

**LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK
MODUL II
“REVIEW STRUKTUR KONTROL”**



Oleh:

MUHAMMAD RAGIEL PRASTYO

2311102183

S1IF-11-02

**S1 TEKNIK INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO**

2024

I. DASAR TEORI

2.1 Struktur Program Go

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 Komentar, bukan bagian dari kode program dan dapat ditulis dimana saja di dalam program:
- satu baris teks yang diawali dengan garis miring ganda (//) s.d. akhir baris, atau
- beberapa baris teks yang dimulai dengan pasangan karakter *'/' dan di akhiri dengan '/'*.

1) Koding, Kompilasi, dan Eksekusi Go

Koding

- Tidak berbeda dengan penulisan program sumber dalam bahasa lain, program Go harus dibuat dengan menggunakan penyuntingan teks dan disimpan dalam format teks, bukan dalam format dokumen (doc, docx, atau lainnya).
- Setiap program go disimpan dalam file teks dengan ekstensi *.go, dengan nama bebas. Sebaiknya nama file adalah nama untuk program tersebut.
- Setiap satu program lengkap Go disimpan dalam satu folder tersendiri. Nama folder merupakan nama program tersebut. Karena itu secara prinsip, satu program Go dapat dipecah dalam beberapa file dengan esktensi *.go selama disimpan dalam folder yang sama.

Kompilasi

Beberapa bahasa pemrograman dibangun untuk digunakan sebagai interpreter, sedangkan yang lain dimaksudkan untuk dikompilasi. Interpreter akan membaca setiap baris instruksi dan kemudian langsung melakukannya, hanya untuk memeriksa apakah penulisan program secara keseluruhan sudah benar atau belum. Setelah memeriksa program sumber secara keseluruhan, compiler mengubahnya menjadi program eksekutabel. Ini memastikan bahwa konsistensi penulisan, seperti penggunaan tipe data, telah dipastikan sebelum program dieksekusi. Program juga dapat dioptimalkan karena dibuat menjadi eksekutabel lebih awal.

Go berfungsi sebagai kompilator. Saat mengkompilasi dan mengeksekusi

program dalam bahasa Go, berikut adalah beberapa sesi yang biasa dilakukan:

- Panggil shell atau terminal (program/utiliti cmd.exe di windows).
- Masuk ke dalam (cd) folder program (normalnya ada di C:\Users\go\src\ atau yang sejenisnya).
- Kemudian panggil perintah `go build` atau `go build file.go` untuk mengkompilasi file.go.
- Jika gagal, akan muncul pesan eror yang sesuai, pelajari dengan baik pesan tersebut, perbaiki teks program sumber, kemudian ulangi proses build-nya.
- Jika berhasil, maka pada folder tersebut akan dibuat program dengan nama yang sama dan diakhiri dengan .exe (untuk windows).
- Panggil progra, eksekutabel tersebut dari terminal yang sama. Jangan memanggil program tersebut dengan mengklik eksekutabel tersebut dari folder karena program kalian hanya berbasis teks, bukan/belum dirancang dengan tampilan windows.

Catatan!!!

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 sumbernya saja.

2.2 Tipe Data dan Instruksi Dasar

1. Data dan Variabel

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

- Nama variabel dimulai dengan huruf dan dapat diikuti dengan sejumlah huruf, angka, atau garis bawah.
- Tipe data yang umum tersedia adalah integer, real, boolean, karakter, dan string. Lihat tabel berikut ini untuk variasi tipe data yang disediakan dalam bahasa Go.
- Nilai data yang tersimpan dalam variabel dapat diperoleh dengan menyebutkan langsung nama variabelnya contoh: menyebutkan nama

found akan mengambil nilai tersimpan dalam memori untuk variabel found, pastinya.

- Informasi alamat atau lokasi dari variabel dapat diperoleh dengan menambahkan prefiks & di depan nama variabel tersebut contoh: &found akan mendapatkan alamat memori untuk menyimpan data pada found.
- Jika variabel berisi alamat memori, prefiks * pada variabel tersebut akan memberikan nilai yang tersimpan dalam memori yang lokasinya disimpan dalam variabel tersebut contoh: *mem akan mendapatkan data di memori yang alamatnya tersimpan di mem, karenanya *(&found) akan mendapatkan data dari lokasi memori variabel found berada, alias sama saja dengan menyebutkan langsung found 8=).
- Operasi yang dapat dilakukan terhadap tipe data diatas 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 per-bit 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 boolean AND, OR, dan NOT
* &	variabel apasaja	mendapatkan data dari lokasi memori dan mendapatkan lokasi dari variabel

