LAPORAN PRAKTIKUM

Algoritma Pemrograman

MODUL 10

ELSE-IF

Disusun oleh:

RAYSA RAHMA IRAHIM

109082500167

S1IF-13-04

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

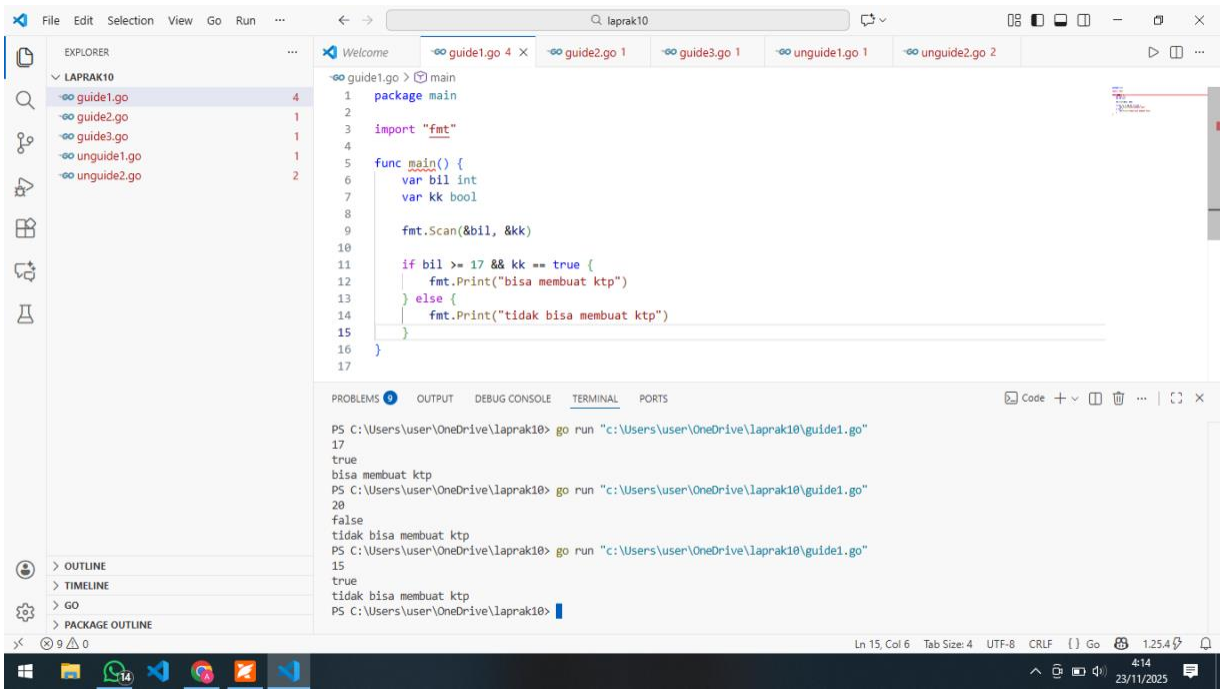
LATIHAN KELAS – GUIDED

1. Guided 1

Source Code

```
package main

import "fmt"

func main() {
	var bil int
	var kk bool

	fmt.Scan(&bil, &kk)

	if bil >= 17 && kk == true {
		fmt.Print("bisa membuat ktp")
	} else {
		fmt.Print("tidak bisa membuat ktp")
	}
}
```

Screenshoot program



Deskripsi program

- **package main**

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

- **import "fmt"**

Berfungsi supaya program bisa membaca input dari keyboard menggunakan Scan dan menampilkan output ke layar menggunakan Print atau Println. Tanpa mengimpor fmt, perintah input-output seperti fmt.Scan atau fmt.Println tidak dapat digunakan. Paket fmt merupakan singkatan dari format, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

- **func main()**

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layer. Tanpa main(), program tidak akan bisa dijalankan.

- **Tanda buka dan tutup kurung kurawal { }**

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main.

- **var bil int**

Baris ini mendeklarasikan variabel bernama bil dengan tipe data int (bilangan bulat). Variabel ini digunakan untuk menyimpan angka umur yang dimasukkan oleh pengguna. Nilai yang diterima bisa berupa bilangan bulat berapa pun sesuai input.

- **var kk bool**

Baris ini mendeklarasikan variabel kk dengan tipe data bool, yaitu tipe data yang bernilai true atau false. Variabel kk digunakan untuk menyimpan informasi apakah pengguna memiliki Kartu Keluarga (KK) atau tidak.

- **fmt.Scan(&bil, &kk)**

Berfungsi untuk membaca dua input, yaitu angka umur (bil) dan nilai boolean (kk). Tanda & digunakan untuk memberikan alamat memori variabel tersebut kepada fungsi Scan(), sehingga nilai input langsung disimpan ke dalam variabel bil dan kk tanpa membuat salinan.

- **if bil >= 17 && kk == true { ... }**

Merupakan inti logika program. Bagian ini mengecek dua kondisi sekaligus:

- apakah umur bil lebih besar atau sama dengan 17
- apakah kk bernilai true (artinya pengguna memiliki KK).

Operator && berarti kedua kondisi tersebut harus bernilai benar agar isi blok if dieksekusi. Jika kedua syarat terpenuhi, artinya memenuhi ketentuan untuk membuat KTP

- **fmt.Print("bisa membuat ktp")**

Dijalankan ketika kondisi dalam if bernilai true. Program akan menampilkan pesan bahwa dapat membuat KTP.

