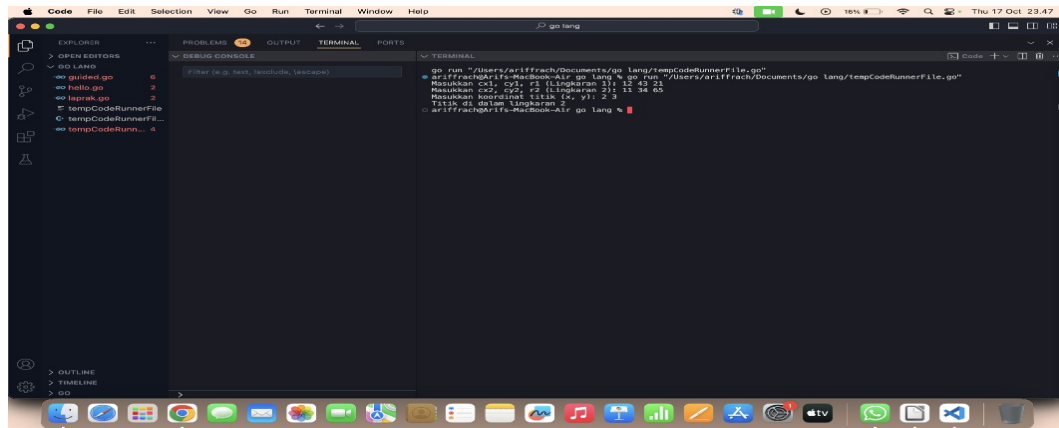
    var x, y float64
    fmt.Println("Masukkan koordinat titik (x, y): ")
    fmt.Scan(&x, &y)

    dalam1 := dalamLingkaran(cx1, cy1, r1, x, y)
    dalam2 := dalamLingkaran(cx2, cy2, r2, x, y)

    if dalam1 && dalam2 {
        fmt.Println("Titik di dalam lingkaran 1 dan 2")
    } else if dalam1 {
        fmt.Println("Titik di dalam lingkaran 1")
    } else if dalam2 {
        fmt.Println("Titik di dalam lingkaran 2")
    } else {
        fmt.Println("Titik di luar lingkaran 1 dan 2")
    }
}
```



## Screenshot Output Program:

A screenshot of a macOS desktop showing a Visual Studio Code editor window. The editor has a dark theme and shows a Go file named 'goLang.go'. The file explorer on the left shows the project structure. The terminal window at the bottom right shows the execution of the program. The output of the program is as follows:

```
go run "/Users/arifrach/Documents/goLang/tempCodeRunnerFile.go"  
Masukkan cx, cy, r2 (lingkaran 1): 11 43 21  
Masukkan cx2, cy2, r2 (lingkaran 2): 11 34 85  
Masukkan koordinat titik (x, y): 2 3  
Titik di dalam lingkaran 1  
arifrach@arifra-macbook-air goLang %
```

## Deskripsi Program:

Program ini menghitung apakah sebuah titik berada di dalam satu atau dua lingkaran yang diberikan. Pengguna diminta memasukkan pusat ('cx', 'cy') dan jari-jari ('r') dari dua lingkaran, serta koordinat titik yang akan diuji. Program kemudian menghitung jarak titik ke pusat masing-masing lingkaran dan menentukan apakah titik tersebut berada di dalam lingkaran 1, lingkaran 2, atau keduanya, serta menampilkan hasilnya.

## Modul 4

### Unguided 4

1. Minggu ini, mahasiswa Fakultas Informatika mendapatkan tugas dari mata kuliah matematika diskrit untuk mempelajari kombinasi dan permutasi. Jonas salah seorang mahasiswa, iseng untuk mengimplementasikannya ke dalam suatu program. Oleh karena itu bersediakah kalian membantu Jonas? (tidak tentunya ya :p)

### Contoh

No	Masukan	Keluaran	Penjelasan
1	5 10 3 10	60 10 3628800 1	$P(5,3) = 5!/2! = 120/2 = 60$ $C(5,3) = 5!/(3! \times 2!) = 120/12 = 10$ $P(10,10) = 10!/0! = 3628800/1 = 3628800$ $C(10,10) = 10!/(10! \times 0!) = 10!/10! = 1$
2	8 0 2 0	56 28 1 1	

Selesaikan program tersebut dengan memanfaatkan prosedur yang diberikan berikut ini!

