

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



Oleh:

MUHAMMAD AGHA ZULFADHLI

2311102015

S1-IF11-02

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

## I. DASAR TEORI

Dalam kerangka program yang ditulis dalam bahasa pemrograman Go, program utama selalu mempunyai dua komponen berikut:

- `package main` merupakan penanda bahwa file ini berisi program utama.
- `func main()` berisi kode utama dari sebuah program Go.

Semua proses terkait bahasa Go dilakukan melalui utilitas `go`. Beberapa opsi dengan utilitas `go`:

- **go build**: mengkompilasi program sumber yang ada dalam folder menjadi sebuah program.
- **go build file.go**: mengkompilasi program sumber `file.go` saja.
- **go fmt**: membaca semua program sumber dalam folder dan mereformat penulisannya agar sesuai dengan standar penulisan program sumber Go.
- **go clean**: membersihkan file-file dalam folder sehingga tersisa program sumber nya saja.

Variabel adalah nama dari suatu lokasi di memori, yang data dengan tipe tertentu dapat disimpan.

Notasi tipe dasar	Tipe dalam Go	Keterangan
integer	<code>int</code> <code>int8</code> <code>int32 //rune</code> <code>int64</code> <code>uint</code> <code>uint8 //byte</code> <code>uint32</code> <code>uint64</code>	bergantung platform 8 bit: -128..127 32 bit: -10 <sup>9</sup> ..10 <sup>9</sup> 64 bit: -10 <sup>19</sup> ..10 <sup>19</sup> bergantung platform 0..255 0..4294967295 0..(2 <sup>64</sup> -1)
real	<code>float32</code> <code>float64</code>	32bit: -3.4E+38 .. 3.4E+38 64bit: -1.7E+308 .. 1.7E+308
boolean (atau logikal)	<b><code>bool</code></b>	<b><code>false</code> dan <code>true</code></b>
karakter	<code>byte //uint8</code> <code>rune //int32</code>	tabel ASCII/UTF-8 tabel UTF-16
string	<code>string</code>	

Operasi yang dapat dilakukan terhadap tipe data di atas adalah

Operator dalam Go	Tipe data terkait	Keterangan
+	string integer dan real	konkatenasi 2 string operasi penjumlahan
- * /	integer dan real	operasi pengurangan, perkalian, dan pembagian
%	integer	operasi sisa pembagian integer (modulo)
&   ^ &^	integer	operasi <b>per-bit</b> AND, OR, XOR, AND-NOT
<< >>	integer dan unsigned integer	operasi geser bit kiri/kanan sebanyak unsigned integer yang diberikan
< <= >= > == !=	selain boolean	komparasi menghasilkan nilai boolean komparasi karakter sesuai dengan posisi karakter tersebut dalam tabel ASCII/UTF-16 komparasi string sesuai dengan operasi karakter per karakter, dimulai dari karakter paling kiri (awal)
&&    !	boolean	operasi <b>boolean</b> AND, OR, dan NOT
* &	variabel apasaja	mendapatkan data dari lokasi memori dan mendapatkan lokasi dari variabel

Bahasa Go menerapkan sistem tipe data yang ketat, sehingga tipe data yang berbeda tidak dapat dicampurkan dalam satu ekspresi, termasuk tipe yang sejenis seperti int dan int32. Untuk menyesuaikan tipe data, ada beberapa metode yang dapat digunakan, seperti casting, yang mengubah tipe data yang ada menjadi tipe yang diinginkan dengan sintaks tipe(data).

