

**LAPORAN PRAKTIKUM**  
**ALGORITMA DAN PEMROGRAMAN 1**  
**MODUL 4**  
**“I/O, TIPE DATA & VARIABEL (latihan 2)”**



**DISUSUN OLEH:**  
**DAFFA TSAQIFNA FAUZTSANY**  
**103112400032**  
**S1 IF-12-01**

**DOSEN:**  
**Yohani Setiya Rafika Nur, M. Kom.**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2024/2025**

## DASAR TEORI:

Pemrograman bisa diibaratkan sebagai sebuah alat yang membantu kita memecahkan berbagai masalah dengan cara yang lebih efisien dan terstruktur. Di dalam dunia pemrograman, ada beberapa konsep dasar yang penting untuk dipahami agar program bisa berjalan dengan baik, yaitu Input/Output (I/O), tipe data, dan variabel. Ketiga hal ini merupakan komponen dasar yang memungkinkan program menerima, mengolah, dan memberikan data kepada pengguna.

### Input/Output (I/O)

Bayangkan program seperti manusia yang berkomunikasi dengan dunia luar. Input adalah informasi yang diberikan kepada program, seperti saat kita memasukkan angka di kalkulator. Setelah itu, program akan memproses informasi tersebut, lalu memberikan hasilnya kepada kita, yang disebut sebagai output. Dalam pemrograman, proses ini sangat penting untuk memastikan program dapat "mendengar" apa yang kita minta dan "berbicara" kembali dengan memberikan hasil yang sesuai.

Misalnya, ketika kita meminta program menghitung luas segitiga, kita memasukkan nilai alas dan tinggi (input), kemudian program akan mengolahnya menggunakan rumus matematika, lalu hasilnya akan ditampilkan di layar (output).

### Tipe Data

Tipe data bisa diibaratkan seperti kategori atau label yang kita berikan pada informasi yang ingin disimpan oleh program. Setiap jenis data memiliki caranya sendiri untuk diproses, seperti angka bulat (integer), angka desimal (float), teks (string), atau nilai benar-salah (boolean).

Misalnya, jika kita ingin menyimpan angka 3,14 (nilai desimal), kita harus memastikan bahwa variabel yang menampungnya diatur sebagai tipe data float, karena jika diatur sebagai integer, maka nilai desimalnya akan diabaikan. Memahami tipe data sangat penting agar program bisa menangani informasi dengan benar dan tidak membuat kesalahan.

### Variabel

Dalam kehidupan sehari-hari, variabel seperti kotak penyimpanan tempat kita menaruh barang. Di dalam pemrograman, variabel adalah wadah yang menyimpan data yang bisa berubah-ubah sesuai kebutuhan program. Misalnya, kita bisa membuat variabel untuk menyimpan panjang sisi kubus, lalu menggunakan variabel ini untuk menghitung volume kubus tersebut.

Variabel membantu program menyimpan dan mengelola informasi dengan mudah. Ketika kita memberi nama pada variabel, kita seperti memberi label pada kotak tersebut agar mudah diakses kembali saat dibutuhkan.

### Operasi Integer Division dan Modulo

Operasi integer division adalah pembagian yang selalu membulatkan hasilnya ke bawah, sehingga hasilnya adalah bilangan bulat. Sebaliknya, \*modulo\* adalah operasi yang memberi tahu kita sisa hasil pembagian. Contohnya, jika kita membagi 10 dengan 3, hasil pembagiannya adalah 3 (integer division), dan sisanya adalah 1 (modulo). Kedua operasi ini sangat berguna, misalnya saat kita ingin memeriksa apakah sebuah angka genap atau ganjil.

## Konversi Tipe Data atau Casting

Terkadang, dalam suatu program, kita butuh mengubah jenis data dari satu bentuk ke bentuk lain, yang dikenal sebagai casting. Misalnya, jika kita memiliki angka desimal seperti 3,14, tetapi hanya ingin mengambil nilai bulatnya, kita bisa mengonversi (mengubah) tipe datanya dari float menjadi integer. Ini sangat berguna saat program membutuhkan jenis data tertentu untuk melakukan perhitungan atau operasi lainnya.