- Bahasa Go menganut kesesuaian tipe data yang ketat. Tipe data yang berbeda tidak boleh dicampur dalam satu ekspresi, bahkan tipe data masih yang sejenis, misalnya masih sama-sama integer (int dan int32). Untuk menyesuaikan tipe data, ada beberapa cara yang dapat dilakukan:
 - Casting, tipe (data), mengubah tipe dari data yang diberikan ke tipe yang diinginkan.
 - Memanfaatkan fungsi Sprint dan Sscan dari paket fmt.
 - Memanfaatkan fungsi-fungsi dalam paket strconv, seperti Atol, Itoa, dan ParseBool. Lihat lampiran berikut untuk contoh penggunaan.

Operasi	Hasil
<code>7878.8 % 19</code>	will be an illegal expression error
<code>int(2828.8) % 19</code>	6

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
<code>fmt.Sprintf("%v",v)</code>	any type	string	tulis output ke string
<code>fmt.Sprintf("%c",v)</code>	karakter	string	tulis karakter ke string
<code>fmt.Scanf(s,"%v",&v)</code>	string	any type	baca string ke variabel dengan tipe tertentu
<code>fmt.Scanf(s,"%c",&v)</code>	string	karakter	baca string ke variabel bertipe karakter

- Variabel harus dideklarasikan dulu sebelum digunakan. Variabel juga harus diinisialisasi dulu (diisi data) agar nilai yang tersimpan diketahui dengan jelas dan eksekusi algoritma menjadi terprediksi. Dalam bahasa Go, variabel yang tidak diinisialisasi lebih dulu otomatis diisi dengan nilai default yang ekuivalen dengan bit 0.
 - Nilai 0 untuk bilangan integer.
 - 0.0E+0 untuk bilangan real
 - false untuk boolean
 - karakter NUL (lihat tabel ASCII) untuk karakter
 - ""(string kosong, string dengan panjang 0) untuk string
 - nil untuk alamat memori

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

Contoh:

```

1 func main() {
2     var a int
3     a = 2019
4     r := 2019.0707
5     b := false
6     c := 'x'
7     s := "a string is a string"
8 }
```

2. Instruksi Dasar

Notasi instruksi dasar	Penulisan dalam bahasa Go	Keterangan
<code>v1 <- e1</code> <code>v1 <- v1 + e1</code> <code>v1 <- v1 - e1</code> <code>v1 <- v1 + 1</code> <code>v1 <- v1 - 1</code>	<code>v1 = e1</code> <code>v1 += e1 // atau v1 = v1 + e1</code> <code>v1 -= e1 // atau v1 = v1 - e1</code> <code>v1++ // atau v1 = v1 + 1</code> <code>v1-- // atau v1 = v1 - 1</code>	operasi assignment, mengisi data ke lokasi memori (variabel)
<code>input(v1, v2)</code>	<code>fmt.Scan(&v1, &v2)</code> <code>fmt.Scanln(&v1, &v2)</code> <code>fmt.Scanf("%v %v", &v1, &v2)</code>	Pembacaan data memerlukan alamat memori ke mana data akan disimpan.
<code>output(e1, e2)</code>	<code>fmt.Print(e1, e2)</code> <code>fmt.Println(e1, e2)</code> <code>fmt.Printf("%v %v\n", e1, e2)</code>	Penulisan data memerlukan nilai data yang akan ditulis.

3. Konstanta Simbolik

Konstanta dapat diberi nama untuk memudahkan mengingat maksud dan manfaat dari nilai yang diberi nama tersebut.

```
1 const PI = 3.1415926535897932384626433
2 const MARKER = "AKHIR"
```

2.3 Struktur Kontrol Perulangan

Go hanya mempunyai kata kunci `for` untuk semua jenis perulangan yang kita pelajari dalam kondisi notasi algoritma. Dua bentuk yang kita gunakan disini adalah struktur `while-loop` dan `repeat-until`. 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, seperti:

- Satu pintu keluar dari kondisi `for` dan satu lagi dari instruksi `if-break`.
 - Atau mempunyai instruksi `if-break` yang lebih dari satu.
- I. Bentuk While-Loop Bentuk While-Loop memastikan setiap kali memasuki loop, ada kondisi yang harus terpenuhi (benar/true). Ini juga berarti saat keluar dari loop, maka nilai kondisi tersebut pasti salah/false!

