**LAPORAN PRAKTIKUM  
PEMROGRAMAN BERORIENTASI OBJEK**

**MODUL 2**

**REVIEW STUKTUR KONTROL**



Oleh:

Jonathan Kantohe

2311102281

S1IF-11-02

**S1 TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

**2024**

1. **DASAR TEORI**

* **Struktur Program GO**

Dalam kerangka program yang ditulis dalam Bahasa pemograman 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 di mana saja di dalam program:

* Satu baris teks yang diawali dengan garis miring danda (“//”) s.d. akhir baris, atau
* Beberapa baris teks yang dimulai dengan pasangan karakter ‘/’ dan diakhiri dengan ‘/’.

**1) Koding, Kompilasi, dan Eksekusi Go**

**Koding**

* Tidak berbeda dengan penulisan program sumber dalam Bahasa lain, program go harus dibuat
* menggunakan penyuting teks dan disimpan dalam format teks, bukan dalam format dokumen.
* Setiap program go disimpan dalam file teks dengan ekstensi “nama\_file.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

mana program tersebut. Karena itu secara prinsip, satu program Go dapat dipecah dalam

beberapa file dengan ekstensi “nama\_file.go” selama disimpan dalam folder yang sama.

**Komplikasi**

Beberapa Bahasa pemograman dirancang untuk diimplementasikan sebagai interpreter dan lainnya sebagai

kompilator. Interpreter akan membaca setiap baris instruksi dan kemudian langsung mengeksekusinya,

dengan hanya sedikit pemeriksaan apakah penulisan keseluruhan program sudah benar atau belum

Kompilator akan memeriksa keseluruhan program sumber dan kemuadian mengubahnya menjadi program

eksekutabel, sehingga konsisten penulisan (seperti penggunaan tipe data) sudah diperiksa sebelum eksekusi.

Selain itu karena program dibuat menjadi eksekutabel lebih dahulu, proses optimasi dapat dilakukan

sehingga program menjadi sangat efisien.

Go diimplementasikan sebagai komplikator. Beritkut adalah contoh sesi yang biasa dilakukan saat

mengkomplikasi dan mengeksekusi program dalam Bahasa Go:

* + Panggil shell atau terminal (programlutiliti cmd.exe di Windows)
  + Masuk ke dalam (cd) folder program
  + Kemudian panggil perintah GO build atau GO build file.go untuk mengkompilasi file.go
  + Jika gagal, ahan 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 ahan dibuat program dengan nama yang sama dan diakhiri
  + dengan .exe (untuk Windows)
  + Panggil program 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 bulld: 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.

**2). Latihan**

1. Selidiki bahasa-bahasa pemrograman berikut, apakah termasuk diinterpretasi, dikompilasi, dikompilasi

(ke instruksi perantara) kemudian diinterpretasi:

* Pascal
* C dan C++
* Java
* Python

2. Unduh kompilator GO di perangkat yang anda gunakan. Kemudian salin contoh program di atas ke dalam

folder C:\user\userid\go\hello\hello.go yaitu buat folder go dalam direktori home anda, kemudian buat

subfolder hello dan taruh file hello.go di dalamnya. Hidupkan terminal (cmd.exe), dan panggil go build ke

dalam folder tersebut. Periksa apakah hello.exe muncul di folder tersebut, jika iya ciba eksekui program

tersebut, juga melalui terminal cmd.exe tersebut.

**2. Tipe Data dan Instruksi Dasar**

**1) Data dan Variabel**

Variable merupakan nama dari lokasi di memori

* Nama variable
* Tipe data umum
* Informasi alamat atau lokasi
* Operasi yang dapat dilakukan pada tipe data di atas
* Bahasa Go memiliki kesesuaian tipe data yang ketat
* Variabel harus dideklarasikan dahulu sebelum digunakan

1. **GUIDED**

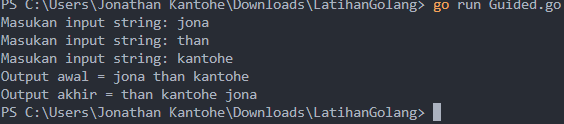
* **GUIDED 1**

Telusuri program berikut dengan cara mengkompilasi dan mengeksekusi program. Silakan masukan 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.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)  } |

**Out put :**

****

**Penjelasan :**

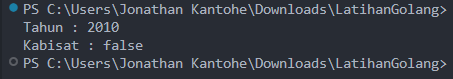
Program ini meninta pengguna memasukan tiga buah string, kemudian menukar posisi ketiga string tersebut, dan akhinya menampilkan hasil penukasaran, mendeklarasikan beberapa variable string, menggabungkan string untuk menampilkan output awal, dan menukar kemudian menukar nilai variable menggunakan bariabel sementara.

