**LAPORAN PRAKTIKUM**

**ALGORITMA DAN PEMROGRAMAN 2**

**MODUL II**

**REVIEW STRUKTUR KONTROL**

**Disusun Oeh :**

**WILDAN MAULANA ZIDAN**

**2311102162**

**IF 11 02**

**S1 TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

1. **DASAR TEORI**

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 di mana 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 diakhiri dengan '/'.

1. Tipe data dan instruksi dasar

Tipe data yang digunakan dalam go diantaranya ;

|  |  |
| --- | --- |
| Notasi Tipe Data | Dalam Go |
| Integer | Int  Int8  Int32//rune  Int64  Uint  Uint8//byte  Uint32  Uint64 |
| Boolean | Bool |
| String | String |
| Real | Float32  Float64 |
| Char | Byte  Rune |

1. Operator

Operator aritmatika dan logika dalam golang tidak banyak berbeda seperti pada c++ seperti (+) untuk pertambahan, (-) untuk pengurangan, (\*) untuk perkalian, (/) untuk pembagian. Dan untuk operator logika (&&) untuk AND, (||) untuk OR, (!) untuk not

1. Konstanta

Konstanta dapat diberi nama untuk memudahkan mengingat maksud dan manfaat dari nilai yang diberi nama tersebut. Seperti Pl untuk merepresentasikan konstanta π.

1. Perulangan

Go mempunyai notasi for untuk perulangan nya dan beberapa jenis penggunaan seperti :

|  |
| --- |
| for inisialisasi; kondisi; update {  // for-loop ala C  // ke-3 bagian opsional, tetapi ";" tetap harus ada  } |
| for kondisi {  // ulangi kode di sini selama kondisi terpenuhi  // sama seperti "for; kondisi; {"  } |
| for {  // tanpa kondisi, berarti loop tanpa henti (perlu if-break)  } |
| for ndx, var := range slice/array {  // iterator mengunjungi seluruh isi slice/array  // pada setiap iterasi ndx diset indeks dan var diisi nilainya  } |

1. Percabangan

Go mempunyai notasi if else untuk percabangan nya tidak jauh berbeda seperti :

* Bentuk if else

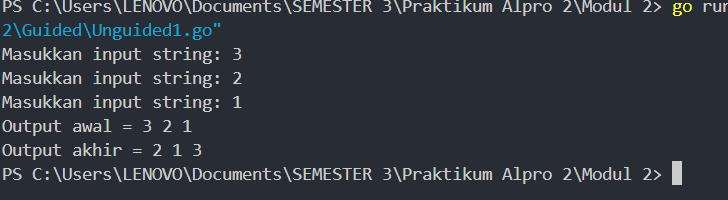
|  |
| --- |
| if kondisi {  kode untuk kondisi true  } |
| if kondisi {  kode untuk kondisi true  } else {  kode untuk kondisi false  } |
| if kondisi\_1 {  kode untuk kondisi\_1 true  } else if kondisi\_2 {  .. kode untuk kondisi\_2 true  .. dst. dst.  } else {  ..  kode jika semua kondisi  .. di atas false ormatics lab  } |

* Bentuk Switch Case

|  |
| --- |
| 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  } |
| 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 } |

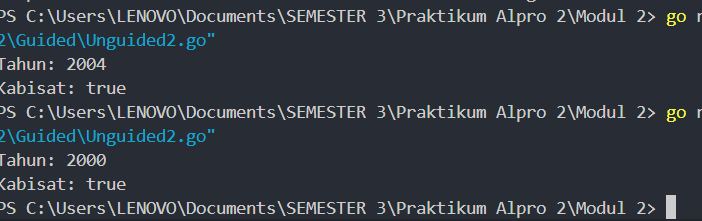
1. **GUIDED**
2. Telusuri program berikut dengan cara mengkompilasi dan mengeksekusi program. Silakan masukan data yang sesuai sebanyak yang diminta program. Perhatikan keluaran yang diperoleh. Coba terangkan apa sebenarnya yang dilakukan program tersebut?

|  |
| --- |
| package main  import "fmt"  func main() {  var satu, dua, tiga string  var temp string  fmt.Print("Masukkan input string: ")  fmt.Scanln(&satu)  fmt.Print("Masukkan input string: ")  fmt.Scanln(&dua)  fmt.Print("Masukkan 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)  } |

Program di atas meminta pengguna memasukkan tiga string, menampilkan urutannya, lalu melakukan rotasi nilai antar variabel menggunakan variabel sementara. Hasilnya, urutan string berubah dimana string pertama menjadi terakhir, kedua menjadi pertama, dan ketiga menjadi kedua. Program menampilkan hasil rotasi ini sebagai output akhir, mendemonstrasikan teknik sederhana pertukaran nilai variabel dalam Go.