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

Contoh penggunaan bentuk **while-loop** untuk menghitung $y = \sqrt{x}$ adalah sebagai berikut:

	Notasi algoritma	Penulisan dalam bahasa Go
1	e <- 0.0000001	e := 0.0000001
2	x <- 2.0	x := 2.0
3	y <- 0.0	y := 0.0
4	y1 <- x	y1 := x
5	while y1-y > e or y1-y < -e do	for y1-y > e y1-y < -e {
6	y <- y1	y = y1

- II. Bentuk Repeat-Until Bentuk repeat-until di perulangan dilakukan terus menerus sampai kondisi keluar terpenuhi. Artinya selama kondisi belum terpenuhi (salah/false) maka perulangan akan terus dilakukan. Pada saat keluar dari loop maka nilai kondisi pasti benar/true!

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

Contoh penggunaan bentuk **repeat-until** untuk mencetak deret bilangan Fibonacci:

	Notasi Algoritma	Penulisan dalam bahasa Go
1	maxF <- 100	maxF := 100
2	f0 <- 0	f0 := 0
3	f1 <- 1	f1 := 1
4	f2 <- 1	f2 := 1
5	output("Bilangan pertama:", f1)	fmt.Println("Bilangan pertama:", f1)
6	repeat	for selesai:=false; !selesai; {
7	f0 <- f1	f0 = f1
8	f1 <- f2	f1 = f2
9	f2 <- f1 + f0	f2 = f1 + f0
10	output("Bilangan	fmt.Println("Bilangan berikutnya:", f1)
11	berikutnya:", f1)	selesai = f2 > maxF
12	until f2 > maxF	}

2.3 Struktur Kontrol Perulangan

Struktur kontrol perulangan adalah fitur pemrograman yang memungkinkan program menjalankan instruksi sesuai kondisi yang ditentukan berulang kali. Ada tiga kategori utama: `for` (untuk perulangan dengan jumlah iterasi yang diketahui), `while` (untuk perulangan selama kondisi terpenuhi), dan `do-while` (untuk perulangan dengan minimal satu eksekusi mirip dengan `while`). Untuk melompati iterasi, program dapat menggunakan `break` untuk menghentikan kesalahan dan `continue`. Selain itu, bahasa pemrograman kontemporer mendukung `foreach` untuk iterasi dalam pengumpulan data dan `nested loop` untuk perulangan bersarang. Dengan penggunaan yang tepat, program dapat bekerja lebih efisien dan menghindari loop berulang.

2.4 Struktur Kontrol Percabangan

Konsep yang dikenal sebagai struktur kontrol percabangan dalam pemrograman memungkinkan eksekusi kode sesuai dengan kondisi tertentu. Percabangan memungkinkan program untuk menggunakan hasil evaluasi kondisi untuk membuat keputusan yang berbeda. Struktur `if` memeriksa satu atau lebih kondisi; jika kondisi benar, blok kode yang sesuai akan dijalankan, dan jika dan kemudian dapat menangani kondisi lainnya. Namun, struktur `switch` memungkinkan pemeriksaan lebih efisien dari variabel terhadap berbagai nilai yang mungkin. Percabangan adalah komponen penting dalam pengembangan perangkat lunak karena memungkinkan pengendalian alur program dan penyesuaian dengan berbagai situasi dan input pengguna.

II. GUIDED

1. Telusuri program berikut dengan cara mengkompilasi dan mengeksekusi program. Silahkan masukkan data yang sesuai sebanyak yang diminta program. Perhatikan keluaran yang diperoleh. Coba terangkan apa sebenarnya yang dilakukan program tersebut?

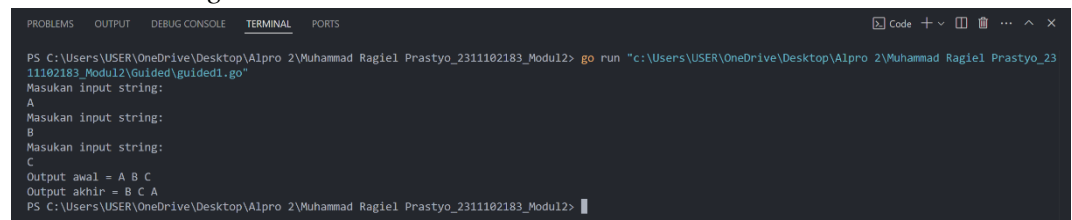
Source Code

```
package main

import "fmt"

func main() {
    var (
        satu, dua, tiga string
        temp string
    )
    fmt.Println("Masukan input string: ")
    fmt.Scanln(&satu)
    fmt.Println("Masukan input string: ")
    fmt.Scanln(&dua)
    fmt.Println("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)
}
```

