

**LAPORAN PRAKTIKUM  
ALGORITMA DAN PEMROGRAMAN 2  
MODUL 2  
REVIEW STRUKTUR KONTROL**



Oleh:

SAVILA NUR FADILLA

103112400031

IF-12-01

**S1 TEKNIK INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO  
2025**

# I. DASAR TEORI

## 2.1 Struktur Program Go

```
// Setiap program utama dimulai dengan "package main"
package main
// Impor paket yang dibutuhkan, "fmt" berisi proses I/O standar
import "fmt"
// Kode program utama dalam "fungsi main"
func main() {
...
}
```

## 2.2 Tipe Data dan Instruksi Dasar

Notasi tipe dasar	Tipe dalam Go
integer	int
real	float32 float64
boolean	bool
karakter	byte rune
string	string

## 2.3 Struktur Kontrol Perulangan

- 1.) Bentuk while loop  
while (kondisi) do  
... kode yang diulang  
endwhile
- 2.) Bentuk repeat until  
repeat  
.. kode yang diulang  
until (kondisi)

## 2.4 Struktur Kontrol Percabangan

- 1.) Bentuk if else
  - a. if (kondisi) then  
.. kode untuk kondisi true  
endif

- b. if (kondisi) then
  - .. kode untuk kondisi true
  - else
  - .. kode untuk kondisi false
  - endif
  
- c. if (kondisi-1) then
  - .. kode untuk kondisi-1 true
  - else if (kondisi-2) then
  - .. kode untuk kondisi-2 true
  - .. dst. dst.
  - else
  - .. kode jika semua kondisi
  - .. di atas false
  - endif

2.) Bentuk switch case  
depend on expresi  
nilai\_1:  
.. kode jika ekspresi bernilai\_1  
nilai\_2:  
.. kode jika ekspresi bernilai\_2  
.. dst. dst.  
}

## II. GUIDED

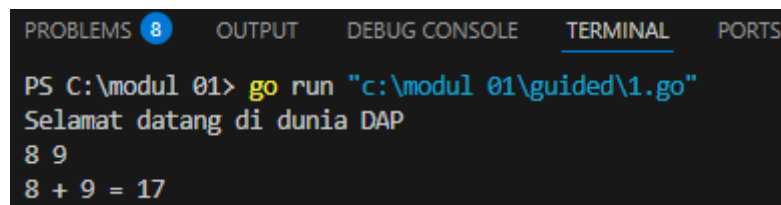
### 1.) Source Code

```
package main

import "fmt"

func main() {
    var greetings = "Selamat datang di dunia DAP"
    var a, b int
    fmt.Println(greetings)
    fmt.Scanln(&a, &b)
    fmt.Printf("%v + %v = %v\n", a, b, a+b)
}
```

Output :

A screenshot of a terminal window with a dark background. At the top, there are tabs for 'PROBLEMS' (with a blue circle containing the number 8), 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL' (which is selected and underlined), and 'PORTS'. The terminal shows the command 'PS C:\modul 01> go run "c:\modul 01\guided\1.go"' followed by the output: 'Selamat datang di dunia DAP', '8 9', and '8 + 9 = 17'.

Penjelasan : Program ini bertujuan untuk mencetak penjumlahan dari dua bilangan bulat a dan b.

- `var greetings = "Selamat datang di dunia DAP"` → deklarasi variable greetings berupa teks Selamat datang di dunia DAP
- `var a, b int` → deklarasi variable a dan b berupa integer atau bilangan bulat
- `fmt.Println(greetings)` → mencetak variable greetings
- `fmt.Scanln(&a, &b)` → membaca dan menerima inputan variable a dan b (inputan berupa bilangan bulat)
- `fmt.Printf("%v + %v = %v\n", a, b, a+b)` → mencetak variable a, b, dan hasil penjumlahan dari a dan b

