**LAPORAN PRAKTIKUM   
ALGORITMA DAN PEMROGRAMAN 2**

**MODUL IV**

**PROSEDUR**

**Sebuah gambar berisi logo, simbol, Grafis, Font

Deskripsi dibuat secara otomatis**

**Disusun Oleh :**

**M. Haidar Akhbiyani / 2311102276**

**S1-IF-11-06**

**Dosen Pengampu :**

**Abednego Dwi Septiadi**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

1. **DASAR TEORI**
2. **GUIDED**
   * + 1. Guided 1

**Sourcecode**

|  |
| --- |
| package main  import "fmt"  func main(){  var bilangan int  var pesan string  fmt.Scan(&bilangan, &pesan)  cetakPesan(pesan, bilangan)  }  func cetakPesan(M string, flag int){  var jenis string =""  if flag == 0 {  jenis = "error"  }else if flag == 1 {  jenis = "warning"  }else if flag == 2 {  jenis = "informasi"  }  fmt.Println(M, jenis)  } |

**Screenshoot Output**

****

**Deskripsi Program**

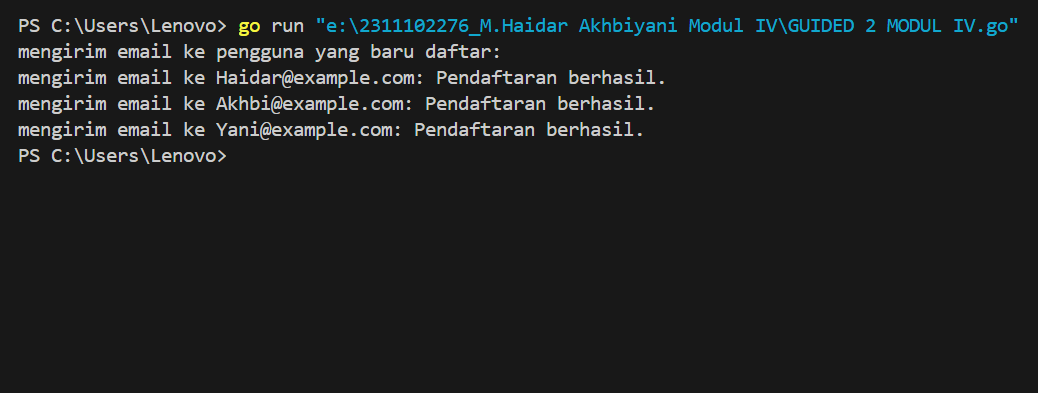
Program diatas menerima inputan bilangan ataupun pesan,kemudian dilanjut mencetak pesan dengan jenis sesuai bilanganya. 0 untuk "error", 1 untuk "warning", dan 2 untuk "informasi".

* + - 1. Guided 2

**Sourcecode**

|  |
| --- |
| package main  import "fmt"  func sendEmailNotification(email string) {      fmt.Printf("mengirim email ke %s: Pendaftaran berhasil.\n" , email)  }  func main() {      emails := []string{"Haidar@example.com", "Akhbi@example.com", "Yani@example.com"}      fmt.Println("mengirim email ke pengguna yang baru daftar:")      for \_, email := range emails {          sendEmailNotification(email)      }  } |

**Screenshoot Output**

****

**Deskripsi Program**

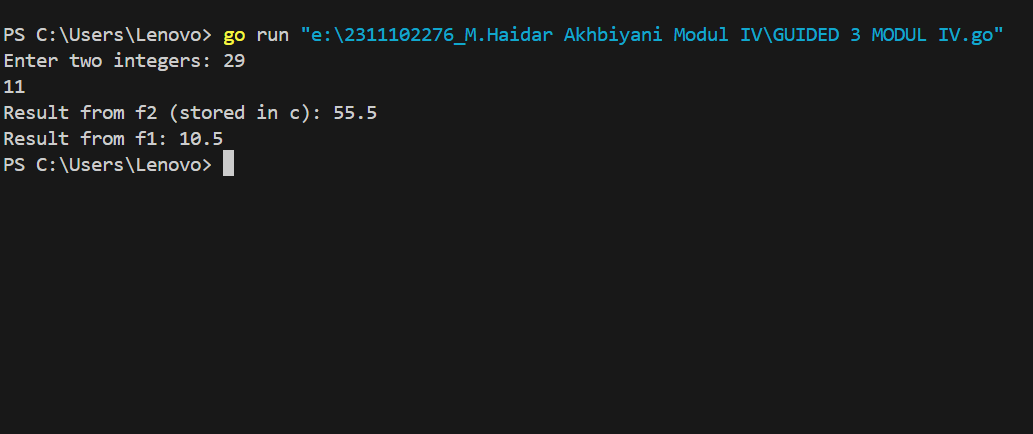
Program ini mengirimkan notifikasi email kepada daftar pengguna yang baru mendaftar. Fungsi sendEmailNotification digunakan untuk mencetak pesan konfirmasi pendaftaran yang berhasil ke alamat email yang diberikan. Dalam fungsi main, sebuah slice berisi alamat email pengguna di-loop, dan untuk setiap email, fungsi sendEmailNotification dipanggil untuk mengirimkan pesan tersebut.