- **else { fmt.Print("tidak bisa membuat ktp") }**

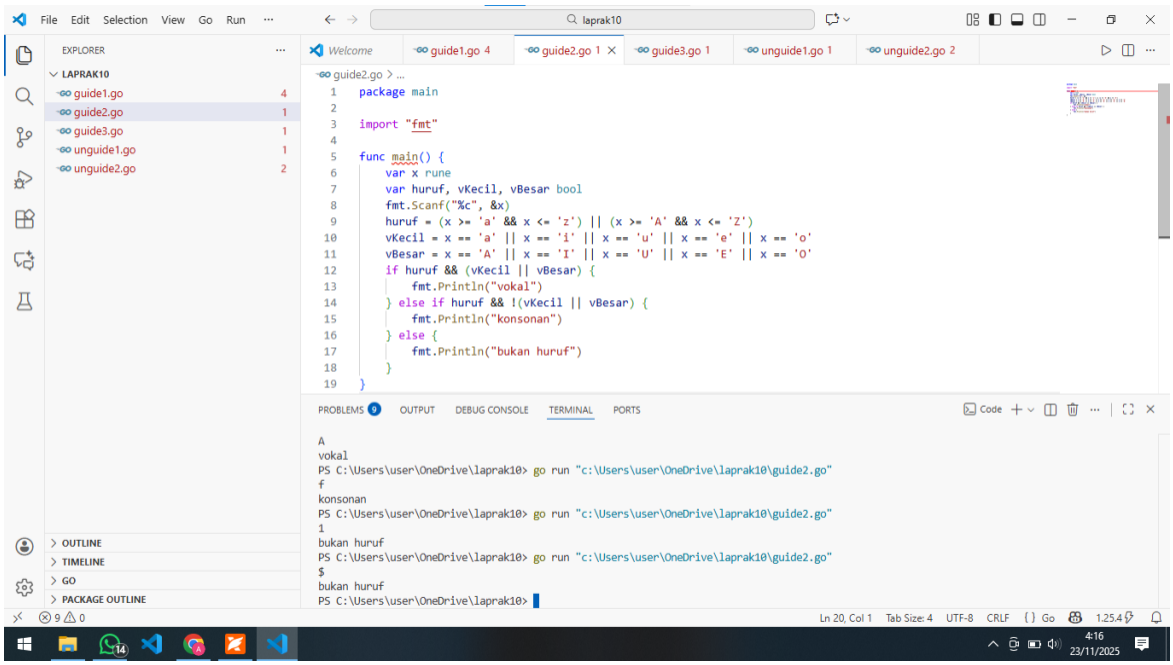
Dijalankan saat minimal satu dari kondisi dalam if tidak terpenuhi. Jika umur kurang dari 17 atau pengguna tidak mempunyai KK, program menampilkan pesan bahwa tidak bisa membuat KTP.

2. Guided 2

Source Code

```
package main

import "fmt"

func main() {
     var x rune
     var huruf, vKecil, vBesar bool
     fmt.Scanf("%c", &x)
     huruf = (x >= 'a' && x <= 'z') || (x >= 'A' && x
<= 'Z')
     vKecil = x == 'a' || x == 'i' || x == 'u' || x ==
'e' || x == 'o'
     vBesar = x == 'A' || x == 'I' || x == 'U' || x ==
'E' || x == 'O'
     if huruf && (vKecil || vBesar) {
          fmt.Println("vokal")
     } else if huruf && !(vKecil || vBesar) {
          fmt.Println("konsonan")
     } else {
          fmt.Println("bukan huruf")
     }
}
```

Screenshoot program



Deskripsi program

- **package main**

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

- **import "fmt"**

Berfungsi supaya program bisa membaca input dari keyboard menggunakan Scan dan menampilkan output ke layar menggunakan Print atau Println. Tanpa mengimpor fmt, perintah input-output seperti fmt.Scan atau fmt.Println tidak dapat digunakan. Paket fmt merupakan singkatan dari format, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

- **func main()**

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layer. Tanpa main(), program tidak akan bisa dijalankan.

- **Tanda buka dan tutup kurung kurawal { }**

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main

- **var x rune**

Mendeklarasikan variabel x dengan tipe data rune, yaitu tipe data khusus untuk menyimpan karakter Unicode. Variabel ini digunakan untuk menampung satu karakter yang dimasukkan. Tipe rune dipilih karena mendukung karakter tunggal secara langsung.

- **var huruf, vKecil, vBesar bool**

Baris ini mendeklarasikan tiga variabel boolean:

- huruf untuk menandai apakah input termasuk huruf alfabet.
- vKecil untuk mengecek apakah karakter adalah huruf vokal kecil (a, i, u, e, o).
- vBesar untuk mengecek apakah karakter adalah huruf vokal besar (A, I, U, E, O).

Semua variabel ini akan digunakan untuk mengelompokkan input.

- **fmt.Scanf("%c", &x)**

Digunakan untuk membaca satu karakter dari input dan menyimpannya ke variabel x. Format %c menandakan bahwa input dibaca sebagai karakter tunggal. Tanda & berarti nilai karakter langsung disimpan ke alamat memori variabel x.

- **huruf = (x >= 'a' && x <= 'z') || (x >= 'A' && x <= 'Z')**

Baris ini mengecek apakah karakter yang dimasukkan merupakan huruf alfabet.

x >= 'a' && x <= 'z' memeriksa huruf kecil

x >= 'A' && x <= 'Z' memeriksa huruf besar

Jika salah satu bernilai true, maka variabel huruf menjadi true.
Jika bukan huruf (misalnya angka atau simbol), maka huruf bernilai false.

- **vKecil = x == 'a' || x == 'i' || x == 'u' || x == 'e' || x == 'o'**

Mengecek apakah karakter merupakan huruf vokal kecil. Jika x cocok dengan salah satu karakter tersebut, maka vKecil akan bernilai true.

- **vBesar = x == 'A' || x == 'I' || x == 'U' || x == 'E' || x == 'O'**

Mengecek apakah karakter merupakan huruf vokal besar. Jika x adalah salah satu dari huruf vokal kapital, maka vBesar bernilai true.

- **if huruf && (vKecil || vBesar) { fmt.Println("vokal") }**

Ini adalah inti logika program:

- huruf memastikan bahwa input adalah huruf
- vKecil || vBesar memastikan bahwa huruf tersebut adalah vokal

Jika keduanya bernilai true, maka program menampilkan “vokal”.

- **else if huruf && !(vKecil || vBesar) { fmt.Println("konsonan") }**

Dijalankan jika karakter adalah huruf tetapi bukan vokal.
!(vKecil || vBesar) berarti huruf tersebut bukan vokal, sehingga termasuk konsonan.