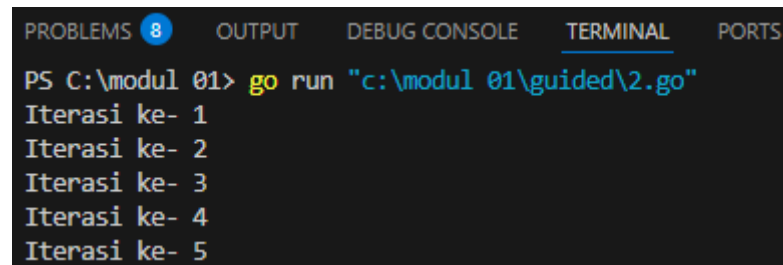
## 2.) Source Code

```
package main

import "fmt"

func main() {
    for i := 1; i <= 5; i++ {
        fmt.Println("Iterasi ke-", i)
    }
}
```

Output :



```
PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\modul 01> go run "c:\modul 01\guided\2.go"
Iterasi ke- 1
Iterasi ke- 2
Iterasi ke- 3
Iterasi ke- 4
Iterasi ke- 5
```

Penjelasan : Program ini bertujuan untuk mencetak iterasi ke sekian.

- for i := 1; i <= 5 → inialisasi variable i yang dimulai dari 1 sampai dengan 5. Kondisi perulangan akan berjalan selama nilai i di antara 1 sampai dengan 5
- i++ → nilai i bertambah 1 di setiap iterasi
- fmt.Println("Iterasi ke-", i) → mencetak iterasi ke sekian

### 3.) Source Code

```
package main

import "fmt"

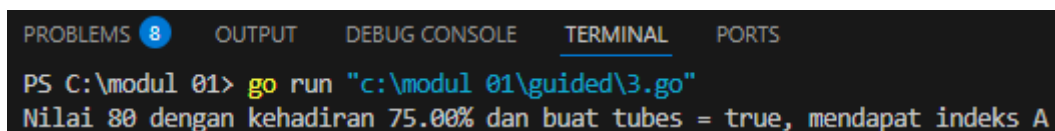
func main() {
    //variabel bisa diganti sesuai kebutuhan
    nilai := 80
    pctHadir := 0.75
    adaTubes := true

    var indeks string

    if nilai > 75 && adaTubes {
        indeks = "A"
    } else if nilai > 65 {
        indeks = "B"
    } else if nilai > 50 && pctHadir > 0.7 {
        indeks = "C"
    } else {
        indeks = "F"
    }

    fmt.Printf("Nilai %v dengan kehadiran %.2f%% dan buat tubes = %t,
mendapat indeks %s\n", nilai, pctHadir*100, adaTubes, indeks)
}
```

Output :

A screenshot of a terminal window with a dark background. At the top, there are tabs labeled 'PROBLEMS' (with a blue circle containing the number 8), 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL' (which is selected and underlined), and 'PORTS'. Below the tabs, the command prompt shows 'PS C:\modul 01> go run "c:\modul 01\guided\3.go"'. The output of the program is displayed below the command: 'Nilai 80 dengan kehadiran 75.00% dan buat tubes = true, mendapat indeks A'.

Penjelasan : Program ini bertujuan untuk mencetak indeks nilai.

- nilai := 80 → nilai
- pctHadir := 0.75 → persentase hadir
- adaTubes := true → mengerjakan tugas besar
- if nilai > 75 && adaTubes, indeks = "A" → jika nilai di atas 75 dan mengerjakan tugas besar, maka mendapat indeks A
- else if nilai > 65, indeks = "B" → jika nilai di atas 65, maka mendapat indeks B
- else if nilai > 50 && pctHadir > 0.7, indeks = "C" → jika nilai di atas 50 dan persentase hadir di atas 0,7 maka mendapat indeks C
- else, indeks = "F" → jika tidak memenuhi semua syarat, maka mendapat indeks F

- `fmt.Printf("Nilai %v dengan kehadiran %.2f%% dan buat tubes = %t, mendapat indeks %s\n", nilai, pctHadir*100, adaTubes, indeks)` → mencetak hasil akhir berupa indeks nilai dengan format nilai, persentase hadir, buat tubes atau tidak, dan indeks nilai yang didapatkan berdasarkan syarat syaratnya

### III. UNGUIDED

- 1.) Tahun kabisat adalah tahun yang habis dibagi 400 atau habis dibagi 4 tetapi tidak habis dibagi 100. Buatlah sebuah program yang menerima input sebuah bilangan bulat dan memeriksa apakah bilangan tersebut merupakan tahun kabisat (true) atau bukan (false).