## CONTOH SOAL

Conso 1:

```
package main

import "fmt"

func main() {
    var s, h, m int
    fmt.Scan(&s)
    h = s / 3600
    m = (s % 3600) / 60
    s = s % 60
    fmt.Print(h, " jam ", m, " menit ", s, " dekit")
}
```

Output:

```
PS D:\test bs> go run 'd:\test bs\3rd meet\conso3\conso3-1.go'
3661
1 jam 1 menit 1 dekit
PS D:\test bs> go run 'd:\test bs\3rd meet\conso3\conso3-1.go'
7322
2 jam 2 menit 2 dekit
PS D:\test bs> go run 'd:\test bs\3rd meet\conso3\conso3-1.go'
3600
1 jam 0 menit 0 dekit
```

Deskripsi Program:

Program Go di atas digunakan untuk mengonversi jumlah detik yang dimasukkan oleh pengguna menjadi format waktu dalam **jam**, **menit**, dan **detik**. Berikut adalah penjelasan rinci mengenai setiap bagian dari program:

### 1. Deklarasi variabel:

```
var s, h, m int
```

Program mendeklarasikan tiga variabel bertipe `int` (bilangan bulat):

- `s` untuk menyimpan jumlah detik yang diinputkan oleh pengguna.
- `h` untuk menyimpan hasil perhitungan jumlah jam.
- `m` untuk menyimpan hasil perhitungan jumlah menit.

### 2. Membaca input dari pengguna:

```
fmt.Scan(&s)
```

- Program menggunakan `fmt.Scan(&s)` untuk membaca jumlah detik yang dimasukkan oleh pengguna dan menyimpannya ke dalam variabel `s`.

3. Menghitung jumlah jam, menit, dan detik:

```
h = s / 3600
m = (s % 3600) / 60
s = s % 60
```

- Program membagi total detik s dengan 3600 (jumlah detik dalam satu jam) untuk menghitung jumlah jam. Misalnya, jika pengguna memasukkan 5000 detik, maka  $5000 / 3600$  akan menghasilkan 1 jam.
- Program menghitung sisa detik setelah dibagi 3600 menggunakan operator modulus ( $s \% 3600$ ) untuk mendapatkan detik yang tersisa setelah jam dihitung. Kemudian, program membagi sisa tersebut dengan 60 untuk menghitung jumlah menit.
- Misalnya, jika 5000 detik, sisa detik setelah dibagi 3600 adalah 1400 detik.  $1400 / 60$  menghasilkan 23 menit.
- Program menghitung sisa detik yang tidak bisa dibagi menjadi menit dengan menggunakan operator modulus ( $s \% 60$ ). Ini memberikan detik yang tersisa setelah jam dan menit dihitung.
- Misalnya, dari 1400 detik, sisa detik setelah dihitung menit adalah  $1400 \% 60$ , yang memberikan 20 detik.

4. Menampilkan hasil:

```
fmt.Print(h, " jam ", m, " menit ", s, " dekit")
```

- Program mencetak hasil konversi waktu dalam format jam, menit, dan detik, dengan menampilkan variabel h, m, dan s.

Contoh: Jika pengguna memasukkan  $s = 5000$ , maka perhitungan pada program adalah sebagai berikut:

```
5000(input)
1 jam 23 menit 20 dekit(output)
```

2.cosol2

```
package main
import "fmt"
func main() {
    var x, d1, d2, d3 int
    fmt.Scan(&x)
    d1 = x / 100
    d2 = x % 100 / 10
    d3 = x % 10
    fmt.Print(d1 <= d2 && d2 <= d3)
}
```

Output:

```

PS D:\test bs> go run 'd:\test bs\3rd meet\conso3\conso3-2.go'
362
false
PS D:\test bs> go run 'd:\test bs\3rd meet\conso3\conso3-2.go'
256
true
PS D:\test bs> go run 'd:\test bs\3rd meet\conso3\conso3-2.go'
189
true

```

#### Deskripsi Program:

Program Go di atas adalah program sederhana yang memeriksa apakah tiga digit dari suatu angka yang diinputkan oleh pengguna berada dalam urutan yang **menaik** (dari kecil ke besar). Berikut adalah penjelasan rinci mengenai setiap bagian dari program:

##### 1. Deklarasi variabel:

```
var x, d1, d2, d3 int
```

Program mendeklarasikan empat variabel bertipe `int` (bilangan bulat):

- `x`: untuk menyimpan bilangan tiga digit yang diinputkan oleh pengguna.
- `d1`: untuk menyimpan digit pertama (ratusan) dari bilangan `x`.
- `d2`: untuk menyimpan digit kedua (puluhan) dari bilangan `x`.
- `d3`: untuk menyimpan digit ketiga (satuan) dari bilangan `x`.