* + - 1. Guided 3

**Sourcecode**

|  |
| --- |
| package main  import "fmt"  func f1(x, y int) float64 {      var hasil float64      hasil = float64(2\*x) - 0.5\*float64(y) + 3.0      return hasil  }  func f2(x, y int, hasil \*float64) {      \*hasil = float64(2\*x) - 0.5\*float64(y) + 3.0  }  func main() {      var a, b int      var c float64        fmt.Print("Enter two integers: ")      fmt.Scan(&a, &b)      f2(a, b, &c)      fmt.Println("Result from f2 (stored in c):", c)      resultF1 := f1(b, a)      fmt.Println("Result from f1:", resultF1)  } |

**Screenshoot Output**

****

**Deskripsi Program**

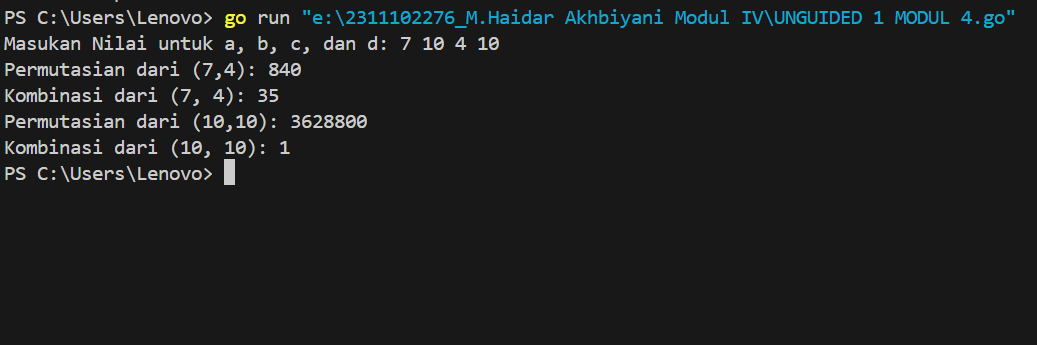
Program ini menghitung hasil dari dua fungsi matematika yang melibatkan dua bilangan bulat. Fungsi f1 menghitung nilai berdasarkan rumus 2\*x - 0.5\*y + 3.0 dan mengembalikan hasilnya dalam tipe data float64. Fungsi f2 melakukan perhitungan yang sama, tetapi hasilnya disimpan melalui pointer pada variabel yang diterima sebagai parameter. Dalam fungsi main, program meminta input dua bilangan bulat dari pengguna, kemudian menghitung dan menampilkan hasil perhitungan dari kedua fungsi (f1 dan f2). Hasil perhitungan dari f2 disimpan dalam variabel c, sementara hasil dari f1 disimpan dalam resultF1.

1. **UNGUIDED**
   * + 1. Unguided 1

**Sourcecode**

|  |
| --- |
| package main  import (      "fmt"  )  // Factorial function  func factorial(n int) int {      hasil := 1      for i := 1; i <= n; i++ {          hasil \*= i      }      return hasil  }  // Permutation function  func permutation(a, b int) int {      kiri := factorial(a)      kanan := factorial(a - b)      return kiri / kanan  }  // Combination function  func combination(a, b int) int {      kiri := factorial(a)      kanan1 := factorial(b)      kanan2 := factorial(a - b)      return kiri / (kanan1 \* kanan2)  }  // TaskDiskrit handles permutation and combination calculations  func taskDiskrit(a, b, c, d int) {      fmt.Printf("Permutasian dari (%d,%d): %d\n", a, c, permutation(a, c))      fmt.Printf("Kombinasi dari (%d, %d): %d\n", a, c, combination(a, c))      fmt.Printf("Permutasian dari (%d,%d): %d\n", b, d, permutation(b, d))      fmt.Printf("Kombinasi dari (%d, %d): %d\n", b, d, combination(b, d))  }  func main() {      var a, b, c, d int      fmt.Print("Masukan Nilai untuk a, b, c, dan d: ")      fmt.Scan(&a, &b, &c, &d)      if a >= c && b >= d {          taskDiskrit(a, b, c, d)      } else {          fmt.Println("Invalid input: a harus >= c and b harus >= d.")      }  } |