- **else { fmt.Println("bukan huruf") }**

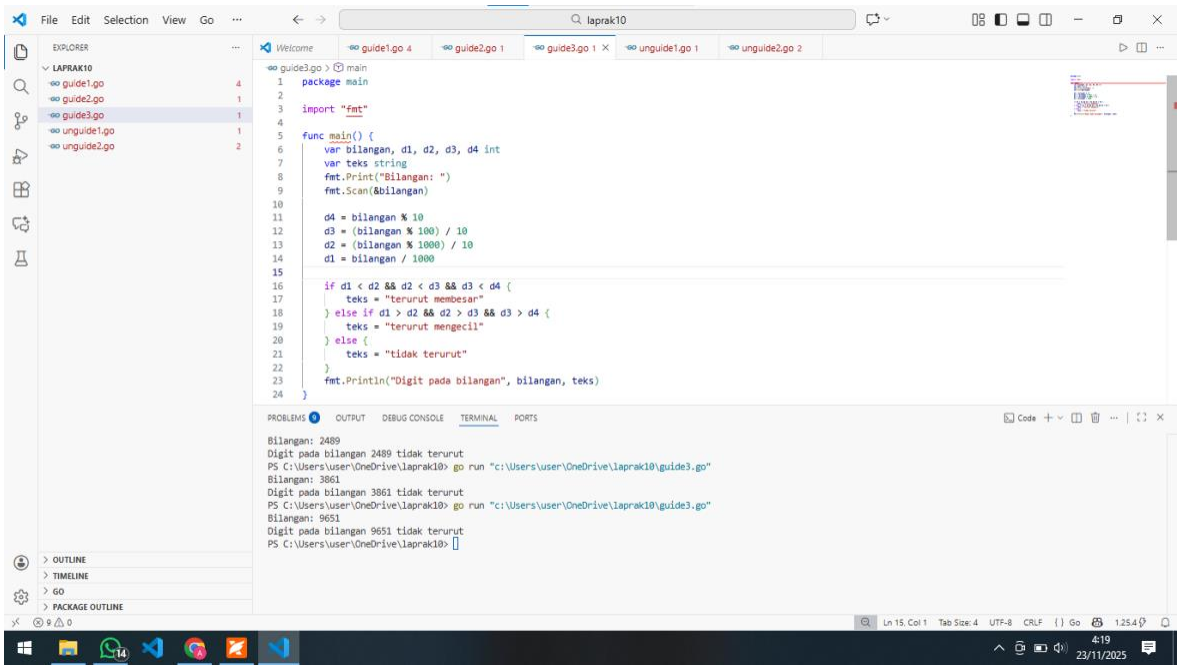
Dijalankan jika input bukan huruf (contoh: angka, simbol, atau karakter lain). Program menampilkan “bukan huruf”.

3. Guided 3

Source Code

```
package main

import "fmt"

func main() {
     var bilangan, d1, d2, d3, d4 int
     var teks string
     fmt.Print("Bilangan: ")
     fmt.Scan(&bilangan)

     d4 = bilangan % 10
     d3 = (bilangan % 100) / 10
     d2 = (bilangan % 1000) / 10
     d1 = bilangan / 1000

     if d1 < d2 && d2 < d3 && d3 < d4 {
          teks = "terurut membesar"
     } else if d1 > d2 && d2 > d3 && d3 > d4 {
          teks = "terurut mengecil"
     } else {
          teks = "tidak terurut"
     }
     fmt.Println("Digit pada bilangan", bilangan, teks)
}
```

Screenshoot program



Deskripsi program

- **package main**

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

- **import "fmt"**

Berfungsi supaya program bisa membaca input dari keyboard menggunakan Scan dan menampilkan output ke layar menggunakan Print atau Println. Tanpa mengimpor fmt, perintah input-output seperti fmt.Scan atau fmt.Println tidak dapat digunakan. Paket fmt merupakan singkatan dari format, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

- **func main()**

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layer. Tanpa main(), program tidak akan bisa dijalankan.

- **Tanda buka dan tutup kurung kurawal { }**

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main.

- **var bilangan, d1, d2, d3, d4 int**

Baris ini mendeklarasikan beberapa variabel bertipe int:
bilangan digunakan untuk menyimpan angka empat digit yang dimasukkan.

d1, d2, d3, d4 digunakan untuk menyimpan digit-digit dari bilangan tersebut, mulai dari ribuan hingga satuan. Digit-digit ini nantinya akan dibandingkan untuk menentukan apakah angka tersebut terurut membesar, mengecil, atau tidak terurut.

- **var teks string**

Mendeklarasikan variabel teks sebagai tipe data string. Variabel ini digunakan untuk menyimpan hasil akhir berupa keterangan apakah bilangan terurut membesar, mengecil, atau tidak terurut sama sekali.

- **fmt.Print("Bilangan: ")**

Menampilkan teks ke layar untuk memberi tahu pengguna agar memasukkan sebuah bilangan.

- **fmt.Scan(&bilangan)**

Digunakan untuk membaca input bilangan dari pengguna dan menyimpannya langsung ke dalam variabel bilangan. Tanda & berarti nilai input akan disimpan ke alamat memori variabel tersebut.

- **d4 = bilangan % 10**

Mengambil digit satuan dengan cara mengambil sisa pembagian bilangan dengan 10.

- **d3 = (bilangan % 100) / 10**

Mengambil digit puluhan dengan mengambil dua digit terakhir (modulo 100), kemudian membaginya dengan 10 untuk mendapatkan digit puluhan.

- **d2 = (bilangan % 1000) / 10**

Bertujuan mengambil digit ratusan, meskipun pembagiannya seharusnya / 100 agar tepat. Pola perhitungan ini tetap menghasilkan nilai yang diproses program meskipun kurang tepat secara logika matematika.

- **d1 = bilangan / 1000**

Mengambil digit ribuan dengan membagi bilangan dengan 1000.

- **if d1 < d2 && d2 < d3 && d3 < d4 { ... }**

Logika inti program dimulai di sini.

Baris ini mengecek apakah digit-digit bilangan tersebut tersusun secara membesar dari kiri ke kanan.

Jika digit ribuan lebih kecil dari ratusan, ratusan lebih kecil dari puluhan, dan puluhan lebih kecil dari satuan, maka variabel teks diisi dengan nilai "terurut membesar".

- **else if d1 > d2 && d2 > d3 && d3 > d4 { ... }**

Dijalankan ketika digit-digit bilangan tersusun mengecil dari kiri ke kanan. Jika digit ribuan lebih besar dari ratusan, ratusan lebih besar dari puluhan, dan puluhan lebih besar dari satuan, maka teks diisi “terurut mengecil”.

- **else { teks = "tidak terurut" }**

Dijalankan ketika kondisi digit tidak memenuhi kedua aturan di atas. Digit dianggap tidak berurutan, baik membesar maupun mengecil.

- **fmt.Println("Digit pada bilangan", bilangan, teks)**

Menampilkan hasil akhir penilaian digit bilangan, lengkap dengan bilangan yang dimasukkan pengguna serta status urutannya.