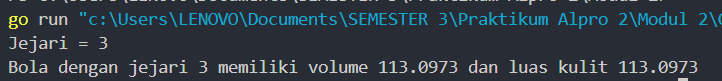
1. Tahun kabisat adalah tahun yang habis dibagi 400 atau habis dibagi 4 tetapi tidak habis dibagi 100. Buatlah sebuah program yang menerima input sebuah bilangan bulat dan memeriksa apakah bilangan tersebut merupakan tahun kabisat (true) atau bukan (false).

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

   
  
  
  
   
  
Program di atas dirancang untuk menentukan apakah suatu tahun adalah tahun kabisat atau bukan. Program meminta pengguna memasukkan angka tahun, kemudian menggunakan rumus matematika untuk menghitung apakah tahun tersebut kabisat. Sebuah tahun dianggap kabisat jika habis dibagi 4 (tahun%4 == 0) dan tidak habis dibagi 100 (tahun%100 != 0), atau jika habis dibagi 400 (tahun%400 == 0). Hasil perhitungan ditampilkan dalam format boolean (true/false), dimana true menandakan tahun kabisat dan false bukan tahun kabisat.

1. Buat program Bola yang menerima input jari-jari suatu bola (bilangan bulat). Tampilkan Volume dan Luas kulit bola. volumebola = ! πr" dan luasbola = 4πr (π = 3.1415926535).

|  |
| --- |
| package main  import (  "fmt"  "math"  )  func main() {  var jari, volume, luasKulit float64  fmt.Print("Jejari = ")  fmt.Scan(&jari)  volume = math.Pi \* (4.0 / 3.0) \* math.Pow(jari, 3)  luasKulit = 4 \* math.Pi \* math.Pow(jari, 2)  fmt.Printf("Bola dengan jejari %v memiliki volume %.4f dan luas kulit %.4f \n", jari, volume, luasKulit)  } |

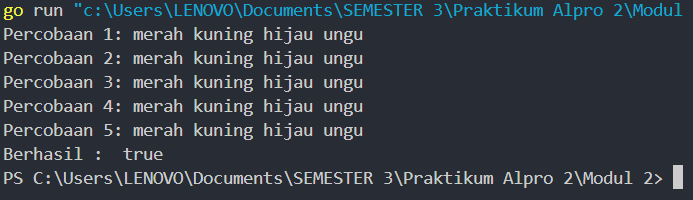
Program di atas menghitung volume dan luas permukaan (luas kulit) sebuah bola berdasarkan jejari yang dimasukkan pengguna. Program menggunakan package math untuk mengakses konstanta Pi dan fungsi Pow untuk perhitungan pangkat. Setelah menerima input jejari, program menghitung volume bola menggunakan rumus (4/3) × π × r³ dan luas permukaan dengan rumus 4 × π × r².

1. **UNGUIDED**

Modul 2B

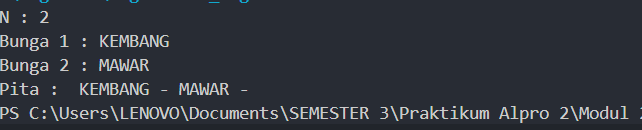
1. Buatlah sebuah program yang menerima input berupa warna dari ke 4 gelas reaksi sebanyak 5 kali percobaan. Kemudian program akan menampilkan true apabila urutan warna sesuai dengan informasi yang diberikan pada paragraf sebelumnya, dan false untuk urutan warna lainnya.