Konversi tipe	Data	Tipe baru	Keterangan
tipe(data)	integer	integer	format data tidak berubah, hanya penyesuaian jumlah bit. Kekurangan bit diisi bit 0 di sebelah kiri (MSB)
	real	real	format data tidak berubah, hanya penyesuaian jumlah bit. Kekurangan bit, maka bit mantisa diisi bit 0.
	real	integer	format data disesuaikan dengan tipe data tujuan
	integer	real	format data disesuaikan dengan tipe data tujuan
fmt.Sprintf("%v",v)	any type	string	tulis output ke string
fmt.Sprintf("%c",v)	karakter	string	tulis karakter ke string
fmt.Sscanf(s,"%v",&v)	string	any type	baca string ke variabel dengan tipe tertentu
fmt.Sscanf(s,"%c",&v)	string	karakter	baca string ke variabel bertipe karakter

Sebelum digunakan, variabel harus dideklarasikan terlebih dahulu. Selain itu, variabel juga perlu diinisialisasi agar nilai yang disimpan dapat diketahui dengan jelas, sehingga eksekusi algoritma menjadi lebih terprediksi. Dalam bahasa Go, jika variabel tidak diinisialisasi, secara otomatis akan diisi dengan nilai default yang setara dengan bit 0.

Notasi deklarasi variabel	Penulisan dalam Go	Keterangan
kamus a : tipe	var a tipe	a diinisialisasi dengan nilai default
kamus a : tipe  algoritma a <- nilai_awal	var a tipe = nilai_awal var a = (tipe)nilai_awal	a diinisialisasi dengan nilai_awal
	a := nilai_awal a := (tipe)nilai_awal	secara <b>implisit</b> , tipe variabel a ditentukan dari nilai inisialisasinya

Konstanta dapat diberi nama agar lebih mudah diingat maksud dan kegunaannya. Contohnya, nama PI dapat digunakan untuk merepresentasikan konstanta PI.

```
const PI = 3.14
```

Go hanya memiliki satu kata kunci, yaitu `for`, yang digunakan untuk semua jenis perulangan yang dipelajari dalam notasi algoritma. Dua bentuk perulangan yang digunakan di sini adalah struktur `while-loop` dan `repeat-until`.

	Notasi algoritma	Penulisan dalam bahasa Go
1 2 3 4	while (kondisi) do ... kode yang diulang endwhile	for kondisi { .. kode yang diulang }

	Notasi Algoritma	Penulisan dalam bahasa Go
1 2 3 4 5 6 7 8 9	repeat .. kode yang diulang until (kondisi)	for selesai:=false; !selesai; { .. kode yang diulang selesai = kondisi }  for selesai:=false; !selesai; selesai=kondisi { .. kode yang diulang }

Dalam analisis kasus, bahasa Go mendukung dua bentuk percabangan yang dapat digunakan, yaitu if-else dan switch-case. Struktur if-else memungkinkan programmer untuk mengeksekusi blok kode berdasarkan kondisi tertentu, sementara `switch-case` menawarkan cara yang lebih terstruktur untuk menangani beberapa kondisi berdasarkan nilai variabel. Kedua bentuk percabangan ini memudahkan pengembang dalam mengatur alur eksekusi program dengan jelas dan efektif.

	Notasi algoritma	Penulisan dalam bahasa Go
1 2 3	if (kondisi) then .. kode untuk kondisi true endif	if kondisi { .. kode untuk kondisi true }
4 5 6 7 8	if (kondisi) then .. kode untuk kondisi true else .. kode untuk kondisi false endif	if kondisi { .. kode untuk kondisi true } else { .. kode untuk kondisi false }
9 10 11 12 13 14 15 16 17	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	if kondisi_1 { .. kode untuk kondisi_1 true } else if kondisi_2 { .. kode untuk kondisi_2 true .. dst. dst. } else { .. kode jika semua kondisi .. di atas false }

	Notasi algoritma	Penulisan dalam bahasa Go
1 2 3 4 5 6 7 8 9 10	depend on ekspresi nilai_1: .. kode jika ekspresi bernilai_1 nilai_2: .. kode jika ekspresi bernilai_2 .. dst. dst. }	switch ekspresi { case nilai_1: .. kode jika ekspresi bernilai_1 case nilai_2: .. kode jika ekspresi bernilai_2 .. dst. dst. default: .. kode jika tidak ada nilai .. yang cocok dengan ekspresi }
11 12 13 14 15 16 17 18 19 20	depend on (daftar variabel) kondisi_1: .. kode jika ekspresi_1 true kondisi_2: .. kode jika ekspresi_2 true .. dst. dst. }	switch { case kondisi_1: .. kode jika ekspresi_1 true case kondisi_2: .. kode jika ekspresi_2 true .. dst. dst. default: .. jika tidak ada ekspresi .. yang bernilai true }