Source code

```
// Savila Nur Fadilla
// 103112400031

package main

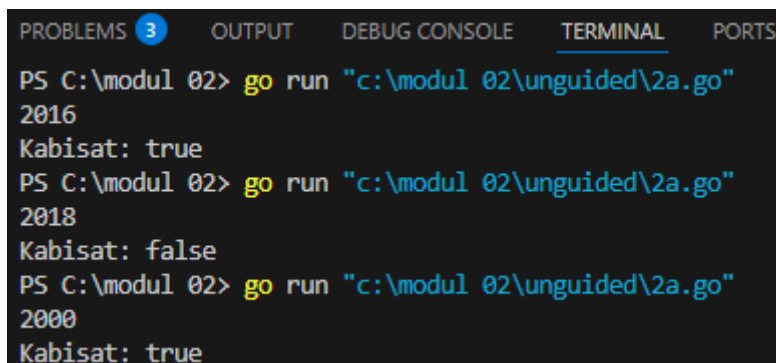
import "fmt"

func main() {
    var tahun int
    fmt.Scan(&tahun)

    kabisat := tahun % 400 == 0 || (tahun % 4 == 0 && tahun % 100 != 0)

    if kabisat {
        fmt.Print("Kabisat: true")
    } else {
        fmt.Print("Kabisat: false")
    }
}
```

Output :



```
PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\modul 02> go run "c:\modul 02\unguided\2a.go"
2016
Kabisat: true
PS C:\modul 02> go run "c:\modul 02\unguided\2a.go"
2018
Kabisat: false
PS C:\modul 02> go run "c:\modul 02\unguided\2a.go"
2000
Kabisat: true
```



Penjelasan : Program ini bertujuan untuk memeriksa apakah tahun tersebut merupakan tahun kabisat atau bukan.

- `var tahun int` → deklarasi variable tahun berupa integer atau bilangan bulat
- `fmt.Scan(&tahun)` → membaca dan menerima inputan tahun
- `kabisat := tahun % 400 == 0 || (tahun % 4 == 0 && tahun % 100 != 0)` → mengecek kabisat. Tahun kabisat adalah jika tahun habis dibagi 400 atau habis dibagi 4 tetapi tidak habis dibagi 100
- `if kabisat, fmt.Print("Kabisat: true")` → jika kabisat, program mencetak output berupa string Kabisat: true
- `else, fmt.Print("Kabisat: false")` → jika tidak, program mencetak output berupa string Kabisat: false

- 2.)  $\sqrt{2}$  merupakan bilangan irasional. Meskipun demikian, nilai tersebut dapat dihampiri dengan rumus berikut:

$$\sqrt{2} = \prod_{k=0}^{\infty} \frac{(4k+2)^2}{(4k+1)(4k+3)}$$

Modifikasi program sebelumnya yang menerima input integer  $K$  dan menghitung  $\sqrt{2}$  untuk  $K$  tersebut. Hampiran  $\sqrt{2}$  dituliskan dalam ketelitian 10 angka di belakang koma.

Source code

```
// Savila Nur Fadilla
// 103112400031

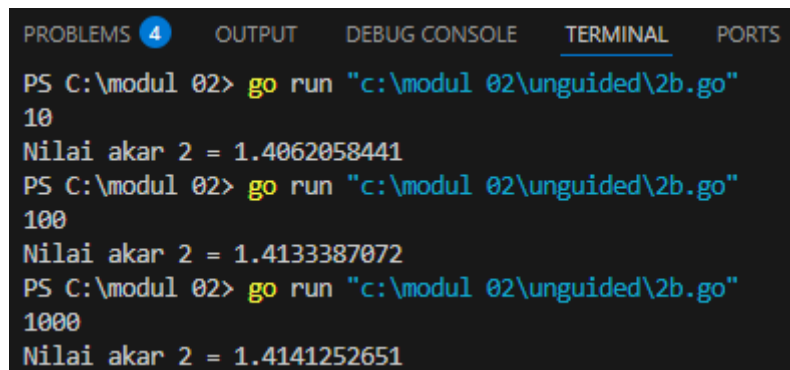
package main

import "fmt"

func main() {
    var k int
    var akar2 float64
    fmt.Scan(&k)
    akar2 = 1.0

    for i := 0; i <= k; i++ {
        akar2 *= float64((4 * i + 2) * (4 * i + 2)) / float64((4 * i + 1) * (4
* i + 3))
    }
    fmt.Printf("Nilai akar 2 = %.10f\n", akar2)
}
```

Output :



```
PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\modul 02> go run "c:\modul 02\unguided\2b.go"
10
Nilai akar 2 = 1.4062058441
PS C:\modul 02> go run "c:\modul 02\unguided\2b.go"
100
Nilai akar 2 = 1.4133387072
PS C:\modul 02> go run "c:\modul 02\unguided\2b.go"
1000
Nilai akar 2 = 1.4141252651
```

Penjelasan : Program ini bertujuan untuk Menghitung nilai akar 2.