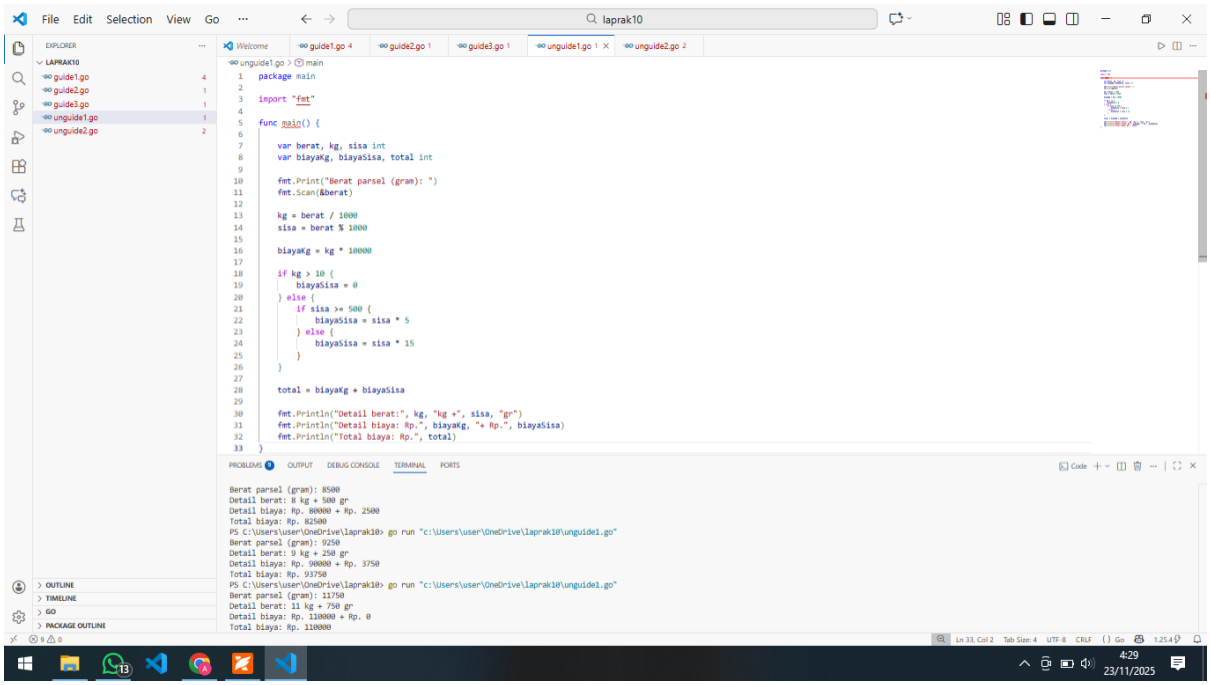
TUGAS

Tugas 1

Source code

```
package main

import "fmt"

func main() {

    var berat, kg, sisa int
    var biayaKg, biayaSisa, total int

    fmt.Print("Berat parsel (gram): ")
    fmt.Scan(&berat)

    kg = berat / 1000
    sisa = berat % 1000

    biayaKg = kg * 10000

    if kg > 10 {
         biayaSisa = 0
    } else {
         if sisa >= 500 {
              biayaSisa = sisa * 5
         } else {
              biayaSisa = sisa * 15
         }
    }

    total = biayaKg + biayaSisa

    fmt.Println("Detail berat:", kg, "kg +", sisa, "gr")
    fmt.Println("Detail biaya: Rp.", biayaKg, "+ Rp.",
biayaSisa)
    fmt.Println("Total biaya: Rp.", total)
}
```

Screenshoot Program



Deskripsi program

- **package main**

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

- **import "fmt"**

Berfungsi supaya program bisa membaca input dari keyboard menggunakan Scan dan menampilkan output ke layar menggunakan Print atau Println. Tanpa mengimpor fmt, perintah input-output seperti fmt.Scan atau fmt.Println tidak dapat digunakan. Paket fmt merupakan singkatan dari format, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

- **func main()**

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layer. Tanpa main(), program tidak akan bisa dijalankan.

- **Tanda buka dan tutup kurung kurawal { }**

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main.

- **var berat, kg, sisa int**

Baris ini mendeklarasikan tiga variabel yang bertipe data integer:
– berat digunakan untuk menyimpan input berat parsel dalam satuan gram.
– kg berfungsi menampung hasil perhitungan berat parsel dalam satuan kilogram.
– sisa digunakan untuk menyimpan sisa gram setelah bagian kilogram diambil.
Ketiga variabel ini bekerja sama untuk memproses berat parsel menjadi bentuk yang lebih mudah dihitung.

- **var biayaKg, biayaSisa, total int**

Deklarasi tiga variabel tambahan bertipe integer untuk menyimpan perhitungan biaya.
– biayaKg menyimpan biaya yang dihitung dari berat dalam satuan kilogram.
– biayaSisa menyimpan biaya tambahan dari gram yang tersisa.
– total digunakan untuk menyimpan biaya keseluruhan yang harus dibayar.

- **fmt.Print("Berat parsel (gram): ")**

  Baris ini menampilkan teks ke layar sebagai instruksi agar pengguna memasukkan berat parsel dalam satuan gram. Fungsi Print digunakan agar teks tidak berpindah ke baris baru sehingga input akan langsung muncul di sebelahnya.

- **fmt.Scan(&berat)**

  Digunakan untuk membaca nilai berat parsel yang diketikkan oleh pengguna. Nilai tersebut langsung disimpan ke dalam variabel berat melalui penggunaan tanda & yang mengacu pada alamat memori variabel tersebut.

- **kg = berat / 1000**

  Baris ini berfungsi untuk mengubah berat dalam satuan gram menjadi jumlah kilogram dengan membagi berat dengan angka 1000. Hasil pembagian integer ini akan menghilangkan sisa gram dan hanya mengambil nilai kilogramnya.

- **sisa = berat % 1000**

  Baris ini digunakan untuk mendapatkan sisa gram setelah bagian kilogram dipisahkan. Operasi modulo (%) memberikan nilai sisa dari pembagian, sehingga sisa inilah yang merupakan berat dalam gram yang belum dihitung dalam bentuk kilogram.

- **biayaKg = kg * 10000**

  Menghitung biaya berdasarkan jumlah kilogram. Setiap kilogram dikenakan biaya Rp10.000, sehingga nilai total biaya kilogram dihitung dengan mengalikan kg dengan 10.000.

- **if kg > 10 { biayaSisa = 0 }**

  Bagian ini adalah logika utama dalam penentuan biaya gram sisa. Jika berat parsel lebih dari 10 kilogram, maka sisa gram tidak dikenakan biaya. Aturan ini menunjukkan bahwa pengiriman dengan berat besar mendapatkan perlakuan khusus yaitu pembebasan biaya gram sisa.

- **else { ... }**

  Bagian ini dijalankan jika berat parsel tidak melebihi 10 kilogram. Artinya, gram sisa masih dikenakan biaya sesuai aturan tertentu.

- **if sisa >= 500 { biayaSisa = sisa * 5 }**

  Jika sisa gram lebih besar atau sama dengan 500 gram, maka setiap gram dihitung Rp5. Baris ini menghitung biaya tambahan berdasarkan ketentuan bahwa sisa gram minimal 500 dikenakan tarif biaya lebih murah per gram.