## II. GUIDED

### 1. Guided 1

#### Source code

```
package main

import "fmt"

func main() {
    var (
        satu, dua, tiga string
        temp            string
    )
    fmt.Print("Masukan input string: ")
    fmt.Scanln(&satu)
    fmt.Print("Masukan input string: ")
    fmt.Scanln(&dua)
    fmt.Print("Masukan input string: ")
    fmt.Scanln(&tiga)
    fmt.Println("Output awal = " + satu + " " + dua + " " + tiga)
    temp = satu
    satu = dua
    dua = tiga
    tiga = temp
    fmt.Println("Output akhir = " + satu + " " + dua + " " + tiga)
}
```

#### Screenshoot program



```
PS D:\Personal\KULIAH\ALPRO 2\MODUL 2> go run .\GUIDED1.go
Masukan input string: satu
Masukan input string: dua
Masukan input string: tiga
Output awal = satu dua tiga
Output akhir = dua tiga satu
PS D:\Personal\KULIAH\ALPRO 2\MODUL 2>
```

#### Deskripsi program

Kode di atas meminta pengguna untuk memasukkan tiga input string, menyimpannya ke dalam variabel satu, dua, dan tiga. Setelah input diberikan, program menampilkan output awal dengan menggabungkan ketiga string yang diinput. Kemudian, program menukar nilai string dengan memindahkan nilai variabel dua ke satu, nilai tiga ke dua, dan nilai satu awal ke tiga menggunakan

variabel sementara temp. Setelah proses pertukaran selesai, program menampilkan hasil akhir dari pertukaran tersebut.

## 2. Guided 2

### Source code

```
package main

import "fmt"

func main() {
    year := 0
    fmt.Scan(&year)
    fmt.Println((year%4 == 0 && year%100 != 0) || (year%400 == 0))
}
```

### Screenshoot program



```
PS D:\Personal\KULIAH\ALPRO 2\MODUL 2> go run .\GUIDED2.go
1982
false
PS D:\Personal\KULIAH\ALPRO 2\MODUL 2> go run .\GUIDED2.go
2004
true
PS D:\Personal\KULIAH\ALPRO 2\MODUL 2>
```

### Deskripsi program

Kode di atas memeriksa apakah suatu tahun adalah tahun kabisat atau bukan. Pengguna diminta memasukkan tahun yang akan disimpan dalam variabel year. Logika di dalam fmt.Println mengecek dua kondisi: (1) apakah tahun tersebut habis dibagi 4 tetapi tidak habis dibagi 100, atau (2) apakah tahun tersebut habis dibagi 400. Jika salah satu dari kondisi tersebut terpenuhi, program akan mencetak true, menandakan bahwa tahun tersebut adalah tahun kabisat; jika tidak, akan mencetak false.

## 3. Guided 3

### Source code

```
package main

import (
    "fmt"
    "math"
```

```

)

func main() {
    var r float64
    fmt.Print("Jejari = ")
    fmt.Scan(&r)
    volume := 4.0 / 3.0 * math.Pi * math.Pow(r, 3)
    luas := 4 * math.Pi * math.Pow(r, 2)
    fmt.Println("Bola dengan jejari ", r, "memiliki volume",
volume, "dan luas kulit", luas)
}

```

### Screenshoot program



```

PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2A3\2A3.go"
Jejari = 5
Bola dengan jejari 5 memiliki volume 523.598775598299 dan luas kulit 314.1592653589793
PS D:\Personal\KULIAH\ALPRO_2>

```

### Deskripsi program

Program digunakan untuk menghitung volume dan luas permukaan bola berdasarkan jejari (radius) yang dimasukkan oleh pengguna. Setelah meminta input nilai jejari  $r$ , program menghitung volume bola menggunakan rumus  $\frac{4}{3}\pi r^3$  dan luas permukaan menggunakan rumus  $4\pi r^2$ , dengan memanfaatkan konstanta  $\pi$  dari paket `math`. Hasil perhitungan volume dan luas permukaan kemudian dicetak ke layar, memberikan informasi lengkap tentang bola yang memiliki jejari tersebut.