##### 2. Membaca input dari pengguna:

```
fmt.Scan(&x)
```

- Program menggunakan `fmt.Scan(&x)` untuk membaca input bilangan bulat `x` yang diberikan oleh pengguna. Input yang diminta adalah bilangan tiga digit.

##### 3. Memisahkan digit-digit bilangan:

```

d1 = x / 100
d2 = x % 100 / 10
d3 = x % 10

```

- Program membagi `x` dengan 100 untuk mendapatkan digit pertama (ratusan). Misalnya, jika `x = 123`, maka `123 / 100` akan menghasilkan 1.
- Program menggunakan operator modulus (`x % 100`) untuk menghilangkan digit pertama, sehingga hasilnya adalah sisa dari dua digit terakhir (misalnya `123 % 100` menghasilkan 23), kemudian membagi hasil tersebut dengan 10 untuk mendapatkan digit kedua (puluhan). Dalam contoh ini, `23 / 10` akan menghasilkan 2.
- Program menggunakan operator modulus (`x % 10`) untuk mendapatkan digit terakhir (satuan). Misalnya, `123 % 10` menghasilkan 3.

##### 4. Memeriksa urutan digit:

```
fmt.Print(d1 <= d2 && d2 <= d3)
```

- Program memeriksa apakah digit pertama (`d1`) lebih kecil atau sama dengan digit kedua (`d2`), dan digit kedua lebih kecil atau sama dengan digit ketiga (`d3`).

- Kondisi  $d1 \leq d2 \ \&\& \ d2 \leq d3$  akan mengembalikan nilai true jika digit-digit tersebut dalam urutan menaik atau sama, dan false jika tidak.

Contoh: Jika pengguna memasukkan  $x = 123$ , maka perhitungan pada program adalah sebagai berikut:

123(input)
true(output)

### 3.cosol3

```
package main
import "fmt"
func main() {
    var x, y, z float64
    fmt.Print("masukan berat badan (kg): ")
    fmt.Scan(&x)
    fmt.Print("masukan tinggi badan (m): ")
    fmt.Scan(&y)
    z = x / (y * y)
    fmt.Printf("BMI anda: %.2f", z)
}
```

Output:

```
PS D:\test bs> go run 'd:\test bs\3rd meet\conso3\conso3-3.go'
masukan berat badan (kg): 70
masukan tinggi badan (m): 1.75
BMI anda: 22.86
PS D:\test bs> go run 'd:\test bs\3rd meet\conso3\conso3-3.go'
masukan berat badan (kg): 60
masukan tinggi badan (m): 1.6
BMI anda: 23.44
PS D:\test bs> go run 'd:\test bs\3rd meet\conso3\conso3-3.go'
masukan berat badan (kg): 80
masukan tinggi badan (m): 1.8
BMI anda: 24.69
```

### Deskripsi Program:

Program Go di atas digunakan untuk menghitung **BMI (Body Mass Index)** berdasarkan berat badan dan tinggi badan yang dimasukkan oleh pengguna. Berikut adalah penjelasan rinci mengenai setiap bagian dari program:

#### 1. Deklarasi variabel:

```
var x, y, z float64
```

Program mendeklarasikan tiga variabel bertipe `float64` (bilangan desimal):

- `x`: untuk menyimpan berat badan pengguna dalam kilogram (kg).
- `y`: untuk menyimpan tinggi badan pengguna dalam meter (m).
- `z`: untuk menyimpan hasil perhitungan BMI.

2. Membaca input dari pengguna:

```
fmt.Print("masukan berat badan (kg): ")
fmt.Scan(&x)
fmt.Print("masukan tinggi badan (m): ")
fmt.Scan(&y)
```

- Program menampilkan pesan **"masukan berat badan (kg): "** untuk meminta pengguna memasukkan berat badan dalam kilogram.
- `fmt.Scan(&x)` digunakan untuk membaca input berat badan dari pengguna dan menyimpannya ke dalam variabel `x`.
- Program menampilkan pesan **"masukan tinggi badan (m): "** untuk meminta pengguna memasukkan tinggi badan dalam meter.
- `fmt.Scan(&y)` digunakan untuk membaca input tinggi badan dari pengguna dan menyimpannya ke dalam variabel `y`.