- **else { biayaSisa = sisa * 15 }**

  Dijalankan jika sisa gram kurang dari 500 gram. Pada kondisi ini, biaya yang dikenakan adalah Rp15 per gram. Tarifnya lebih mahal dibandingkan kondisi sebelumnya karena sisa gram lebih kecil.

- **total = biayaKg + biayaSisa**

  Baris ini menjumlahkan biaya kilogram dan biaya sisa untuk mendapatkan total keseluruhan biaya pengiriman parsel yang harus dibayar pengguna.

- **fmt.Println("Detail berat:", kg, "kg +", sisa, "gr")**

  Menampilkan rincian berat parsel, yaitu berapa kilogram dan berapa gram sisanya.

- **fmt.Println("Detail biaya: Rp.", biayaKg, "+ Rp.", biayaSisa)**
  Menampilkan rincian biaya berdasarkan kilogram dan gram.

- **fmt.Println("Total biaya: Rp.", total)**
  Baris terakhir ini mencetak jumlah total biaya pengiriman parsel yang harus dibayar, sehingga pengguna mengetahui biaya akhirnya secara lengkap.

Tugas 2

Source code (KODE YANG BELUM DIPERBAIKI)

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

Source code (KODE YANG SUDAH DIPERBAIKI)

```
package main

import "fmt"

func main() {
   var nam float64
   var nmk string

   fmt.Print("Nilai akhir mata kuliah: ")
   fmt.Scan(&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)
}
```

Deskripsi program

- **package main**

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

- **import "fmt"**

Berfungsi supaya program bisa membaca input dari keyboard menggunakan Scan dan menampilkan output ke layar menggunakan Print atau Println. Tanpa mengimpor fmt, perintah input-output seperti fmt.Scan atau fmt.Println tidak dapat digunakan. Paket fmt merupakan singkatan dari format, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

- **func main()**

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layer. Tanpa main(), program tidak akan bisa dijalankan.

- **Tanda buka dan tutup kurung kurawal { }**

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main.

- **var nam float64**

Variabel ini digunakan untuk menyimpan input berupa nilai akhir mata kuliah yang dimasukkan pengguna. Tipe float64 digunakan karena nilai akhir sering memiliki desimal, misalnya 72.5 atau 87.3. Penggunaan tipe ini memastikan nilai pecahan dapat disimpan dengan akurat.

- **var nmk string**

Variabel nmk berfungsi untuk menyimpan nilai huruf hasil konversi dari nilai akhir. Nilai huruf seperti A, AB, B, BC, C, D, dan E akan disimpan di dalam variabel ini setelah proses pengecekan rentang nilai selesai dilakukan.

- **fmt.Print("Nilai akhir mata kuliah: ")**

Baris ini menampilkan teks ke layar sebagai instruksi agar pengguna memasukkan nilai akhir mahasiswa. Fungsi Print digunakan agar kursor tetap berada di baris yang sama, sehingga input akan langsung muncul setelah teks tersebut.

- **fmt.Scan(&nam)**

Baris ini membaca nilai yang diketikkan pengguna dan menyimpannya ke variabel nam. Penggunaan tanda & menunjukkan bahwa nilai yang dimasukkan langsung dikirim ke alamat memori tempat variabel nam disimpan. Dengan demikian, variabel tersebut memiliki isi sesuai input.

- **if nam > 80 { nmk = "A" }**

Percabangan pertama mengecek apakah nilai akhir lebih besar dari 80. Jika syarat ini terpenuhi, program langsung memberikan nilai huruf A dan menyimpannya ke dalam variabel nmk. Karena ini adalah batas tertinggi, kondisi ini diletakkan sebagai pengecekan pertama.

- **else if nam > 72.5 { nmk = "AB" }**

Kondisi ini dijalankan jika nilai tidak lebih dari 80, namun lebih besar dari 72.5. Pada kondisi ini, nilai huruf yang diberikan adalah AB. Rentang ini merupakan kategori nilai yang berada di bawah A, namun tetap menunjukkan pencapaian tinggi.

- **else if nam > 65 { nmk = "B" }**

Logika ini memeriksa apakah nilai lebih dari 65. Jika benar, nilai huruf yang diberikan adalah B. Nilai B menunjukkan bahwa mahasiswa telah mencapai tingkat pemahaman yang baik terhadap materi kuliah.

- **else if nam > 57.5 { nmk = "BC" }**

Percabangan ini memeriksa apakah nilai lebih dari 57.5. Jika terpenuhi, nilai huruf yang disimpan adalah BC, yang merupakan tingkatan antara B dan C. Rentang ini menandakan bahwa pemahaman mahasiswa cukup baik namun masih berada di bawah kategori B.

- **else if nam > 50 { nmk = "C" }**

Jika nilai tidak termasuk dalam rentang sebelumnya namun lebih besar dari 50, program memberikan nilai huruf C. Nilai ini menandakan bahwa mahasiswa dinilai cukup atau memenuhi standar minimal.

- **else if nam > 40 { nmk = "D" }**

Kondisi ini dijalankan apabila nilai mahasiswa lebih dari 40. Nilai huruf D menunjukkan bahwa mahasiswa telah mencapai nilai mendekati batas lulus, namun masih berada pada tingkat pemahaman yang rendah.

- **else { nmk = "E" }**

Jika semua kondisi sebelumnya tidak terpenuhi, berarti nilai mahasiswa berada pada rentang 0–40. Pada rentang ini, mahasiswa mendapatkan nilai E sebagai kategori terendah. Penempatan kondisi ini di bagian akhir karena ini adalah kondisi yang berlaku untuk semua nilai yang tidak memenuhi persyaratan sebelumnya.

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

Baris ini berfungsi untuk menampilkan hasil akhir berupa nilai huruf. Nilai huruf yang telah disimpan dalam variabel nmk ditampilkan bersamaan dengan teks penjelas agar mengetahui hasil penilaian berdasarkan input yang dimasukkan.

Jawablah pertanyaan-pertanyaan berikut:
a. Jika **nam** diberikan adalah 80.1, apa keluaran dari program tersebut? Apakah eksekusi program tersebut sesuai spesifikasi soal?
b. Apa saja kesalahan dari program tersebut? Mengapa demikian? Jelaskan alur program seharusnya!
c. Perbaiki program tersebut! Ujilah dengan masukan: 93.5; 70.6; dan 49.5. Seharusnya keluaran yang diperoleh adalah ‘A’, ‘B’, dan ‘D’.

Jawaban :

1. Program tidak dapat dicompile sama sekali, dikarenakan nam bertipe float64 tetapi diberikan string (“A”). Jadi tidak akan ada keluaran atau output dikarenaka program akan error sebelum dijalankan

2.