### III. UNGUIDED

#### 1. 2B1


##### Source code

```
package main

import (
    "fmt"
    "strings"
)

func main() {
    urutan := "merahkuninghijauungu"
    var gelas1, gelas2, gelas3, gelas4 string
    var hasil bool = true
    for i := 0; i < 5; i++ {
        fmt.Print("Percobaan ", i+1, ": ")
        fmt.Scan(&gelas1, &gelas2, &gelas3, &gelas4)
        if urutan != strings.ToLower(gelas1+gelas2+gelas3+gelas4) {
            hasil = false
        }
    }
    fmt.Print("Berhasil : ", hasil)
}
```

##### Screenshoot program



```
TERMINAL OUTPUT PROBLEMS ... Code + - [ ] [ ] ... ^ x
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B1\2B1.go"
Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: merah kuning hijau ungu
Percobaan 5: merah kuning hijau ungu
Berhasil : true
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B1\2B1.go"
Percobaan 1: merah kuning hijau ungu
Percobaan 2: merah kuning hijau ungu
Percobaan 3: merah kuning hijau ungu
Percobaan 4: ungu kuning hijau merah
Percobaan 5: merah kuning hijau ungu
Berhasil : false
PS D:\Personal\KULIAH\ALPRO_2> [ ]
```

##### Deskripsi program

Kode di atas adalah program Go yang meminta pengguna memasukkan 4 string (gelas1, gelas2, gelas3, gelas4) sebanyak 5 kali, untuk mencocokkan urutan warna

"merahkuninghijauungu" dengan input yang diberikan. Setiap kali input dimasukkan, program menggabungkan keempat string tersebut menjadi satu, mengubahnya menjadi huruf kecil dengan fungsi `strings.ToLower()`, lalu membandingkannya dengan string urutan. Jika ada satu percobaan di mana urutannya tidak sesuai, variabel hasil diubah menjadi false. Setelah 5 percobaan, program akan menampilkan apakah input berhasil (semua percobaan sesuai) atau tidak, dengan mencetak nilai variabel hasil.

## 2. 2B2

### Source code

```
package main

import "fmt"

func main() {
    var n int
    var bunga, pita string
    fmt.Print("N : ")
    fmt.Scan(&n)
    for i := 0; i < n; i++ {
        fmt.Print("Bunga ", i+1, ": ")
        fmt.Scan(&bunga)
        pita += bunga + " - "
    }
    fmt.Print("Pita: ", pita)
}
```

### Screenshoot program



```
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B2\2B2.go"
N : 3
Bunga 1: Kertas
Bunga 2: Mawar
Bunga 3: Tulip
Pita: Kertas - Mawar - Tulip -
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B2\2B2.go"
N : 0
Pita:
PS D:\Personal\KULIAH\ALPRO_2>
```

### Deskripsi program

Kode di atas adalah program Go yang meminta pengguna untuk memasukkan sejumlah nama bunga, kemudian menambahkan setiap nama bunga ke dalam string pita, dipisahkan dengan tanda " - ". Program pertama-tama meminta input integer n, yang menentukan jumlah bunga yang akan dimasukkan. Dalam loop sebanyak n kali, pengguna diminta memasukkan nama bunga, lalu setiap nama bunga ditambahkan ke variabel pita diikuti oleh " - ". Setelah semua bunga dimasukkan, program menampilkan hasil akhir string pita, yang berisi semua nama bunga yang dimasukkan dengan format "bunga1 - bunga2 - ...".