|  |
| --- |
| 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)  } |

Program di atas dirancang untuk melakukan verifikasi hasil uji kimia berdasarkan warna. Program menggunakan array dua dimensi untuk menyimpan hasil dari 5 percobaan, dimana setiap percobaan memiliki 4 warna. Terdapat array referensi yang berisi urutan warna yang benar: merah, kuning, hijau, dan ungu. Program meminta input untuk kelima percobaan, kemudian membandingkan setiap warna dalam setiap percobaan dengan urutan warna referensi. Perbandingan dilakukan dengan mengabaikan perbedaan huruf besar/kecil menggunakan strings.ToLower(). Jika semua warna dalam suatu percobaan cocok dengan referensi, variabel 'beda' akan tetap true. Jika ada ketidakcocokan, program akan menghentikan pengecekan dan mengatur 'beda' menjadi false.

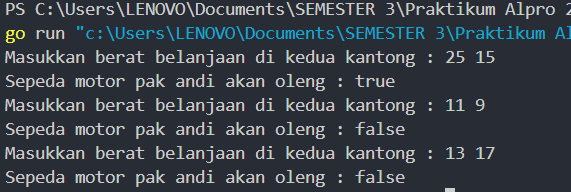
1. Buatlah sebuah program yang menerima input sebuah bilangan bulat positif (dan tidak nol) N, kemudian program akan meminta input berupa nama bunga secara berulang sebanyak N kali dan nama tersebut disimpan ke dalam pita.

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

Program di atas membuat sebuah "pita" yang terdiri dari rangkaian nama bunga. Program pertama-tama meminta pengguna untuk memasukkan jumlah bunga (N) yang akan dirangkai. Kemudian, melalui sebuah loop yang berjalan sebanyak N kali, program meminta nama bunga satu per satu. Setiap nama bunga yang dimasukkan akan ditambahkan ke variabel string 'pita', dipisahkan dengan tanda " - ". Setelah semua nama bunga dimasukkan, program menampilkan rangkaian lengkap nama-nama bunga tersebut sebagai sebuah pita.

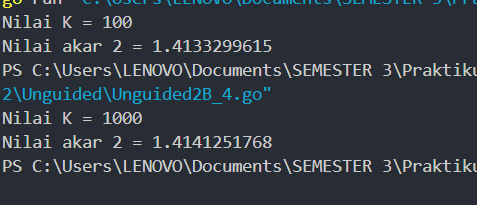
1. Buatlah program Pak Andi yang menerima input dua buah bilangan real positif yang menyatakan berat total masing-masing isi kantong terpal. Program akan terus meminta input bilangan tersebut hingga salah satu kantong terpal berisi 9 kg atau lebih.

|  |
| --- |
| package main  import (  "fmt"  "math"  )  func main() {  var berat1, berat2 float64  for {  fmt.Print("Masukkan berat belanjaan di kedua kantong : ")  fmt.Scan(&berat1, &berat2)  if !((berat1+berat2 > 150) || berat1 < 0 || berat2 < 0) {  fmt.Println("Sepeda motor pak andi akan oleng :", math.Abs(berat1-berat2) >= 9)  } else {  fmt.Println("Proses Selesai")  break  }  }  } |

Program di atas mensimulasikan pengecekan keseimbangan beban pada sepeda motor berdasarkan berat belanjaan di dua kantong. Program berjalan dalam loop tak terbatas, meminta pengguna memasukkan berat untuk kedua kantong. Program akan memeriksa tiga kondisi: apakah total berat tidak melebihi 150, dan apakah kedua berat tidak negatif. Jika kondisi ini terpenuhi, program menghitung selisih absolut antara kedua berat menggunakan math.Abs() dan menentukan apakah sepeda motor akan oleng (jika selisih lebih dari atau sama dengan 9). Jika total berat melebihi 150 atau salah satu berat negatif, program akan menampilkan "Proses Selesai" dan mengakhiri loop.