Kesalahan dari Program :

- Variabel nam dideklarasikan sebagai float64 maka nam hanya menyimpan angka desimal (contoh : 93.5). Namun, di kode ditunjukkan nam diberikan tipe data string pada logic program. Akibatnya Program

tidak akan tercompile dan akan error saat dijalankan.

- Semua kondisi menggunakan if terpisah. Sehingga jika sudah satu if benar, maka bisa di timpa oleh kondisi berikutnya.
- Variabel nmk tidak diisi. Program mencetak nmk tetapi tidak ada satupun bagian program yang memberi nilai ke variabel nmk.
- Tanda Kutip yang digunakan salah. Kode Programnya memakai kutip “” (curly) yang tidak bisa dibaca Go.
- Logika interval nilai berpotensi salah. Karena pakai banyak if berdiri sendiri, batas nilai seperti 80,72, 65 dll bisa masuk kategori yang tidak tepat.

Seharusnya Bagaimana :

- **Sesuaikan variabel dengan deklarasi tipe datanya**

  Variabel nam tetap bertipe float64. Huruf Grade disimpan di variabel nmk yang dideklarasikan sebagai string, jadi ubah nam menjadi nmk untuk melihat hasil akhir nilai mata kuliahnya.

- **Gunakan if .. else if .. else**

  Agar jika sudah satu kondisi yang terpenuhi dan sesuai maka kondisi lainnya diabaikan.

- **Pastikan nmk terisi**

  Dikarenakan nmk adalh tempat menyimpan huruf nilai (A,B,C,D dst). Program itu membaca nilai angka(nam) lalu menentukan hurufnya (nmk). Kalau nmk tidak pernah terisi, berarti program tidak pernah menentukan huruf nilainya,

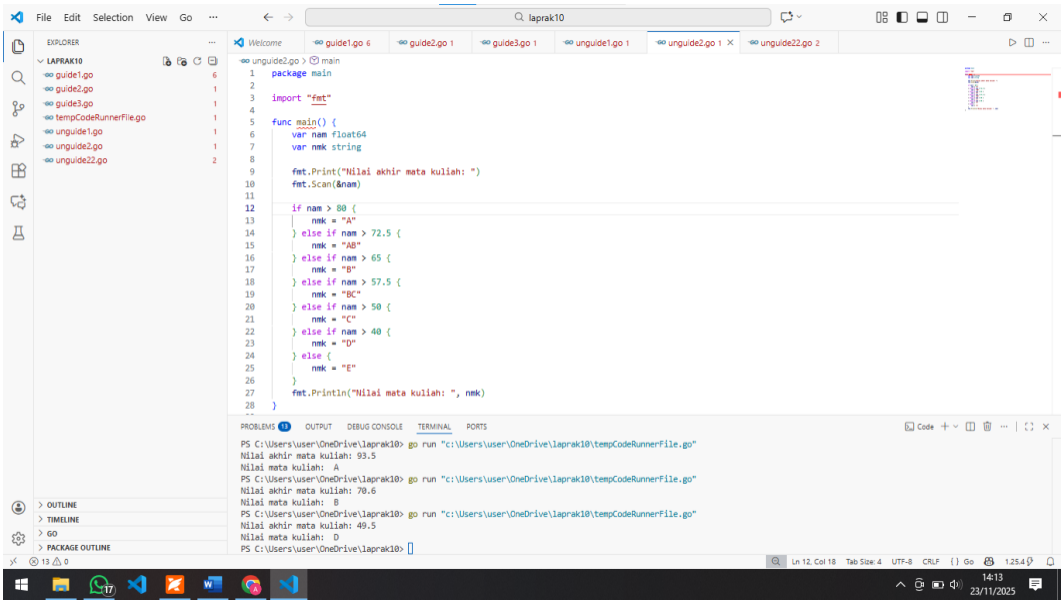
- **Gunakan Kutip ASCII “ “**

  Seluruh teks output dan string di Go harus memakai tanda kutip biasa, bukan kutip miring atau curly.

- **Atur interval nilai dari tertinggi ke terendah**

  Periksa nilai mulai dari yang terbesar, sehingga nilai batas akan jauh ke kategori yang benar tanpa bentrok.