3. Menghitung BMI:

```
z = x / (y * y)
```

- **BMI** (Body Mass Index) dihitung menggunakan rumus: BMI= Berat Badan (kg)/Tinggi Badan (m)<sup>2</sup>
- Program mengambil berat badan (`x`) dan membaginya dengan kuadrat dari tinggi badan (`y * y`).

4. Menampilkan hasil:

```
fmt.Printf("BMI anda: %.2f", z)
```

- Program menampilkan hasil perhitungan BMI dengan format **"BMI anda: [nilai z]"**, di mana nilai BMI ditampilkan dengan dua angka di belakang koma (`%.2f`).
- Misalnya, jika hasil BMI adalah 22.3456, maka program akan menampilkan BMI anda: 22.35.

Contoh: jika pengguna memasukkan:

```
masukan berat badan (kg): 70(input)
masukan tinggi badan (m): 1.75(input)
BMI anda: 22.86(output)
```



## SOAL LATIHAN

### 1. Latihan1

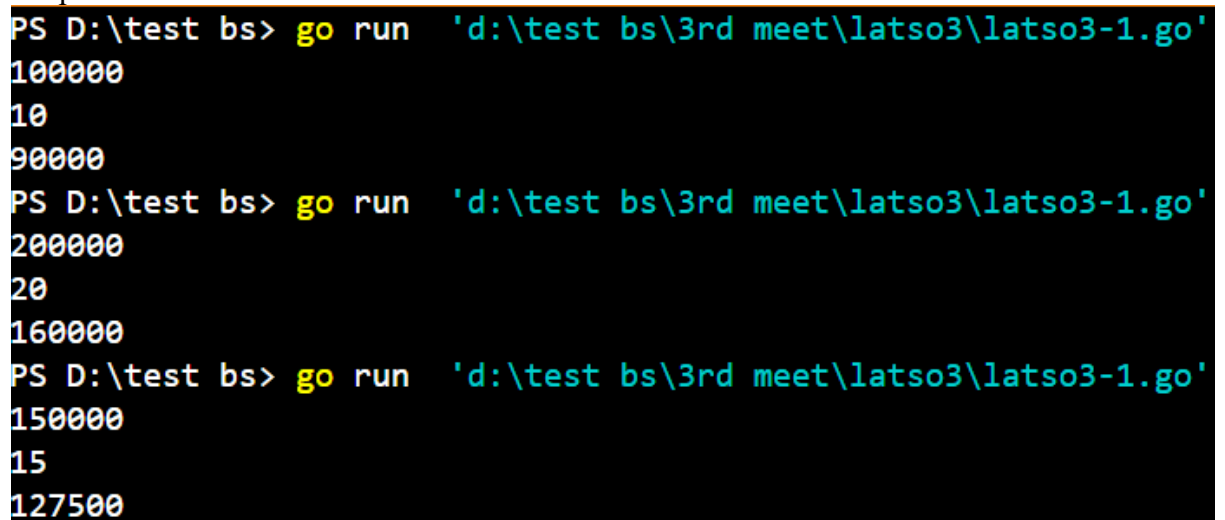
Source Code:

```
package main

import "fmt"

func main() {
    var x, y, z int
    fmt.Scan(&x, &y)
    z = x - (x * y / 100)
    fmt.Print(z)
}
```

Output:



```
PS D:\test bs> go run 'd:\test bs\3rd meet\latso3\latso3-1.go'
100000
10
90000
PS D:\test bs> go run 'd:\test bs\3rd meet\latso3\latso3-1.go'
200000
20
160000
PS D:\test bs> go run 'd:\test bs\3rd meet\latso3\latso3-1.go'
150000
15
127500
```

Deskripsi Program:

Program Go di atas digunakan untuk menghitung nilai setelah dikurangi persentase diskon dari nilai awal. Program ini mengambil dua input dari pengguna: **nilai awal** dan **persentase diskon**, lalu menghitung nilai akhir setelah diskon diterapkan. Berikut adalah penjelasan rinci mengenai setiap bagian dari program:

#### 1. Deklarasi variabel:

```
var x, y, z int
```

Program mendeklarasikan tiga variabel bertipe `int` (bilangan bulat):

- `x`: untuk menyimpan nilai awal (sebelum diskon).
- `y`: untuk menyimpan persentase diskon.
- `z`: untuk menyimpan hasil akhir setelah diskon diterapkan.