* **GUIDED 2**

**Source code :**

|  |
| --- |
| package main  import "fmt"  func main() {  var tahun int  fmt.Print("Tahun : ")  fmt.Scan(&tahun)  fmt.Printf("Kabisat : %t \n", (tahun%4 == 0 && (tahun%100 != 0 || tahun%400 == 0)))  } |

**Out put :**

****

**Penjelaskan :**

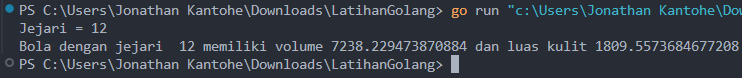
Program ini dirancang untuk menentukan apakah suatu tahun merupakan tahun kabisat atau bukan, program ini akan memeriksa apakah tahun tersebut habis dibagi 4, jika tidak habis di bagi 100, kecuali jika juga habis dibagi 400.

* **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)  } |

**Out put :**



**Penjelasan :**

Program ini menghitung volume dan luas permukaan sebuah bola, dengan memasukan nilai jari-jari bola menggunakan rumus menghitung volume dan luas permukaan bola berdasarkan jari-jari yang di in put.

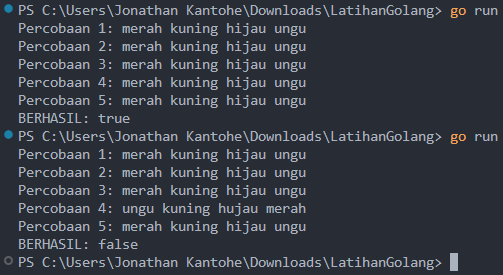
1. **UNGUIDED**

* **UNGUIDED 2B 1**

**Source Code :**

|  |
| --- |
| package main  import (  "bufio"  "fmt"  "os"  "strings"  )  func main() {  correctOrder := []string{"merah", "kuning", "hijau", "ungu"}  reader := bufio.NewReader(os.Stdin)  var hadError bool  for i := 1; i <= 5; i++ {  fmt.Printf("Percobaan %d: ", i)  input, \_ := reader.ReadString('\n')  input = strings.TrimSpace(input)  colors := strings.Split(input, " ")  for j := 0; j < 4; j++ {  if colors[j] != correctOrder[j] {  hadError = true  break  }  }  }  if !hadError {  fmt.Println("BERHASIL: true")  } else {  fmt.Println("BERHASIL: false")  }  } |

**Out put :**

****

**Penjelasan :**

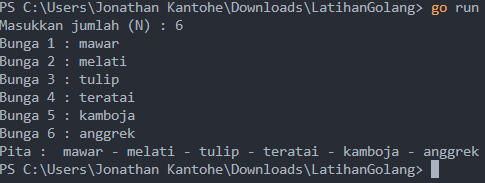
Program ini memeriksa apakah urutan warna yang di masukan sudah sesuai atau belum dengan ururtan yang telag di tentukan sebelumnya, dengan memasukan urutan warna sebanyak 5 kali, setiap inputan akan di bandingkan dengan urutan warna yang benar. Jika ada satu urutan warna yang tidak benar maka program akan di anggap salah.

* **UNGUIDED 2B 2**

**Source Code :**

|  |
| --- |
| package main  import "fmt"  func main() {  var pita string  var bunga string  var banyak int  fmt.Print("Masukkan jumlah (N) : ")  fmt.Scan(&banyak)  for i := 0; i < banyak; i++ {  fmt.Print("Bunga ", i+1, " : ")  fmt.Scan(&bunga)  pita += bunga + " - "  }  fmt.Println("Pita : ", pita)  } |

**Out put :**

****

**Penjelasan :**

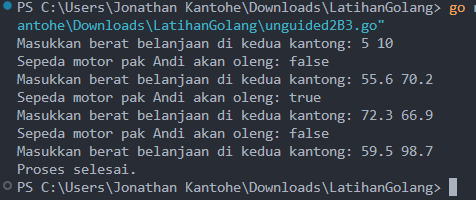
Program ini dibuat untuk merangkai bunga secara interaktif, dengan memasukan kumlah bunga yang ingin diinginkan, selajutnya masukan nama-nama bunga yang di inginkan kemudian akan digabungan menjadu satu string dan ditampilkan sebagai rangkaian bunga.

* **UNGUIDED 2B 3**

**Source Code :**

|  |
| --- |
| 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)  }  } |

**Out put :**

****

**Penjelasan :**

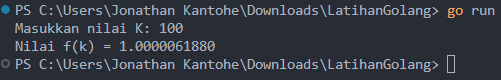
Program ini untuk mensimulasikan sebuah scenario dimana seseorang membawa dua kantong belanja menggunakan sepedah motor, dengan memasukan berat kedua pantong belanjaan, program akan menghitung hasil total berat belanjaan dan memeriksa apakah total berat melelibihi batas yang di tentukan. Selain itu program ini juga menhitung selisih berat antar kedua kantong untuk menentukan apakah sepedah motor akan oleng atau tidak.