```

procedure factorial(in n: integer, in/out hasil:integer)
{I.S. terdefinisi bilangan bulat positif n
 F.S. hasil berisi nilai faktorial dari n}

procedure permutation(in n,r : integer, in/out hasil:integer)
{I.S. terdefinisi bilangan bulat positif n dan r, dan n >= r
 F.S. hasil berisi nilai dari n permutasi r}

procedure combination(in n,r : integer, in/out hasil:integer)
{I.S. terdefinisi bilangan bulat positif n dan r, dan n >= r
 F.S. hasil berisi nilai dari n kombinasi r}
  
```

### Source Code:

```
package main
//M.Arif Rachman
import "fmt"

func factorial(n int, result *int) {
if n == 0 {
*result = 1
} else {
*result = n
temp := 1
factorial(n-1, &temp)
*result *= temp
}
}

func permutation(n, r int, result *int) {
var factN, factNR int
factorial(n, &factN)
factorial(n-r, &factNR)
*result = factN / factNR
}

func combination(n, r int, result *int) {
var factN, factR, factNR int
factorial(n, &factN)
factorial(r, &factR)
factorial(n-r, &factNR)
*result = factN / (factR * factNR)
}

func main() {
var a, b, c, d int

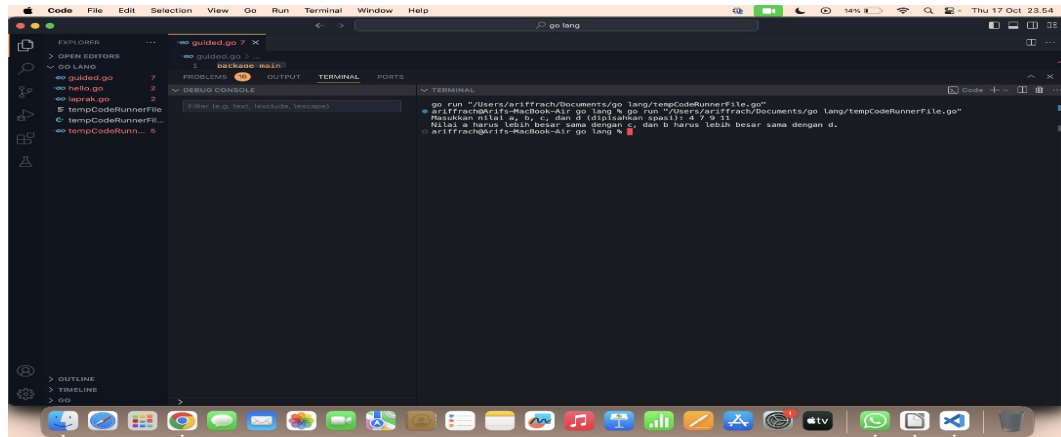
fmt.Print("Masukkan nilai a, b, c, dan d (dipisahkan spasi): ")
fmt.Scan(&a, &b, &c, &d)

if a < c || b < d {
fmt.Println("Nilai a harus lebih besar sama dengan c, dan b harus lebih besar sama
dengan d.")
return
}

var permAC, combAC, permBD, combBD int

permutation(a, c, &permAC)
```

## Screenshot Output Program:

A screenshot of the Visual Studio Code interface showing the output of a Go program. The Explorer sidebar on the left shows a project named 'go-lang' with files 'guided.go', 'hello.go', 'israk.go', 'tempCodeRunnerFile', and 'tempCodeRunner...'. The main editor area displays the 'guided.go' file. The Output panel at the bottom right shows the program's execution results. The output text is as follows:

```
go run "/Users/arifrach/Documents/go-lang/tempCodeRunnerFile.go"  
# arifrach@Arif-MacBook-Air go-lang % go run "/Users/arifrach/Documents/go-lang/tempCodeRunnerFile.go"  
Masukkan nilai a, b, c, dan d (dipisahkan spasi): 4 7 9 11  
Nilai a harus lebih besar sama dengan c, dan b harus lebih besar sama dengan d.  
arifrach@Arif-MacBook-Air go-lang %
```

## Deskripsi Program:

Program ini menghitung permutasi dan kombinasi dari dua set nilai yang dimasukkan oleh pengguna. Fungsi `factorial` menggunakan rekursi untuk menghitung faktorial, yang kemudian digunakan oleh fungsi `permutation` dan `combination` untuk menghitung permutasi dan kombinasi berdasarkan rumus matematika. Program meminta pengguna memasukkan empat nilai: `a`, `b`, `c`, dan `d`, lalu menghitung dan menampilkan hasil permutasi dan kombinasi dari pasangan (`a`, `c`) dan (`b`, `d`).

## Unguided 5

2. Kompetisi pemrograman tingkat nasional berlangsung ketat. Setiap peserta diberikan 8 soal yang harus dapat diselesaikan dalam waktu 5 jam saja. Peserta yang berhasil menyelesaikan soal paling banyak dalam waktu paling singkat adalah pemenangnya.

Buat program **gema** yang mencari pemenang dari daftar peserta yang diberikan. Program harus dibuat modular, yaitu dengan membuat prosedur `hitungSkor` yang mengembalikan total soal dan total skor yang dikerjakan oleh seorang peserta, melalui parameter formal. Pembacaan nama peserta dilakukan di program utama, sedangkan waktu pengerjaan dibaca di dalam prosedur.

**prosedure** `hitungSkor`(in/out soal, skor : integer)

Setiap baris **masukan** dimulai dengan satu string nama peserta tersebut diikuti dengan adalah 8 integer yang menyatakan berapa lama (dalam menit) peserta tersebut menyelesaikan soal. Jika tidak berhasil atau tidak mengirimkan jawaban maka otomatis dianggap menyelesaikan dalam waktu 5 jam 1 menit (301 menit).

Satu baris **keluaran** berisi nama pemenang, jumlah soal yang diselesaikan, dan nilai yang diperoleh. Nilai adalah total waktu yang dibutuhkan untuk menyelesaikan soal yang berhasil diselesaikan.

No	Masukan	Keluaran
1	Astuti 20 50 301 301 61 71 75 10 Bertha 25 47 301 26 50 60 65 21 Selesai	Bertha 7 294

### Keterangan:

Astuti menyelesaikan 6 soal dalam waktu 287 menit, sedangkan Bertha 7 soal dalam waktu 294 menit. Karena Bertha menyelesaikan lebih banyak, maka Bertha menang. Jika keduanya menyelesaikan sama banyak, maka pemenang adalah yang menyelesaikan dengan total waktu paling kecil.

### Source Code:

```
package main
import "fmt"
type Peserta struct {
    Nama    string
    Waktu   []int
    Skor    int
    Selesai int
}

func hitungSkor(peserta *Peserta) {
    peserta.Skor = 0
    peserta.Selesai = 0

    for _, waktu := range peserta.Waktu {
        if waktu == 301 {
            // Skor tidak bertambah jika waktu 301
            continue
        } else {
            peserta.Skor += waktu
            peserta.Selesai++
        }
    }
}

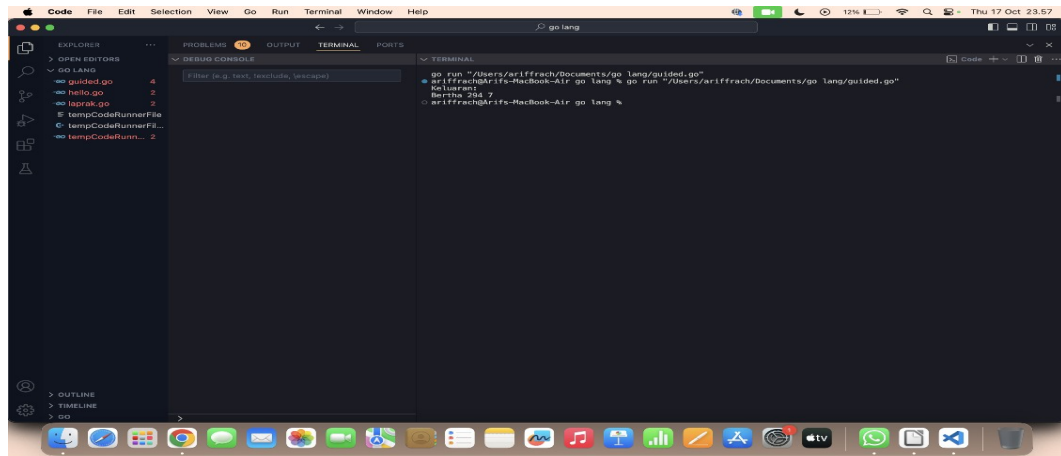
func main() {
    peserta := []Peserta{
        {Nama: "Astuti", Waktu: []int{20, 50, 301, 301, 61, 71, 75, 10}},
        {Nama: "Bertha", Waktu: []int{25, 47, 301, 26, 50, 60, 65, 21}},
    }