Screenshot Program



```
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\guided1.go"
Masukan input string:
A
Masukan input string:
B
Masukan input string:
C
Output awal = A B C
Output akhir = B C A
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2>
```

Penjelasan:

String yang dimasukkan oleh pengguna diminta oleh program untuk ditampilkan sebelum dan sesudah penukaran. Program mengambil input setelah mendeklarasikan variabel dan menampilkan "Output awal". Kemudian, nilai diputar: satu ke dua, dua ke tiga, dan nilai awal satu ke tiga menggunakan variabel sementara temp. Akhirnya, hasil rotasi ditampilkan

dalam "Output akhir". Misalnya input A B C akan menghasilkan output akhir B C A.

2. 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). (Contoh input/output, teks bergaris bawah adalah input dari user)

Source Code

```
package main

import "fmt"

func main() {
    var year int
    fmt.Print("Masukkan tahun: ")
    fmt.Scan(&year)

    if (year%400 == 0) || (year%4 == 0 && year%100 != 0) {
        fmt.Println("Kabisat:", true)
    } else {
        fmt.Println("Kabisat:", false)
    }
}
```

Screenshot Program



```
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2\Guided\guided1.go"
Masukkan tahun: 2016
Kabisat: true
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2\Guided\guided1.go"
Masukkan tahun: 2000
Kabisat: true
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2\Guided\guided1.go"
Masukkan tahun: 2018
Kabisat: false
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2> 
```

Penjelasan:

Untuk menentukan apakah suatu tahun adalah tahun kabisat, program di atas ditulis dalam bahasa Go. Untuk menyimpan data, program mendefinisikan variabel tahun dan meminta pengguna memasukkan tahun tersebut. Setelah itu, program memeriksa apakah tahun tersebut kabisat dengan dua kondisi: jika tahun tersebut dibagi 400, atau jika tahun tersebut dibagi 4 tetapi tidak habis dibagi 100. Program menampilkan output Kabisat "true" jika salah satu kondisi terpenuhi. Jika tidak, program menampilkan

output Kabisat “false”. Oleh karena itu, program ini menunjukkan tahun kabisat.

3. Buat program Bola yang menerima input jari-jari suatu bola (bilangan bulat). Tampilkan volume dan luas kulit bola. volume bola = $\frac{4}{3} \pi r^3$ dan luas bola = $4 \pi r^2$ ($\pi = 3.14$) (contoh input/output, teks bergaris bawah adalah input dari user)

Source Code

```
package main

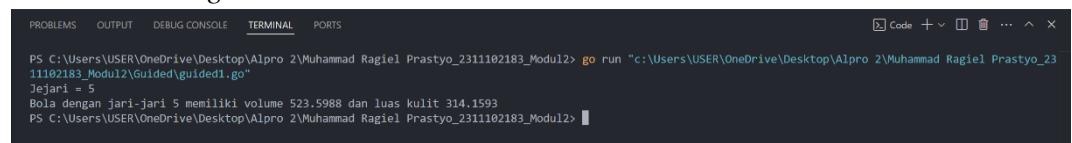
import (
    "fmt"
    "math"
)

func main() {
    var jariJari int
    fmt.Print("Jeari = ")
    fmt.Scan(&jariJari)

    volume := (4.0 / 3.0) * math.Pi *
float64(jariJari*jariJari*jariJari)
    luas := 4 * math.Pi * float64(jariJari*jariJari)

    fmt.Printf("Bola dengan jari-jari %d memiliki volume %.4f
dan luas kulit %.4f\n", jariJari, volume, luas)
}
```

Screenshot Program



```
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2\Guided\guided1.go"
Jeari = 5
Bola dengan jari-jari 5 memiliki volume 523.5988 dan luas kulit 314.1593
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi Prastyo_231102183_Modul2>
```

Penjelasan:

Program ini ditulis dalam bahasa Go dan menggunakan jari-jari pengguna untuk menghitung volume dan luas kulit bola. Setelah menerima input jari-jari, program menghitung volume dengan rumus $\frac{4}{3} \pi r^3$ dan luas dengan rumus $4 \pi r^2$. Hasilnya ditampilkan dalam format yang jelas, menunjukkan jari-jari, volume, dan luas kulit bola.