3. Perbaikan kode program serta keluaran yang diberikan

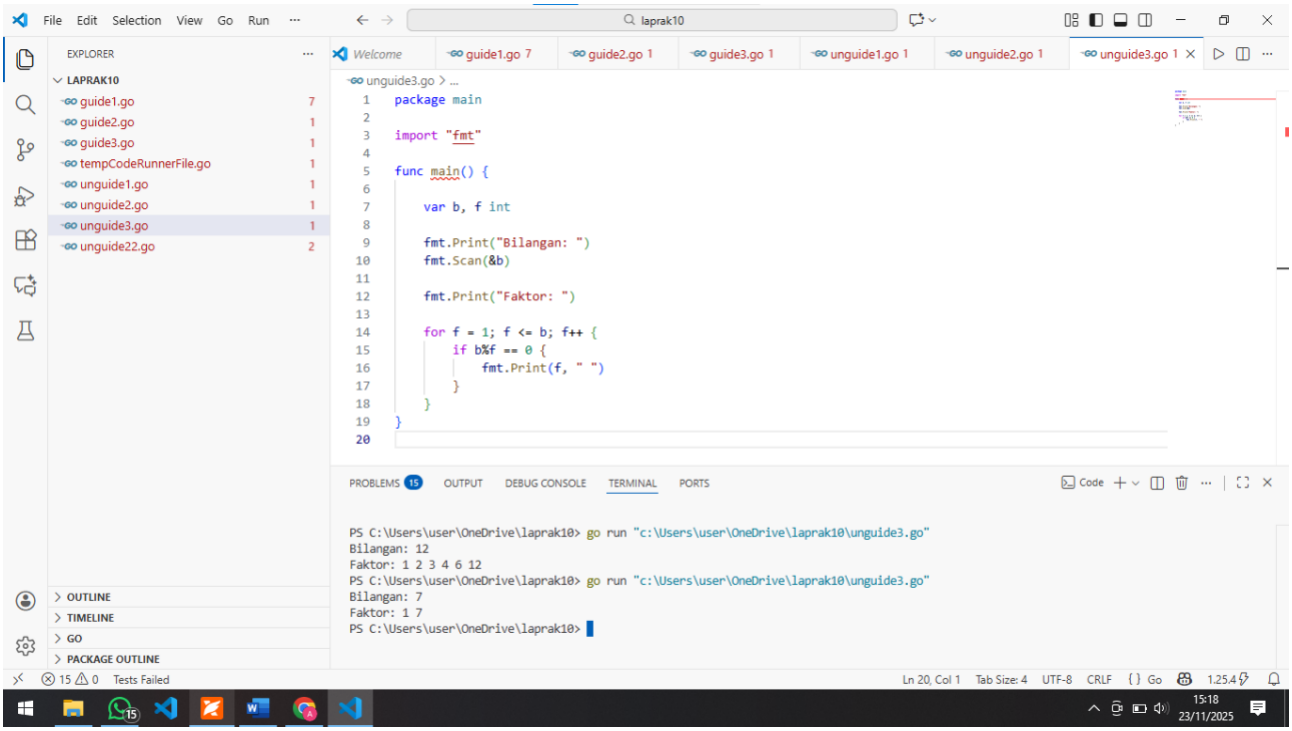


Tugas 3

Source code (Kode Sebelum : hanya mencari faktor)

```
package main

import "fmt"

func main() {

    var b, f int

    fmt.Print("Bilangan: ")
    fmt.Scan(&b)

    fmt.Print("Faktor: ")

    for f = 1; f <= b; f++ {
        if b%f == 0 {
            fmt.Print(f, " ")
        }
    }
}
```

Screenshoot program



Deskripsi program

- **package main**

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

- **import "fmt"**

Berfungsi supaya program bisa membaca input dari keyboard menggunakan Scan dan menampilkan output ke layar menggunakan Print atau Println. Tanpa mengimpor fmt, perintah input-output seperti fmt.Scan atau fmt.Println tidak dapat digunakan. Paket fmt merupakan singkatan dari format, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

- **func main()**

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layer. Tanpa main(), program tidak akan bisa dijalankan.

- **Tanda buka dan tutup kurung kurawal { }**

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main.

- **var b, f int**

Baris ini mendeklarasikan dua variabel bertipe integer, yaitu b dan f.

– b digunakan untuk menyimpan bilangan yang dimasukkan.

– f digunakan sebagai variabel penghitung dalam perulangan yang memeriksa setiap angka dari 1 hingga bilangan b.

Variabel-variabel ini diperlukan agar program dapat menjalankan proses pengecekan faktor.

- **fmt.Print("Bilangan: ")**

Baris ini menampilkan teks ke layar sebagai instruksi bagi pengguna untuk memasukkan sebuah bilangan. Fungsi Print digunakan karena tidak otomatis menambahkan baris baru, sehingga kursor tetap berada di baris yang sama dan pengguna langsung dapat mengetikkan input.

- **fmt.Scan(&b)**

Perintah ini membaca input angka dan menyimpannya ke dalam variabel b. Tanda & digunakan karena fmt.Scan membutuhkan alamat memori tempat nilai input akan disimpan. Dengan demikian, variabel b akan berisi bilangan yang akan dicari faktor-faktornya.

- **fmt.Print("Faktor: ")**

Baris ini menampilkan teks untuk memberi tahu bahwa program akan menampilkan daftar faktor dari bilangan yang dimasukkan. Teks ini dicetak sebelum proses perulangan dimulai

- **for f = 1; f <= b; f++ { … }**

Ini adalah struktur perulangan for dalam bahasa Go yang digunakan untuk memeriksa setiap bilangan dari 1 sampai b (bilangan yang dimasukkan pengguna).

– f = 1 berarti perulangan dimulai dari angka 1.

– f <= b menjadi kondisi perulangan: selama nilai f masih lebih kecil atau sama dengan b, perulangan tetap berjalan.

– f++ berarti nilai f bertambah 1 setiap kali perulangan selesai berjalan satu putaran.

Dengan cara ini, program secara sistematis mengecek seluruh bilangan dari 1 hingga b untuk melihat apakah bilangan itu merupakan faktor.

- **if b%f == 0 { … }**

Dalam setiap iterasi perulangan, program menjalankan percabangan ini untuk mengecek apakah f adalah faktor dari b.

Operator modulo (%) digunakan untuk mendapatkan sisa hasil bagi. Jika b%f == 0, berarti b habis dibagi oleh f, dan f adalah faktor dari b. Hanya bilangan yang memenuhi kondisi ini yang akan dicetak sebagai faktor**.**

- **fmt.Print(f, " ")**

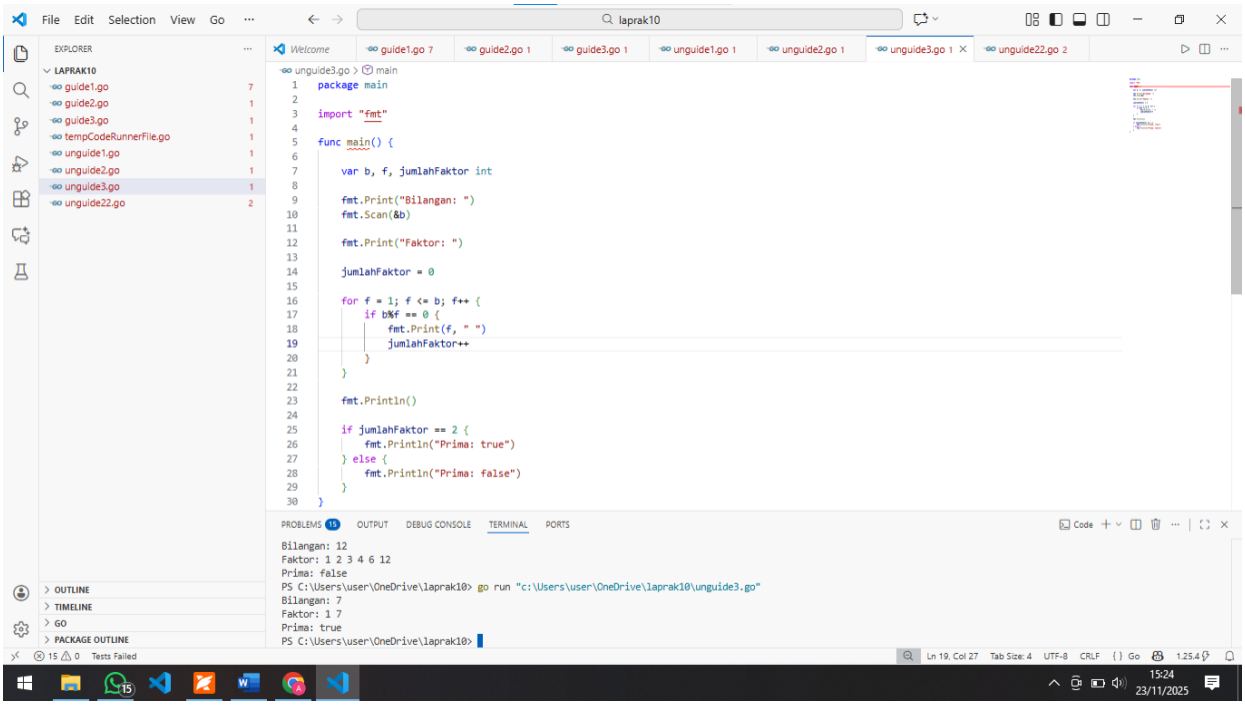
Jika kondisi pada percabangan terpenuhi, baris ini mencetak nilai f ke layar, diikuti dengan spasi agar faktor-faktor yang dicetak tampil berjajar secara rapi dalam satu baris. Setiap faktor akan ditampilkan sesuai urutan pengecekan dari 1 hingga b.