- `var k int` → deklarasi variable k berupa integer atau bilangan bulat
- `var akar2 float64` → deklarasi variable akar 2 berupa bilangan real
- `fmt.Scan(&k)` → membaca dan menerima inputan k
- `akar2 = 1.0` → nilai akar 2 dimulai dari 1.0 (berupa bilangan decimal/real)
- `for i := 0; i <= k; i++` → iterasi dimulai dari 0 sampai dengan k, perulangan terus berlanjut sampai nilai k yang memenuhi
- `i++` → nilai i bertambah 1 di setiap iterasi
- `akar2 *= float64((4 * i + 2) * (4 * i + 2)) / float64((4 * i + 1) * (4 * i + 3))` → rumus untuk menentukan nilai akar 2
- `fmt.Printf("Nilai akar 2 = %.10f\n", akar2)` → mencetak nilai akar 2 dengan 10 angka di belakang koma

- 3.) PT POS membutuhkan aplikasi perhitungan biaya kirim berdasarkan berat parcel. Maka, buatlah program BiayaPos untuk menghitung biaya pengiriman tersebut dengan ketentuan sebagai berikut! Dari berat parcel (dalam gram), harus dihitung total berat dalam kg dan sisanya (dalam gram). Biaya jasa pengiriman adalah Rp. 10.000,- per kg. Jika sisa berat tidak kurang dari 500 gram, maka tambahan biaya kirim hanya Rp. 5,- per gram saja. Tetapi jika kurang dari 500 gram, maka tambahan biaya akan dibebankan sebesar Rp. 15,- per gram. Sisa berat (yang kurang dari 1kg) digratiskan biayanya apabila total berat ternyata lebih dari 10kg.

Source code

```
// Savila Nur Fadilla
// 10311240031

package main

import "fmt"

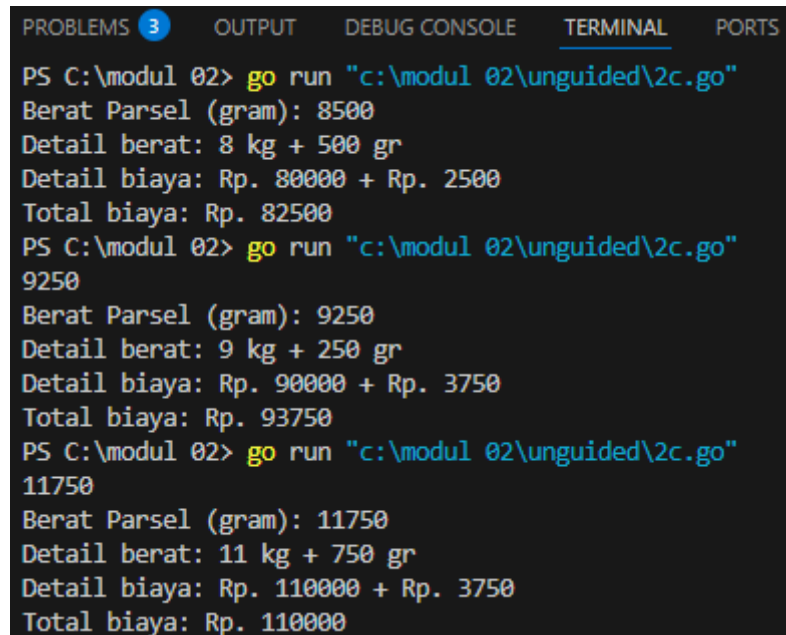
func main() {
    var berat, kg, gr, biayaKg, biayaGr, total int
    fmt.Scan(&berat)
    kg = berat / 1000
    gr = berat % 1000
    biayaKg = kg * 10000

    if gr >= 500 {
        biayaGr = gr * 5
    } else {
        biayaGr = gr * 15
    }

    if kg > 10 {
        total = biayaKg
    } else {
        total = biayaKg + biayaGr
    }

    fmt.Printf("Berat berat (gram): %v\n", berat)
    fmt.Printf("Detail berat: %v kg + %v gr\n", kg, gr)
    fmt.Printf("Detail biaya: Rp. %v + Rp. %v\n", biayaKg, biayaGr)
    fmt.Printf("Total biaya: Rp. %v", total)
}
```