### 3. 2B3

#### Source code

```
package main

import "fmt"

func main() {
    var kiri, kanan float32
    for kiri < 9 && kanan < 9 {
        fmt.Print("Masukan berat belanjaan di kedua kantong: ")
        fmt.Scan(&kiri, &kanan)
    }
    fmt.Println("Proses selesai.")
}
```

#### Screenshoot program



```
2B3.go - ALPRO_2 - Visual Studio Code
TERMINAL OUTPUT PROBLEMS DEBUG CONSOLE PORT
PS D:\Personal\KULIAH\ALPRO_2> go run d:\Personal\KULIAH\ALPRO_2\MOOUL_2\2B3\2B3.go
Masukan berat belanjaan di kedua kantong: 5.5 1.0
Masukan berat belanjaan di kedua kantong: 7.1 8.5
Masukan berat belanjaan di kedua kantong: 2.6
Masukan berat belanjaan di kedua kantong: 9.5.8
Proses selesai.
PS D:\Personal\KULIAH\ALPRO_2>
```

### Deskripsi program

Program Go di atas dirancang untuk menerima input berat belanjaan dari dua kantong (kiri dan kanan) dan menentukan apakah sepeda motor pak Andi akan oleng berdasarkan perbandingan berat kedua kantong tersebut. Program akan

meminta pengguna untuk memasukkan berat belanjaan di kedua kantong berulang kali selama total berat (jumlah dari kiri dan kanan) kurang dari 150.

**modifikasi program tersebut, program akan menampilkan true jika selisih kedua isi kantong lebih dari atau sama dengan 9 kg. Program berhenti memproses apabila total berat isi kedua kantong melebihi 150 kg atau salah satu kantong beratnya negatif.**

```
package main

import "fmt"

func main() {
    var kiri, kanan float32
    for kiri+kanan < 150 {
        fmt.Print("Masukan berat belanjaan di kedua kantong: ")
        fmt.Scan(&kiri, &kanan)
        fmt.Println("Sepeda motor pak Andi akan oleng: ", (kiri >
kanan+9 || kanan > kiri+9))
    }
    fmt.Println("Proses selesai.")
}
```

### Screenshoot program



```
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B3\tempCodeRunnerFile.go"
Masukan berat belanjaan di kedua kantong: 5 10
Sepeda motor pak Andi akan oleng: false
Masukan berat belanjaan di kedua kantong: 55.6 70.2
Sepeda motor pak Andi akan oleng: true
Masukan berat belanjaan di kedua kantong: 72.3 66.9
Sepeda motor pak Andi akan oleng: false
Masukan berat belanjaan di kedua kantong: 59.5 98.7
Sepeda motor pak Andi akan oleng: true
Proses selesai.
PS D:\Personal\KULIAH\ALPRO_2>
```

### Deskripsi program

Program Go di atas dirancang untuk menerima input berat belanjaan dari dua kantong (kiri dan kanan) dan menentukan apakah sepeda motor pak Andi akan oleng berdasarkan perbandingan berat kedua kantong tersebut. Program akan meminta pengguna untuk memasukkan berat belanjaan di kedua kantong berulang

kali selama total berat (jumlah dari kiri dan kanan) kurang dari 150. Di dalam loop, setelah input berat dimasukkan, program mencetak pernyataan apakah sepeda motor akan oleng atau tidak, dengan menggunakan kondisi ( $kiri > kanan + 9$  ||  $kanan > kiri + 9$ ). Ini berarti sepeda motor akan oleng jika berat salah satu kantong lebih dari 9 kg lebih berat daripada kantong lainnya. Setelah total berat mencapai atau melebihi 150, program keluar dari loop dan mencetak "Proses selesai."

**Modifikasi program sebelumnya, proses input akan berhenti apabila user mengetikkan 'SELESAI'. Kemudian tampilkan isi pita beserta banyaknya bunga yang ada di dalam pita**

```
package main

import (
    "fmt"
    "strings"
)

func main() {
    var n int
    var bunga, pita string
    for i := 0; ; i++ {
        n = i
        fmt.Print("Bunga ", i+1, ": ")
        fmt.Scan(&bunga)
        if strings.ToLower(bunga) == "selesai" {
            break
        }
        pita += bunga + " - "
    }
    fmt.Println("Pita: ", pita)
    fmt.Println("Bunga: ", n)
}
```