Tugas 3

Source code (Kode Sesudah: mencari faktor ditambah cek prima)

```
package main

import "fmt"

func main() {

	var b, f, jumlahFaktor int

	fmt.Print("Bilangan: ")
	fmt.Scan(&b)

	fmt.Print("Faktor: ")

	jumlahFaktor = 0

	for f = 1; f <= b; f++ {
		if b%f == 0 {
			fmt.Print(f, " ")
			jumlahFaktor++
		}
	}

	fmt.Println()

	if jumlahFaktor == 2 {
		fmt.Println("Prima: true")
	} else {
		fmt.Println("Prima: false")
	}
}
```

Screenshoot Program



Deskripsi program

- **package main**

Biasanya digunakan untuk menandai bahwa file tersebut adalah program utama, bagian dari kode yang pertama kali dijalankan untuk menjalankan program Go. Package ini digunakan bersamaan dengan fungsi main() sebagai titik awal berjalannya program. Tanpa package main, program tidak bisa dijalankan sebagai aplikasi.

- **import "fmt"**

Berfungsi supaya program bisa membaca input dari keyboard menggunakan Scan dan menampilkan output ke layar menggunakan Print atau Println. Tanpa mengimpor fmt, perintah input-output seperti fmt.Scan atau fmt.Println tidak dapat digunakan. Paket fmt merupakan singkatan dari format, karena digunakan untuk memformat teks, angka, dan data yang ditampilkan.

- **func main()**

Sebagai panggung utama dari program Go. Semua proses utama dijalankan di dalam fungsi ini. Berfungsi untuk mengatur alur kerja program, yaitu mulai dari membaca input, memproses data, hingga menampilkan hasil ke layer. Tanpa main(), program tidak akan bisa dijalankan.

- **Tanda buka dan tutup kurung kurawal { }**

Artinya kode utama program dimulai dan diakhiri di dalam kurung kurawal tersebut. Go menggunakan kurung kurawal untuk menandai batas awal dan akhir blok kode. Jika kurung ini tidak ada, maka program tidak tahu batas instruksi mana saja yang termasuk dalam fungsi main.

- **var b, f, jumlahFaktor int**

Baris ini mendeklarasikan tiga variabel bertipe integer:

– b digunakan untuk menyimpan bilangan yang dimasukkan .

– f digunakan sebagai variabel penghitung dalam perulangan yang memeriksa setiap kemungkinan faktor.

– jumlahFaktor digunakan untuk mencatat berapa banyak faktor yang dimiliki oleh bilangan b. Variabel ini nantinya menjadi dasar penentuan apakah bilangan tersebut prima atau tidak.

- **fmt.Print("Bilangan: ")**

Baris ini mencetak teks ke layar sebagai instruksi agar memasukkan sebuah bilangan. Fungsi Print digunakan sehingga kursor tetap berada pada baris yang sama dengan teks instruksi.

- **fmt.Scan(&b)**

Digunakan untuk membaca bilangan yang dimasukkan melalui keyboard dan menyimpannya ke variabel b. Tanda & menunjukkan alamat memori variabel b sehingga nilai input langsung disimpan pada variabel tersebut.

- **fmt.Print("Faktor: ")**

Baris ini memberi tahu bahwa program akan menampilkan daftar faktor dari bilangan yang dimasukkan. Teks ini dicetak sebelum proses pencarian faktor dimulai.

- **jumlahFaktor = 0**

Variabel jumlahFaktor diinisialisasikan dengan nilai 0. Nilai ini akan bertambah setiap kali program menemukan sebuah bilangan yang merupakan faktor dari b. Dengan demikian, variabel ini berfungsi sebagai penghitung faktor.

- **for f = 1; f <= b; f++ { … }**

Perintah for ini digunakan untuk melakukan perulangan dari angka 1 hingga angka b.

– Perulangan dimulai dengan f bernilai 1.

– Perulangan terus berjalan selama f kurang dari atau sama dengan b.

– Pada setiap iterasi, nilai f ditambah 1 (f++).

Dengan cara ini, program mengecek semua calon faktor dari bilangan 1 sampai bilangan b.

- **if b%f == 0 { … }**

Pada setiap putaran perulangan, program mengecek apakah f adalah faktor dari b.

Operator modulo (%) digunakan untuk memeriksa sisa pembagian. Jika sisa pembagian b%f bernilai 0, maka f adalah faktor dari b. Bila kondisi ini terpenuhi, program mengeksekusi isi blok if.

- **fmt.Print(f, " ")**

Jika f terbukti menjadi faktor dari b, baris ini akan mencetak f ke layar, diikuti dengan spasi untuk memisahkan faktor-faktor lainnya. Semua faktor ditampilkan dalam satu baris agar mudah dibaca.

- **jumlahFaktor++**

Setiap kali ditemukan faktor baru, variabel jumlahFaktor ditambah 1. Dengan demikian, pada akhir perulangan, jumlahFaktor akan berisi total faktor yang dimiliki bilangan b, yang nantinya digunakan untuk menentukan apakah bilangan tersebut prima.

- **fmt.Println()**

Setelah seluruh faktor dicetak, baris ini mencetak baris baru agar tampilan hasil penentuan prima berada di baris terpisah dan lebih rapi dibaca.

- **if jumlahFaktor == 2 { … } else { … }**

Percabangan ini menentukan apakah bilangan b adalah bilangan prima. Suatu bilangan dikatakan prima jika dan hanya jika memiliki tepat dua faktor, yaitu 1 dan dirinya sendiri.

– Jika jumlahFaktor bernilai 2, program menampilkan “Prima: true”.

– Jika jumlahFaktor tidak sama dengan 2, program menampilkan “Prima: false”.

- **fmt.Println("Prima: true") / fmt.Println("Prima: false")**

Baris terakhir ini menampilkan hasil akhir, yaitu status apakah bilangan tersebut merupakan bilangan prima atau bukan. Output dituliskan secara jelas agar pengguna mengetahui hasil evaluasi program.