III. UNGUIDED

SOAL LATIHAN MODUL 2B

1. Buatlah sebuah program yang menerima input berupa warna dari ke 4 gelas reaksi sebanyak 5 kali percobaan. Kemudian program akan menampilkan true apabila urutan warna sesuai dengan informasi yang diberikan pada paragraf sebelumnya, dan false untuk urutan warna lainnya.

Source Code

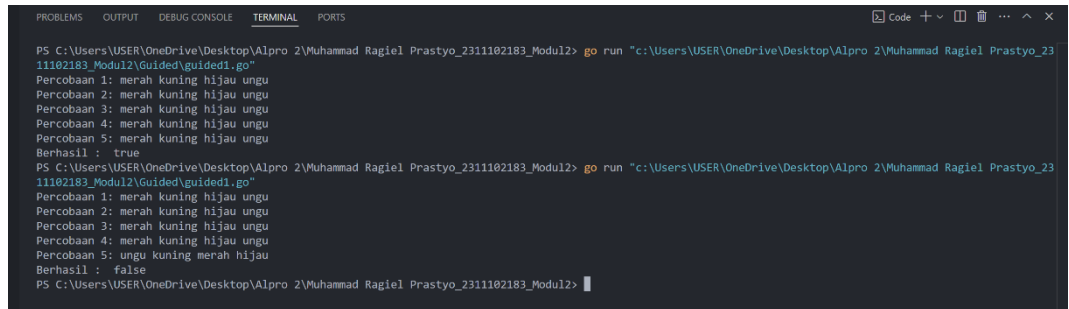
```
// Muhammad Ragi Prastyo
// IF-11-02
// 2311102183
package main

import (
    "fmt"
    "strings"
)

func main() {
    var ujiKimia = [5][4]string{ }
    var warna = [4]string{ "merah", "kuning", "hijau", "ungu" }
    var beda bool = true
    for i := 0; i < 5; i++ {
        fmt.Print("Percobaan ", i+1, ": ")
        fmt.Scan(&ujiKimia[i][0], &ujiKimia[i][1], &ujiKimia[i][2],
        &ujiKimia[i][3])
    }

    for i := 0; i < 5; i++ {
        for j := 0; j < 4; j++ {
            beda = strings.ToLower(ujiKimia[i][j]) == warna[j]
            if !beda {
                i = 5
                break
            }
        }
    }
    fmt.Println("Berhasil : ", beda)
}
```

Screenshot Program



```
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Guided\guided1.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 C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Guided\guided1.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: ungu kuning merah hijau
Berhasil : false
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2>
```

Penjelan:

Program ini dibuat dalam bahasa Go untuk menilai hasil eksperimen kimia berdasarkan warna yang dimasukkan pengguna. Setelah menentukan array yang akan menyimpan hasil eksperimen dan warna yang diharapkan (merah, kuning, hijau, dan ungu), program meminta pengguna memasukkan data untuk lima eksperimen. Selanjutnya, program memeriksa warna yang dimasukkan dan yang diharapkan. Jika semua warna cocok, variabel perbedaan tetap “benar”. Jika tidak, nilainya menjadi “salah”. Hasil akhir menunjukkan apakah percobaan tersebut berhasil atau tidak.

2. Buatlah sebuah program yang menerima input sebuah bilangan bulat positif (dan tidak nol) N, kemudian program akan meminta input berupa nama bunga secara berulang sebanyak N kali dan nama tersebut disimpan ke dalam pita.

Source Code

```
// Muhammad Ragiel Prastyo
// IF-11-02
// 2311102183
package main

import (
    "bufio"
    "fmt"
    "os"
    "strings"
)

func main() {
```

```

var pita string
var TotalBunga int

scanner := bufio.NewScanner(os.Stdin)

fmt.Print("Masukkan nama bunga dan ketik 'SELESAI' untuk
mengakhiri. :\n")

for {
    TotalBunga++
    fmt.Printf("Bunga %d: ", TotalBunga)

    scanner.Scan()
    input := scanner.Text()

    if strings.ToUpper(input) == "SELESAI" {
        TotalBunga--
        break
    }

    if pita == "" {
        pita = input
    } else {
        pita += " - " + input
    }
}

fmt.Println("Pita:", pita)
fmt.Println("Total bunga:", TotalBunga )
}