    for i := range peserta {
        hitungSkor(&peserta[i])
    }

    var pemenang Peserta
    for i := range peserta {
        if peserta[i].Selesai > pemenang.Selesai || (peserta[i].Selesai ==
pemenang.Selesai && (pemenang.Skor == 0 || peserta[i].Skor < pemenang.Skor)) {
            pemenang = peserta[i]
        }
    }

    fmt.Printf("Keluaran:\n")
    fmt.Printf("%s %d %d\n", pemenang.Nama, pemenang.Skor, pemenang.Selesai)
}
```

## Screenshot Output Program:



## Deskripsi Program:

Program ini menentukan pemenang dari beberapa peserta berdasarkan waktu yang dihabiskan untuk menyelesaikan tugas. Setiap peserta memiliki waktu dalam detik untuk beberapa tugas, dengan nilai khusus 301 menunjukkan tugas yang tidak selesai. Program menghitung total waktu yang dihabiskan peserta untuk tugas-tugas yang selesai dan berapa banyak tugas yang berhasil diselesaikan. Pemenang adalah peserta yang menyelesaikan paling banyak tugas, dan jika ada kesamaan, peserta dengan total waktu terkecil yang menang.

## Unguided 6

3.

Skiena dan Revilla dalam *Programming Challenges* mendefinisikan sebuah deret bilangan. Deret dimulai dengan sebuah bilangan bulat  $n$ . Jika bilangan  $n$  saat itu genap, maka suku berikutnya adalah  $\frac{1}{2}n$ , tetapi jika ganjil maka suku berikutnya bernilai  $3n+1$ . Rumus yang sama digunakan terus menerus untuk mencari suku berikutnya. Deret berakhir ketika suku terakhir bernilai 1. Sebagai contoh jika dimulai dengan  $n=22$ , maka deret bilangan yang diperoleh adalah:

22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

Untuk suku awal sampai dengan 1000000, diketahui deret selalu mencapai suku dengan nilai 1.

Buat program **skiena** yang akan mencetak setiap suku dari deret yang dijelaskan di atas untuk nilai suku awal yang diberikan. Pencetakan deret harus dibuat dalam prosedur cetakDeret yang mempunyai 1 parameter formal, yaitu nilai dari suku awal.

**prosedure** cetakDeret(in  $n$  : **integer** )

**Masukan** berupa satu bilangan integer positif yang lebih kecil dari 1000000.

**Keluaran** terdiri dari satu baris saja. Setiap suku dari deret tersebut dicetak dalam baris yang dan dipisahkan oleh sebuah spasi.

No	Masukan	Keluaran
1	22	22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1



### Source Code:

```
package main
//.M Arif Rachman
import "fmt"

func cetakDeret(n int) {
    for n != 1 {
        fmt.Printf("%d ", n)
        if n%2 == 0 {
            n = n / 2
        } else {
            n = 3*n + 1
        }
    }
    fmt.Println(n)
}

func main() {
    var n int
    fmt.Print("Masukkan bilangan bulat positif < 1000000: ")
    fmt.Scan(&n)

    if n > 0 && n < 1000000 {
        cetakDeret(n)
    } else {
        fmt.Println("Bilangan harus positif < 1000000")
    }
}
```

### Screenshot Output Program:



```
6 3 10 5 16 8 4 2 1
```

**Deskripsi Program:** Program ini menampilkan deret angka berdasarkan aturan matematika sederhana. Pengguna memasukkan bilangan bulat positif kurang dari 1 juta. Jika angka tersebut genap, angka dibagi dua; jika ganjil, angka dikalikan tiga lalu ditambahkan satu. Proses ini berlanjut hingga angka menjadi 1, dan deret hasilnya ditampilkan. Jika angka yang dimasukkan tidak memenuhi syarat, program akan memberikan pesan kesalahan.