## Screenshoot program



```
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B2_2.go"
Bunga 1: Kertas
Bunga 2: Mawar
Bunga 3: Tulip
Bunga 4: selesai
Bunga: 3
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B2_2.go"
Bunga 1: selesai
Pita:
Bunga: 0
PS D:\Personal\KULIAH\ALPRO_2>
```

## Deskripsi program

Program menerima input nama bunga dari pengguna hingga pengguna memasukkan kata "selesai". Di dalam loop tak terhingga (for i := 0; ; i++), program meminta pengguna untuk memasukkan nama bunga, dan setiap nama bunga yang dimasukkan akan ditambahkan ke string pita, dipisahkan dengan " - ". Variabel n mencatat jumlah bunga yang telah dimasukkan dengan menggunakan nilai i, yang akan bertambah setiap kali loop dijalankan. Ketika pengguna mengetik "selesai" (dalam huruf besar atau kecil, berkat fungsi strings.ToLower), program keluar dari loop. Setelah itu, program mencetak hasil akhir dari string pita, yang berisi semua nama bunga yang dimasukkan, serta total jumlah bunga yang dicatat di variabel n.

## 4. 2B4

### Source code

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var k, hasil float64
    fmt.Print("Nilai K = ")
    fmt.Scan(&k)
    hasil = math.Pow(4*k+2, 2) / ((4*k + 1) * (4*k + 3))
    fmt.Printf("Nilai f(K) = %.10f", hasil)
}
```

## Screenshoot program



```
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\284\284.go"
Nilai K = 100
Nilai f(K) = 1.0000061880
PS D:\Personal\KULIAH\ALPRO_2>
```

## Deskripsi program

Program menghitung nilai fungsi matemati berdasarkan input nilai k yang dimasukkan oleh pengguna. Setelah meminta pengguna untuk memasukkan nilai k, program menggunakan rumus  $f(k) = \frac{(4k + 2)^2}{(4k + 1)(4k + 3)}$  untuk menghitung hasilnya. Hasil akhirnya dicetak dengan format yang menunjukkan 10 angka desimal setelah titik desimal.

**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

```
package main

import (
    "fmt"
    "math"
)

func main() {
    var k, hasil float64 = 0, 1
    fmt.Print("Nilai K = ")
    fmt.Scan(&k)
    for i := 0; i < int(k); i++ {
        hasil *= math.Pow(4*float64(i)+2, 2) / ((4*float64(i) + 1)
        * (4*float64(i) + 3))
    }
    fmt.Printf("Nilai akar 2 = %.10f", hasil)
```

## Screenshoot program



```
2B4_2.go - ALPRO_2 - Visual Studio Code
TERMINAL OUTPUT PROBLEMS DEBUG CONSOLE PORTS
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B4_2\tempCodeRunnerFile.go"
Nilai K = 10
Nilai akar 2 = 1.4054086752
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B4_2\tempCodeRunnerFile.go"
Nilai K = 100
Nilai akar 2 = 1.413329615
PS D:\Personal\KULIAH\ALPRO_2> go run "d:\Personal\KULIAH\ALPRO_2\MODUL_2\2B4_2\tempCodeRunnerFile.go"
Nilai K = 1000
Nilai akar 2 = 1.4142151768
PS D:\Personal\KULIAH\ALPRO_2>
```

## Deskripsi program

Program menghitung nilai perkalian bertingkat berdasarkan input nilai K yang dimasukkan oleh pengguna dan menggunakan rumus untuk menghitung hasil yang berkaitan dengan akar dua. Setelah pengguna memasukkan nilai K, program melakukan iterasi dari 0 hingga K-1 menggunakan loop for. Dalam setiap iterasi, program menghitung nilai berdasarkan rumus deret akar 2 dan mengalikan hasilnya dengan variabel hasil, yang diinisialisasi dengan 1. Setelah menyelesaikan semua iterasi, program mencetak hasil akhir dari perkalian tersebut dengan format yang menunjukkan 10 angka desimal setelah titik desimal

### 5. 2C1

#### Source code