```

Screenshot Program

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Guided\guided1.go"
Masukkan nama bunga dan ketik 'SELESAI' untuk mengakhiri. :
Bunga 1: kertas
Bunga 2: mawar
Bunga 3: tulip
Bunga 4: SELESAI
Pita: kertas - mawar - tulip
Total bunga: 3
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Guided\guided1.go"
Masukkan nama bunga dan ketik 'SELESAI' untuk mengakhiri. :
Bunga 1: SELESAI
Pita:
Total bunga: 0
PS C:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2>

```

Penjelasan:

Program ini dibuat dalam Go untuk mengumpulkan nama-nama bunga dari pengguna hingga mereka mengetik "SELESAI". Variabel pita menyimpan nama-nama bunga, dan variabel "TotalBunga" menghitung jumlah bunga

total. Setelah selesai, program menampilkan daftar bunga dan jumlah totalnya.

3. Buatlah program Pak Andi yang menerima input dua buah bilangan real positif yang menyatakan berat total masing-masing isi kantong terpal. Program akan terus meminta input bilangan tersebut hingga salah satu kantong terpal berisi 9 kg atau lebih.

Source Code

```
// Muhammad Ragi Prastyo
// IF-11-02
// 2311102183
package main

import "fmt"

func main() {
    for {
        var berat1, berat2 float64

        fmt.Print("Masukkan berat belanjaan di kedua kantong: ")
        _, err := fmt.Scan(&berat1, &berat2)
        if err != nil {
            fmt.Println("Input tidak valid.")
            return
        }

        if berat1+berat2 > 150 || berat1 < 0 || berat2 < 0 {
            fmt.Println("Proses selesai.")
            break
        }

        selisih := berat1 - berat2
        if selisih < 0 {
            selisih = -selisih
        }
        akanOleng := selisih >= 9

        fmt.Printf("Sepeda motor pak Andi akan oleng:
        %t\n", akanOleng)
    }
}
```

Screenshot Program

```
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragi1 Prastyo_2311102183_Modul2\Unguided\unguided2b-3.go"
Masukkan berat belanjaan di kedua kantong: 5 10
Sepeda motor pak Andi akan oleng: false
Masukkan berat belanjaan di kedua kantong: 55.6 70.2
Sepeda motor pak Andi akan oleng: true
Masukkan berat belanjaan di kedua kantong: 72.3 66.9
Sepeda motor pak Andi akan oleng: false
Masukkan berat belanjaan di kedua kantong: 59.5 98.7
Proses selesai.
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> █
```

Penjelasan:

Program ini ditulis dalam Go dan dimaksudkan untuk mengetahui apakah sepeda motor Pak Andi akan oleng atau tidak. Program berhenti jika berat kedua kantong dimasukkan oleh pengguna. Jika totalnya melebihi 150 kg atau salah satu beratnya kurang dari 0, program berhenti. Selanjutnya, program menghitung perbedaan berat. Jika lebih dari 9 kg terjadi, sepeda motor dianggap akan oleng.

4. Buatlah sebuah program yang menerima input sebuah bilangan sebagai K, kemudian menghitung dan menampilkan nilai $f(K)$ sesuai persamaan di modul.
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

```
// Muhammad Ragi1 Prastyo
// IF-11-02
// 2311102183
package main

import (
    "fmt"
    "math"
)

func main() {
    var k float64

    fmt.Print("Masukkan nilai K: ")
    fmt.Scan(&k)

    fk := math.Pow(4*k+2, 2) / ((4*k + 1) * (4*k + 3))
```



```
    fmt.Printf("Nilai f(k) = %.10f\n",fk)
}
```

Source Code Setelah Modifikasi

```
// Muhammad Ragiel Prastyo
// IF-11-02
// 2311102183
package main

import (
    "fmt"
    "math"
)

func main() {
    var jumlah int

    fmt.Print("Masukkan jumlah nilai K yang ingin dihitung: ")
    fmt.Scan(&jumlah)

    for i := 0; i < jumlah; i++ {
        var k int
        fmt.Printf("Masukkan nilai K ke-%d : ", i+1)
        fmt.Scan(&k)

        // Menghitung  $\sqrt{2}$  (dalam konteks ini, nilai K tidak digunakan)
        sqrt2 := math.Sqrt(2)

        fmt.Printf("Nilai akar 2 dari %d adalah %.10f\n", k, sqrt2)
    }
}
```

Screenshot Program