#### 2. Membaca input dari pengguna:

```
fmt.Scan(&x, &y)
```

Program menggunakan `fmt.Scan(&x, &y)` untuk membaca dua input dari pengguna:

- **x**: nilai awal sebelum diskon.
- **y**: persentase diskon yang akan diterapkan.

3. Menghitung nilai setelah diskon:

```
z = x - (x * y / 100)
```

- Program menghitung nilai akhir setelah diskon dengan rumus:  $z = x - (x \times y / 100)$
- $x * y / 100$  adalah jumlah diskon yang dihitung sebagai persentase dari nilai awal (x).
- $x - (x * y / 100)$  mengurangi nilai diskon dari nilai awal, sehingga menghasilkan nilai akhir setelah diskon.

4. Menampilkan hasil:

```
fmt.Print(z)
```

- Program mencetak hasil perhitungan, yaitu nilai akhir setelah diskon diterapkan (z).

Contoh: Jika pengguna memasukkan nilai awal  $x = 1000$  dan persentase diskon  $y = 20$ , maka perhitungannya adalah sebagai berikut:

```
1000(input)
```

```
20(input)
```

```
800(output)
```

## 2. .Latihan2

### Source Code:

```
package main

import "fmt"

func main() {
    var x, y, z float64
    fmt.Scan(&x, &y)
    z = x * y * y
    fmt.Printf("%.0f", z)
}
```

Output:

```

PS D:\test bs> go run 'd:\test bs\3rd meet\latso3\latso3-2.go'
22.85 1.75
70
PS D:\test bs> go run 'd:\test bs\3rd meet\latso3\latso3-2.go'
23.43 1.6
60
PS D:\test bs> go run 'd:\test bs\3rd meet\latso3\latso3-2.go'
24.69 1.8
80

```

Deskripsi Program:

Program Go di atas digunakan untuk menghitung hasil perkalian antara dua nilai yang dimasukkan oleh pengguna, di mana salah satu nilai dipangkatkan dua terlebih dahulu sebelum dikalikan dengan nilai lainnya. Berikut adalah penjelasan rinci mengenai setiap bagian dari program:

#### 1. Deklarasi variabel:

```
var x, y, z float64
```

Program mendeklarasikan tiga variabel bertipe float64 (bilangan desimal):

- x: untuk menyimpan nilai pertama yang dimasukkan oleh pengguna.
- y: untuk menyimpan nilai kedua yang dimasukkan oleh pengguna.
- z: untuk menyimpan hasil akhir dari perhitungan.

#### 2. Membaca input dari pengguna:

```
fmt.Scan(&x, &y)
```

Program menggunakan `fmt.Scan(&x, &y)` untuk membaca dua input dari pengguna:

- x: nilai BMI.
- y: nilai Tinggi.

#### 3. Menghitung hasil:

```
z = x * y * y
```

Program menghitung hasil perhitungan dengan rumus:  $z = x \times y^2$

- Nilai kedua (y) dipangkatkan dua terlebih dahulu ( $y * y$ ).
- Hasil pangkat dua dari y kemudian dikalikan dengan nilai pertama (x).
- Hasil ini disimpan ke dalam variabel z.

#### 4. Menampilkan hasil:

```
fmt.Printf("%.0f", z)
```

- Program menggunakan `fmt.Printf` untuk mencetak hasil (z) dalam format **bilangan bulat** tanpa desimal (`%.0f`).
- Meskipun variabel z adalah tipe float64, format `%.0f` digunakan untuk menampilkan hasil dalam bentuk bilangan bulat dengan pembulatan otomatis (menghilangkan angka desimal).