**Screenshoot Output**

****

**Deskripsi Program**

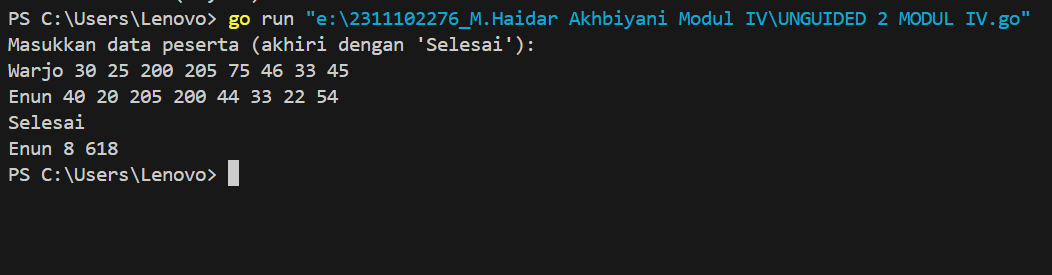
Program ini menghitung nilai permutasi dan kombinasi berdasarkan input empat bilangan bulat. Fungsi factorial digunakan untuk menghitung faktorial dari suatu bilangan. Fungsi permutation dan combination menghitung nilai permutasi dan kombinasi berdasarkan rumus matematika menggunakan hasil faktorial. Fungsi taskDiskrit menampilkan hasil perhitungan permutasi dan kombinasi untuk dua pasang nilai input: (a, c) dan (b, d). Dalam fungsi main, program meminta input dari pengguna untuk nilai a, b, c, dan d, lalu memvalidasi input dengan memastikan bahwa a >= c dan b >= d. Jika valid, program melanjutkan dengan menghitung dan menampilkan hasil perhitungan, jika tidak, program menampilkan pesan kesalahan.

* + - 1. Unguided 2

**Sourcecode**

|  |
| --- |
| package main  import (      "bufio"      "fmt"      "os"      "strconv"      "strings"  )  func hitungSkor(waktu []int) (int, int) {      const waktuMaks = 301      soalTerselesaikan := 0      totalWaktu := 0      for \_, w := range waktu {          if w < waktuMaks {              soalTerselesaikan++              totalWaktu += w          }      }      return soalTerselesaikan, totalWaktu  }  func main() {      scanner := bufio.NewScanner(os.Stdin)      var namaPemenang string      var maxSoal, minTotalWaktu int      maxSoal = 0      minTotalWaktu = 1000000      fmt.Println("Masukkan data peserta (akhiri dengan 'Selesai'): ")      for scanner.Scan() {          input := scanner.Text()          if strings.ToLower(input) == "selesai" {              break          }          parts := strings.Fields(input)          nama := parts[0]          var waktu []int          for i := 1; i < len(parts); i++ {              w, \_ := strconv.Atoi(parts[i])              waktu = append(waktu, w)          }          soalTerselesaikan, totalWaktu := hitungSkor(waktu)          if soalTerselesaikan > maxSoal || (soalTerselesaikan == maxSoal && totalWaktu < minTotalWaktu) {              namaPemenang = nama              maxSoal = soalTerselesaikan              minTotalWaktu = totalWaktu          }      }      fmt.Printf("%s %d %d\n", namaPemenang, maxSoal, minTotalWaktu)  } |

**Screenshoot Output**

****

**Deskripsi Program**

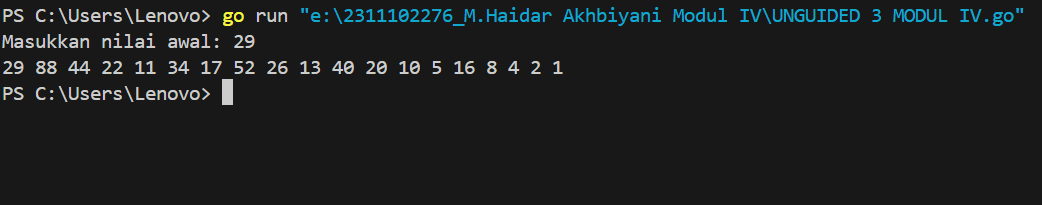
Program diatas memiliki kegunaan untuk menentukan siapa pemenag dari kompetisi. Pemenang ditentukan berdasarkan jumlah soal terbanyak yang diselesaikan, dan jika ada peserta dengan jumlah soal yang sama, maka yang memiliki total waktu paling sedikit yang menang. Program berakhir saat pengguna mengetikkan 'Selesai'.

* + - 1. Unguided 3

**Sourcecode**

|  |
| --- |
| package main  import (      "fmt"  )  func cetakDeret(n int) {      for n != 1 {          fmt.Print(n, " ")          if n%2 == 0 {              n = n / 2          } else {              n = 3\*n + 1          }      }      fmt.Print(n)  }  func main() {      var n int      fmt.Print("Masukkan nilai awal: ")      fmt.Scan(&n)      if n > 0 && n < 1000000 {          cetakDeret(n)      } else {          fmt.Println("Masukan harus bilangan positif kurang dari 1.000.000")      }  } |

**Screenshoot Output**

****

**Deskripsi Program**

Program ini mencetak deret angka berdasarkan aturan Collatz, yang dimulai dengan menerima input bilangan positif kurang dari 1.000.000. Setiap angka dalam deret diproses dengan dua aturan: jika angka genap, dibagi 2; jika ganjil, dikalikan 3 dan ditambah 1. Proses ini berulang hingga angka mencapai 1. Program akan mencetak setiap angka dalam deret tersebut dan mengakhiri eksekusi setelah angka mencapai 1. Jika input tidak memenuhi syarat (bilangan negatif atau lebih dari 1.000.000), program akan memberikan pesan kesalahan.