```
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\unguided2b-4.go"
Masukkan nilai K: 100
Nilai f(k) = 1.0000061880
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> █
```

Screenshot Program Setelah Modifikasi

```
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\unguided2b-4modif.go"
Masukkan jumlah nilai K yang ingin dihitung: 3
Masukkan nilai K ke-1 : 10
Nilai akar 2 dari 10 adalah 1.4142135624
Masukkan nilai K ke-2 : 100
Nilai akar 2 dari 100 adalah 1.4142135624
Masukkan nilai K ke-3 : 1000
Nilai akar 2 dari 1000 adalah 1.4142135624
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> █
```

Penjelasan:

Program Go ini meminta pengguna memasukkan nilai K dan kemudian menggunakan rumus matematika tertentu untuk menghitung nilai $f(k)$. Program ini berfungsi untuk menilai fungsi matematika berdasarkan input. Hasil perhitungan ditampilkan dengan presisi 10 desimal.

Setelah dimodifikasi, Program Go ini meminta pengguna untuk memasukkan nilai K dan menghitung nilai perkiraan akar dua dengan menggunakan rumus tertentu. Kemudian, program melakukan validasi untuk memastikan K adalah angka positif. Selama loop, program menghitung nilai dengan menggunakan rumus yang melibatkan penjumlahan pecahan hingga K dan menampilkan hasilnya dengan presisi 10 desimal. Sampai pengguna memilih untuk menghentikannya, program ini akan terus meminta input.

SOAL LATIHAN MODUL 2C

1. 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

```
// Muhammad Ragiell Prastyo
// IF-11-02
// 2311102183
package main

import (
    "fmt"
)

func main() {
    var berat int
```

```

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

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

    original := kg * 10000
    tambahan := 0

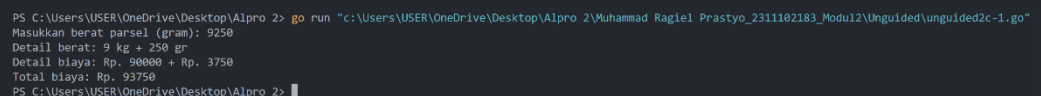
    if kg >= 10 {
        tambahan = 0
    } else {
        if grams < 500 {
            tambahan = grams * 15
        } else {
            tambahan = grams * 5
        }
    }

    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)
}

```

Screenshot Program



```

PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\unguided2c-1.go"
Masukkan berat parcel (gram): 9250
Detail berat: 9 kg + 250 gr
Detail biaya: Rp. 90000 + Rp. 3750
Total biaya: Rp. 93750
PS C:\Users\USER\OneDrive\Desktop\Alpro 2>

```

Penjelasan:

Program ini menghitung biaya pengiriman parcel berdasarkan berat yang dimasukkan dalam gram. Biaya dasar adalah Rp. 10.000 per kilogram, dan biaya tambahan adalah Rp. 15 per gram jika sisa berat kurang dari 500 gram dan Rp. 5 per gram jika lebih. Total biaya ditampilkan kepada pengguna sebagai jumlah dari biaya dasar dan tambahan.

2. Program berikut menerima input sebuah bilangan riil yang menyatakan NAM. Program menghitung NMK dan menampilkannya.

Source Code

```
// Muhammad Ragi Prastyo
// IF-11-02
// 2311102183
package main

import "fmt"

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

    if nam > 80 {
        nm = "A"
    }
    if nam > 72.5 {
        nm = "AB"
    }
    if nam > 65 {
        nm = "B"
    }
    if nam > 57.5 {
        nm = "BC"
    }
    if nam > 50 {
        nm = "C"
    }
    if nam > 40 {
        nm = "D"
    } else if nam <= 40 {
        nm = "E"
    }

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

Jawablah pertanyaan-pertanyaan berikut:

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

Jawab: Seharusnya akan mendapatkan A sesuai sama spesifikasi soal, tetapi dari program diatas tidak dapat di run atau ERROR.

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

Jawab: Ada beberapa kesalahan dalam program diatas. Pertama, program mencoba menetapkan nilai huruf ke variabel nam bertipe float64, yang menyebabkan kesalahan kompilasi karena variabel nmk untuk menyimpan nilai huruf tidak digunakan dengan benar. Selain itu, untuk menghindari pengecekan yang tidak perlu setelah satu kondisi terpenuhi, struktur if seharusnya menggunakan else if. Jika nmk tidak pernah diperbarui di akhir, output akan kosong. Untuk mengatasi hal ini, program harus meminta input nilai akhir, menghitung nilai huruf dengan menggunakan if-else, dan kemudian mencetak nilai huruf tersebut. Saat nilai akhir dimasukkan, program akan berfungsi dengan baik dan memberikan output yang tepat dengan perbaikan ini.