* **UNGUIDED 2B 4**

**Source Code :**

|  |
| --- |
| 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)  } |

**Out put :**

****

**Penjelasan :**

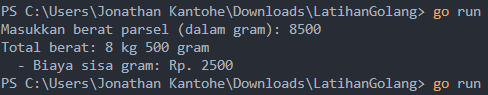
Program ini dapat menerima masukan berupa bilangan decimal (float64), dapat melakukan penghitungan matematis berdasarkan rumus yang melibatkan pangkat dan perkalian, lalu menampilkan hasil akhir perhitungan.

* **UNGUIDED 2C 1**

**Source Code :**

|  |
| --- |
| package main  import "fmt"  func hitungBiaya(beratGram int) (int, int, int) {    beratKg := beratGram / 1000  sisaGram := beratGram % 1000  biayaKg := beratKg \* 10000  var biayaSisa int  if sisaGram >= 500 {  biayaSisa = sisaGram \* 5  } else if sisaGram > 0 && beratKg <= 10 {  biayaSisa = sisaGram \* 15  } else {  biayaSisa = 0  }  return biayaKg, biayaSisa, beratKg  }  func main() {  var beratGram int  fmt.Print("Masukkan berat parsel (dalam gram): ")  fmt.Scanln(&beratGram)    biayaKg, biayaSisa, beratKg := hitungBiaya(beratGram)  fmt.Printf("Total berat: %d kg %d gram\n", beratKg, beratGram%1000)  fmt.Printf("Biaya pengiriman: Rp. %d\n", biayaKg+biayaSisa)  fmt.Printf(" - Biaya per kg: Rp. %d\n", biayaKg)  fmt.Printf(" - Biaya sisa gram: Rp. %d\n", biayaSisa)  } |

**Out put :**

****

**Penjelasan :**

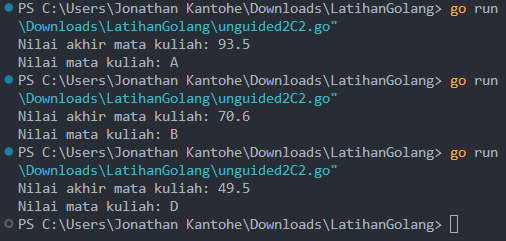
Program ini dapat menghitung biaya pengirim suatu paket berdasarkan beratnya dalam gram. Selanjutnya memasukan berat paket, lalu menghitung biaya pengiriman berdasarkan aturan tertentu yang telah di definisikan dalam fungsi *hitungBiaya,* memperhitungkan berat paket dalam kilogram dan sisa gram, serta menerapkan tariff yang berbeda-beda berdasrkan berat dan sisa gram terseut.

* **UNGUIDED 2C 2**

**Source Code :**

|  |
| --- |
| package main  import "fmt"  func main() {  var nilaiAkhir float64  var nilaiHuruf string  fmt.Print("Nilai akhir mata kuliah: ")  fmt.Scanln(&nilaiAkhir)  switch {  case nilaiAkhir > 80:  nilaiHuruf = "A"  case nilaiAkhir > 72.5:  nilaiHuruf = "AB"  case nilaiAkhir > 65:  nilaiHuruf = "B"  case nilaiAkhir > 57.5:  nilaiHuruf = "BC"  case nilaiAkhir > 50:  nilaiHuruf = "C"  case nilaiAkhir > 40:  nilaiHuruf = "D"  default:  nilaiHuruf = "E"  }  fmt.Println("Nilai mata kuliah:", nilaiHuruf)  } |

**Out put :**



**Penjelasan :**

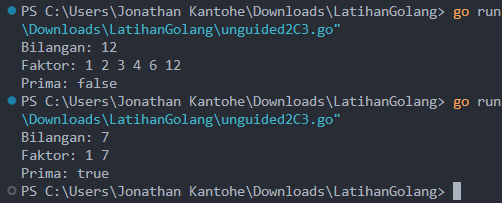
Program ini berfungsi untuk mengubah nilai numerik (nilai akhir mata kuliah) menjadi nilai huruf (A, B, C, D dan E). Dengan memasukan nilai akhir, struktur switch case untuk membandingkan nilai tersebut dengan rentan nilai yang telah ditetapkan.

* **UNGUIDED 2C 3**

**Source Code :**

|  |
| --- |
| package main  import (  "fmt"  )  func main() {  var b int  fmt.Print("Bilangan: ")  fmt.Scanln(&b)  if b <= 1 {  fmt.Println("Silakan masukkan bilangan bulat b > 1.")  return  }  fmt.Print("Faktor: ")  isPrima := true  for i := 1; i <= b; i++ {  if b%i == 0 {  fmt.Print(i, " ")  if i > 1 && i < b {  isPrima = false  }  }  }  fmt.Println()  fmt.Printf("Prima: %t\n", isPrima)  } |

**Out put :**

****

**Penjelasan :**

Program ini dirancang untuk memeriksa apakah sebauh bilangan bulat adalah bilangan prima atau bukan. Dengan cara measukan sebuah bilangan bulat, selanjutnya program akan memeriksa apakah bilangan tersebut hanya dapat di bagi habis 1 dan bilangan itu sendiri