Contoh: Jika pengguna memasukkan:

```
x = 2(input)
```

y = 3(input)

18(output)

### 3. Latihan3

#### Source Code:

```
package main

import (
    "fmt"
    "math"
)

func distance(x1, y1, x2, y2 float64) float64 {
    return math.Sqrt((x2-x1)*(x2-x1) + (y2-y1)*(y2-y1))
}

func main() {
    var ax, ay, bx, by, cx, cy float64
    fmt.Print("masukan koordinat titik A: ")
    fmt.Scan(&ax, &ay)
    fmt.Print("masukan koordinat titik B: ")
    fmt.Scan(&bx, &by)
    fmt.Print("masukan koordinat titik C: ")
    fmt.Scan(&cx, &cy)
    AB := distance(ax, ay, bx, by)
    BC := distance(bx, by, cx, cy)
    CA := distance(cx, cy, ax, ay)
    maxSide := math.Max(AB, math.Max(BC, CA))
    fmt.Printf("%.2f\n", maxSide)
}
```

Output:

```
PS D:\test bs> go run 'd:\test bs\3rd meet\latso3\latso3-3.go'
masukan koordinat titik A: 1.0 1.0
masukan koordinat titik B: 4.0 1.0
masukan koordinat titik C: 1.0 5.0
5.00
PS D:\test bs> go run 'd:\test bs\3rd meet\latso3\latso3-3.go'
masukan koordinat titik A: 0.0 0.0
masukan koordinat titik B: 3.0 0.0
masukan koordinat titik C: 3.0 4.0
5.00
```

## Deskripsi Program:

Program Go di atas digunakan untuk menghitung jarak antara tiga titik dalam sistem koordinat 2D, serta menentukan sisi terpanjang dari segitiga yang terbentuk oleh tiga titik tersebut. Program ini menggunakan fungsi untuk menghitung jarak antar dua titik dan memanfaatkan fungsi dari paket `math`. Berikut adalah penjelasan rinci mengenai setiap bagian dari program:

### 1. Paket yang diimpor:

```
import (  
    "fmt"  
    "math"  
)
```

- **fmt**: digunakan untuk input/output, seperti membaca nilai dari pengguna dan mencetak hasil.
- **math**: digunakan untuk perhitungan matematika, khususnya fungsi `math.Sqrt` (akar kuadrat) dan `math.Max` (nilai maksimum).

### 2. Fungsi distance:

```
func distance(x1, y1, x2, y2 float64) float64 {  
    return math.Sqrt((x2-x1)*(x2-x1) + (y2-y1)*(y2-y1))  
}
```

- **Tujuan**: Fungsi ini menghitung jarak Euclidean antara dua titik  $(x_1, y_1)$  dan  $(x_2, y_2)$  dalam sistem koordinat 2D.
- **Rumus yang digunakan** adalah rumus jarak Euclidean:  
$$\text{distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
- `math.Sqrt` digunakan untuk menghitung akar kuadrat, dan  $(x_2 - x_1)^2 + (y_2 - y_1)^2$  menghitung kuadrat dari perbedaan kedua titik pada sumbu x dan y.

### 3. Bagian utama (fungsi main):

#### a. Deklarasi variabel

```
var ax, ay, bx, by, cx, cy float64
```

Variabel ini digunakan untuk menyimpan koordinat tiga titik:

- Titik **A**:  $(ax, ay)$
- Titik **B**:  $(bx, by)$
- Titik **C**:  $(cx, cy)$

#### b. Input dari pengguna

```
var ax, ay, bx, by, cx, cy float64
```

- Program meminta pengguna memasukkan tiga pasang nilai (koordinat x dan y) untuk tiga titik. Format input menggunakan `%f` untuk membaca nilai bertipe `float64`.

#### c. Menghitung jarak antar titik

```
var ax, ay, bx, by, cx, cy float64
```

Program memanggil fungsi `distance` tiga kali untuk menghitung jarak antara:

- **AB**: jarak antara titik A dan B.
- **BC**: jarak antara titik B dan C.
- **CA**: jarak antara titik C dan A.

d. Menentukan sisi terpanjang

```
var ax, ay, bx, by, cx, cy float64
```

Fungsi `math.Max` digunakan untuk menemukan nilai maksimum dari tiga sisi yang dihitung:

- Mula-mula, `math.Max(BC, CA)` mencari nilai terbesar antara **BC** dan **CA**.
- Kemudian, nilai terbesar tersebut dibandingkan lagi dengan **AB** untuk menemukan sisi terpanjang dari segitiga.

e. Menampilkan hasil

```
var ax, ay, bx, by, cx, cy float64
```

- Program mencetak nilai dari sisi terpanjang dengan dua angka di belakang koma (`%.2f`).

Contoh: Jika pengguna memasukkan:

```
1.0 1.0(input)
4.0 1.0(input)
1.0 5.0(input)
5.00(output)
```