- c. Perbaiki program tersebut! Ujilah dengan masukan: 93.5; 70.6; dan 49,5. Seharusnya keluaran yang diperoleh adalah 'A', 'B', dan 'D'.

Jawab:

Source Code Setelah Perbaikan

```
// Muhammad Ragi Prastyo
// IF-11-02
// 2311102183
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"
    }
```

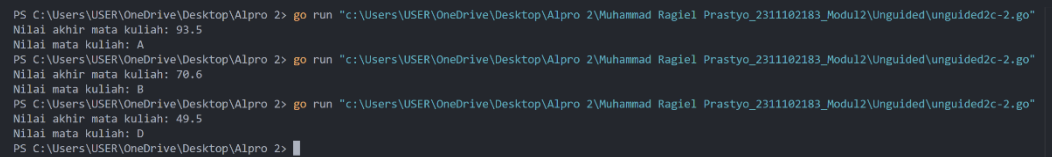
```

    } else {
        nmk = "E"
    }

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

```

Screenshot Program



```

PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\unguided2c-2.go"
Nilai akhir mata kuliah: 93.5
Nilai mata kuliah: A
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\unguided2c-2.go"
Nilai akhir mata kuliah: 70.6
Nilai mata kuliah: B
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\unguided2c-2.go"
Nilai akhir mata kuliah: 49.5
Nilai mata kuliah: D
PS C:\Users\USER\OneDrive\Desktop\Alpro 2>

```

Penjelasan:

Program ini mengubah nilai akhir mata kuliah menjadi nilai huruf. Program meminta pengguna untuk mengimpor nilai akhir yang disimpan dalam variabel `nam`. Program menemukan huruf A, AB, B, BC, C, D, atau E berdasarkan rentang nilai dan mencetak hasilnya menggunakan struktur `if-else`. Program ini sesuai dengan input pengguna dengan alur yang sederhana.

3. Buatlah program yang menerima input sebuah bilangan bulat `b` dan `b > 1`. Program harus dapat mencari dan menampilkan semua factor dari bilangan tersebut!

Lanjutkan program sebelumnya. Setelah menerima masukan sebuah bilangan bulat `b > 0`. Program tersebut mencari dan menampilkan semua faktor bilangan tersebut. Kemudian, program menentukan apakah `b` merupakan bilangan prima.

Source Code

```

// Muhammad Ragiel Prastyo
// IF-11-02
// 2311102183

package main

import "fmt"

func main() {
    var bil int

    fmt.Print("Bilangan: ")
}

```

```

    fmt.Scan(&bil)

    if bil <= 1 {
        fmt.Println("Error: Bilangan harus lebih besar dari 1!")
        return
    }

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

```

Source Code Lanjutan

```

// Muhammad Ragiel Prastyo
// IF-11-02
// 2311102183
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)
}

```

Screenshot Program

```
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\unguided2c-3.go"
Bilangan: 12
Faktor: 1 2 3 4 6 12
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\unguided2c-3.go"
Bilangan: 7
Faktor: 1 7
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> █
```

Screenshot Program Lanjutan

```
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\tempCodeRunnerFile.e.go"
Bilangan: 12
Faktor: 1 2 3 4 6 12
Prima: false
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> go run "c:\Users\USER\OneDrive\Desktop\Alpro 2\Muhammad Ragiel Prastyo_2311102183_Modul2\Unguided\tempCodeRunnerFile.e.go"
Bilangan: 7
Faktor: 1 7
Prima: true
PS C:\Users\USER\OneDrive\Desktop\Alpro 2> █
```

Penjelasan:

Kedua source code dalam bahasa Go ini berfungsi untuk menghitung faktor sebuah angka dan menentukan apakah angka tersebut adalah bilangan prima. Program pertama meminta pengguna memasukkan angka dan mencetak semua faktor jika angka tersebut lebih besar dari 1. Program kedua melakukan hal yang sama, tetapi juga memeriksa apakah angka hanya memiliki dua faktor, yaitu satu dan bilangan itu sendiri, untuk mengetahui apakah angka tersebut adalah prima. Kedua memberikan informasi tentang komponen dan karakteristik keprimaan bilangan.