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

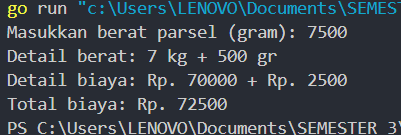
|  |
| --- |
| package main  import (  "fmt"  "math"  )  func main() {  var k, hasil float64 = 0, 1  fmt.Print("Nilai K = ")  fmt.Scan(&k)  for i := 0; i < int(k); i++ {  hasil \*= math.Pow(4\*float64(i)+2, 2) / ((4\*float64(i) + 1) \* (4\*float64(i) + 3))  }  fmt.Printf("Nilai akar 2 = %.10f", hasil)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

Program di atas mendeklarasikan dua variabel float64: k sebagai input dan hasil yang diinisialisasi dengan nilai 1. Program meminta pengguna memasukkan nilai k yang menentukan jumlah iterasi perhitungan. Selanjutnya, program menggunakan loop for untuk menghitung hasil dengan rumus perkalian berurutan, di mana setiap iterasi mengalikan hasil sebelumnya dengan perhitungan (4i+2)² dibagi dengan ((4i+1) \* (4i+3)), dimana i adalah indeks iterasi. Fungsi math.Pow digunakan untuk menghitung pangkat dua.

Modul 2C

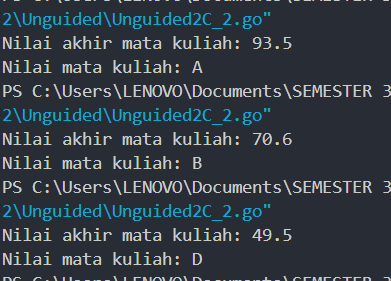
1. PT POS membutuhkan aplikasi perhitungan biaya kirim berdasarkan berat parsel. Maka, buatlah program BiayaPos untuk menghitung biaya pengiriman tersebut dengan ketentuan sebagai berikut!

|  |
| --- |
| 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)  } |

Program di atas sistem perhitungan biaya pengiriman parsel berdasarkan berat. Program meminta input berat parsel dalam gram, kemudian mengkonversinya menjadi kilogram (kg) dan sisa gram. Biaya dasar dihitung dengan mengalikan jumlah kilogram dengan Rp. 10.000. Program memiliki aturan khusus untuk biaya tambahan: jika berat 10 kg atau lebih, tidak ada biaya tambahan; jika kurang dari 10 kg, biaya tambahan dihitung berdasarkan sisa gram - untuk sisa gram kurang dari 500, dikenakan Rp. 15 per gram, sedangkan untuk sisa gram 500 atau lebih dikenakan Rp. 5 per gram.

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

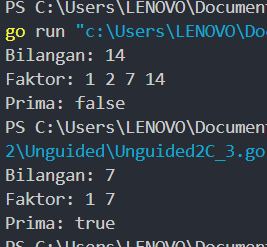
|  |
| --- |
| package main  import "fmt"  func main() {  var nam float64  var nmk string  fmt.Print("Nilai akhir mata kuliah: ")  fmt.Scanln(&nam)  if nam > 80 {  nmk = "A"  } else if nam > 72.5 {  nmk = "AB"  } else if nam > 65 {  nmk = "B"  } else if nam > 57.5 {  nmk = "BC"  } else if nam > 50 {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



Program di atas sistem konversi nilai angka menjadi nilai huruf untuk mata kuliah. Program meminta input berupa nilai akhir mata kuliah dalam bentuk angka (float64), kemudian menggunakan struktur if-else untuk menentukan nilai huruf (nmk) yang sesuai. Konversi dilakukan dengan ketentuan: nilai di atas 80 mendapat A, di atas 72.5 mendapat AB, di atas 65 mendapat B, di atas 57.5 mendapat BC, di atas 50 mendapat C, di atas 40 mendapat D, dan nilai di bawah atau sama dengan 40 mendapat E. Setelah menentukan nilai huruf, program menampilkan hasilnya menggunakan fmt.Println.

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

|  |
| --- |
| 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)  } |



Program di atas mencari faktor-faktor dari suatu bilangan dan menentukan apakah bilangan tersebut adalah bilangan prima. Program meminta input sebuah bilangan bulat, kemudian menggunakan loop untuk mencari semua faktornya dengan cara mengecek sisa bagi (modulo) bilangan tersebut dengan setiap angka dari 1 hingga bilangan itu sendiri. Setiap faktor yang ditemukan akan ditampilkan. Bersamaan dengan itu, program juga menggunakan variabel boolean 'prima' yang diinisialisasi dengan nilai true, dan akan diubah menjadi false jika ditemukan faktor selain 1 dan bilangan itu sendiri.