Output :



```
PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\modul 02> go run "c:\modul 02\unguided\2c.go"
Berat Parsel (gram): 8500
Detail berat: 8 kg + 500 gr
Detail biaya: Rp. 80000 + Rp. 2500
Total biaya: Rp. 82500
PS C:\modul 02> go run "c:\modul 02\unguided\2c.go"
9250
Berat Parsel (gram): 9250
Detail berat: 9 kg + 250 gr
Detail biaya: Rp. 90000 + Rp. 3750
Total biaya: Rp. 93750
PS C:\modul 02> go run "c:\modul 02\unguided\2c.go"
11750
Berat Parsel (gram): 11750
Detail berat: 11 kg + 750 gr
Detail biaya: Rp. 110000 + Rp. 3750
Total biaya: Rp. 110000
```

Penjelasan : Program ini bertujuan untuk menghitung biaya kirim berdasarkan berat parsel.

- var berat, kg, gr, biayaKg, biayaGr, total int → deklarasi variable berupa integer atau bilangan bulat
- fmt.Scan(&berat) → membaca dan menerima inputan berat, yaitu berat parsel dalam gram
- kg = berat / 1000 → menghitung jumlah kilogram, mengkonversi berat total parsel dari gram ke kilogram
- gr = berat % 1000 → menghitung jumlah gram, mengambil sisa bagi dari total berat parsel dengan 1000
- biayaKg = kg \* 10000 → menghitung biaya kilogram, mengalikan jumlah kilogram dengan 1000
- if gr >= 500, biayaGr = gr \* 5 → jika gram lebih dari atau sama dengan 500, maka biaya gram adalah jumlah gram dikali 5
- else, biayaGr = gr \* 15 → jika tidak, maka biaya gram adalah jumlah gram dikali 15
- if kg > 10, total = biayaKg → jika kilogram lebih dari 10, maka total biayanya adalah biaya kilogram
- else, total = biayaKg + biayaGr → jika tidak, maka total biaya adalah biaya kilogram ditambah biaya gram
- fmt.Printf("Berat Parsel (gram): %v\n", berat) → mencetak output berat parsel dalam gram
- fmt.Printf("Detail berat: %v kg + %v gr\n", kg, gr) → mencetak output detail beratnya

- `fmt.Printf("Detail biaya: Rp. %v + Rp. %v\n", biayaKg, biayaGr)` → mencetak output detail biaya
- `fmt.Printf("Total biaya: Rp. %v", total)` → mencetak output total biaya

#### **IV. KESIMPULAN**

Berdasarkan laporan ini, dapat disimpulkan jika soal soal tersebut menggunakan struktur pemrograman Bahasa Go yang berbeda beda. Ada yang menggunakan struktur perulangan, dan ada yang menggunakan struktur percabangan if else. Terdapat berbagai tipe data juga yang digunakan dalam program program tersebut.

Go hanya mempunyai kata kunci for untuk semua jenis perulangan yang kita pelajari dalam notasi algoritma. Dalam konsep pemrograman terstruktur, setiap rancangan algoritma harus memenuhi syarat satu pintu masuk dan satu pintu keluar. Karena itu tidaklah diperkenankan untuk membuat program sumber yang mempunyai struktur loop yang mempunyai pintu keluar lebih dari satu,

Banyak bentuk if-else yang mungkin dilakukan dalam bahasa Go. Ada yang merupakan satu instruksi if-else-endif saja (hanya satu endif). Bentuk if-else yang bersarang (dengan beberapa endif) dapat dibentuk dengan komposisi beberapa if-else-endif tersebut.

## **V. REFERENSI**

Telkom University. (2025). *Modul Praktikum Algoritma dan Pemrograman 2*.