```
package main

import (
    "fmt"
)

func main() {
    var berat int

    fmt.Print("Masukkan berat parsel (gram): ")
    fmt.Scanf("%d", &berat)

    kg := berat / 1000
    grams := berat % 1000

    original := kg * 10000
    tambahan := 0

    if kg >= 10 {
        tambahan = 0
    } else {
        if grams < 500 {
```



```

        tambahan = grams * 15
    } else {
        tambahan = grams * 5
    }
}

total := original + tambahan

fmt.Printf("Detail berat: %d kg + %d gr\n", kg, grams)
    fmt.Printf("Detail biaya: Rp. %d + Rp. %d\n", original,
tambahan)
    fmt.Printf("Total biaya: Rp. %d\n", total)
}

```

### Screenshoot program

```

TERMINAL  OUTPUT  PROBLEMS  ...
PS D:\Personal\KULIAH\ALPRO_2> go run d:\Personal\KULIAH\ALPRO_2\MODUL_2\2C1\2C1.go
Masukkan berat parcel (gram): 8500
Detail berat: 8 kg + 500 gr
Detail biaya: Rp. 80000 + Rp. 2500
Total biaya: Rp. 82500
PS D:\Personal\KULIAH\ALPRO_2> go run d:\Personal\KULIAH\ALPRO_2\MODUL_2\2C1\2C1.go
Masukkan berat parcel (gram): 9250
Detail berat: 9 kg + 250 gr
Detail biaya: Rp. 90000 + Rp. 3750
Total biaya: Rp. 93750
PS D:\Personal\KULIAH\ALPRO_2> go run d:\Personal\KULIAH\ALPRO_2\MODUL_2\2C1\2C1.go
Masukkan berat parcel (gram): 11750
Detail berat: 11 kg + 750 gr
Detail biaya: Rp. 110000 + Rp. 0
Total biaya: Rp. 110000
PS D:\Personal\KULIAH\ALPRO_2>

```

### Deskripsi program

Program Go di atas menghitung total biaya pengiriman berdasarkan berat parcel yang dimasukkan oleh pengguna dalam gram. Setelah pengguna memasukkan berat, program menghitung konversi berat tersebut menjadi kilogram (kg) dan gram (grams). Biaya pengiriman dasar dihitung sebagai original, yang merupakan biaya untuk setiap kilogram, ditetapkan pada Rp. 10.000 per kilogram. Jika berat kilogram kurang dari 10, program kemudian menentukan biaya tambahan berdasarkan berat gram: jika kurang dari 500 gram, biaya tambahan dihitung dengan mengalikan gram dengan 15; jika 500 gram atau lebih, biaya tambahan dihitung dengan mengalikan gram dengan 5. Setelah semua biaya dihitung, program mencetak detail berat, biaya dasar, biaya tambahan, dan total biaya pengiriman.

## 6. 2C2

**a. Jika nam diberikan adalah 80.1, apa keluaran dari program tersebut? Apakah eksekusi program tersebut sesuai spesifikasi soal?**

Baris 8: if nam > 80, kondisi ini benar karena 80.1 lebih besar dari 80, sehingga nilai nmk akan diisi dengan "A". Namun, Keluaran dari program tersebut adalah "D". Ini tidak sesuai dengan spesifikasi soal, karena menurut tabel di soal, untuk nilai NAM > 80 = A, nilai akhir seharusnya "A".

**b. Apa saja kesalahan dari program tersebut? Mengapa demikian? Jelaskan alur program seharusnya!**

Program melakukan pengecekan kondisi satu per satu sehingga menimbulkan nilai yang seharusnya lima puluh ke atas mendapatkan nilai D

Semisal kita memiliki nilai 80 dan ketika pengecekan di baris 8 terjadi, seharusnya program tidak melakukan pengecekan kondisi lainnya. Menyebabkan variabel NAM di didefinisikan lagi dengan nilai yang salah.

**c. Perbaiki program tersebut! Ujilah dengan masukan: 93.5; 70.6; dan 49.5. Seharusnya keluaran yang diperoleh adalah 'A', 'B', dan 'D'.**

Berikut adalah perbaikan program:

```
package main
import "fmt"

func main() {
    var nam float64
    var nmk string
    fmt.Print("Nilai akhir mata kuliah: ")
    fmt.Scanln(&nam)

    if nam > 80 {
        nmk = "A"
    } else if nam > 72.5 {
        nmk = "AB"
    } else if nam > 65 {
        nmk = "B"
    } else if nam > 57.5 {
```

```

        nmk = "BC"
    } else if nam > 50 {
        nmk = "C"
    } else if nam > 40 {
        nmk = "D"
    } else {
        nmk = "E"
    }

    fmt.Println("Nilai mata kuliah:", nmk)
}

```

The screenshot shows a terminal window with the following content:

```

PS D:\Personal\KULIAH\ALPRO_2> go run d:\Personal\KULIAH\ALPRO_2\MODUL_2\2C2\2C2.go
Nilai akhir mata kuliah: 93.5
Nilai mata kuliah: A
PS D:\Personal\KULIAH\ALPRO_2> go run d:\Personal\KULIAH\ALPRO_2\MODUL_2\2C2\2C2.go
Nilai akhir mata kuliah: 70.6
Nilai mata kuliah: B
PS D:\Personal\KULIAH\ALPRO_2> go run d:\Personal\KULIAH\ALPRO_2\MODUL_2\2C2\2C2.go
Nilai akhir mata kuliah: 49.5
Nilai mata kuliah: D
PS D:\Personal\KULIAH\ALPRO_2>

```

Program Go di atas digunakan untuk menentukan nilai huruf dari nilai akhir mata kuliah berdasarkan input nilai yang dimasukkan oleh pengguna. Setelah meminta pengguna untuk memasukkan nilai akhir dalam bentuk angka desimal, program kemudian menggunakan serangkaian pernyataan kondisional if dan else if untuk menentukan kategori nilai huruf yang sesuai.

## 7. 2C3

### Source code

```

package main

import "fmt"

func main() {
    var bil int
    var prima bool = true
    fmt.Print("Bilangan: ")
    fmt.Scan(&bil)
    fmt.Print("Faktor: ")

    for i := 1; i <= bil; i++ {
        if bil%i == 0 {
            fmt.Print(i, " ")
            if !(i == 1 || i == bil) {

```

```

        prima = false
    }
}

fmt.Println()
fmt.Print("Prima: ", prima)
}

```

### Screenshoot program



```

TERMINAL  OUTPUT  PROBLEMS  ...
Code + - [ ] [ ] ... ^ x

• PS D:\Personal\KULIAH\ALPRO_2> go run d:\Personal\KULIAH\ALPRO_2\MODUL_2\2C3\2C3.go
Bilangan: 12
Faktor: 1 2 3 4 6 12
Prima: false
• PS D:\Personal\KULIAH\ALPRO_2> go run d:\Personal\KULIAH\ALPRO_2\MODUL_2\2C3\2C3.go
Bilangan: 7
Faktor: 1 7
Prima: true
○ PS D:\Personal\KULIAH\ALPRO_2>

```

### Deskripsi program

Program Go di atas digunakan untuk menentukan faktor-faktor dari suatu bilangan dan memeriksa apakah bilangan tersebut merupakan bilangan prima. Setelah meminta pengguna untuk memasukkan sebuah bilangan (bil), program akan mencetak semua faktor dari bilangan tersebut dengan menggunakan loop dari 1 hingga bilangan yang dimasukkan. Di dalam loop, program memeriksa apakah bil dapat dibagi habis oleh i, dan jika ya, mencetak nilai i sebagai faktor. Selain itu, program memeriksa apakah faktor tersebut bukan 1 atau bilangan itu sendiri; jika ada faktor lain, maka variabel prima diubah menjadi false, menandakan bahwa bilangan tersebut bukan bilangan prima. Setelah loop selesai, program mencetak semua faktor dan menunjukkan apakah bilangan tersebut adalah bilangan prima atau tidak.