

**LAPORAN PRAKTIKUM ALGORITMA  
DAN PEMROGRAMAN 1**

**MODUL 5 & 6 – FOOR-LOOP  
ALGORITMA DAN PEMOGRAMAN 1**



**Disusun oleh:**

**NAMA : PRIMATAMA SIGALINGGING**

**NIM : 109082500076**

**S1IF-13-07**

**Asisten Praktikum**

**Adithana dharma putra**

**Apri pandu wicaksono**

**PROGRAM STUDI S1 INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2025**

## LATIHAN KELAS – GUIDED

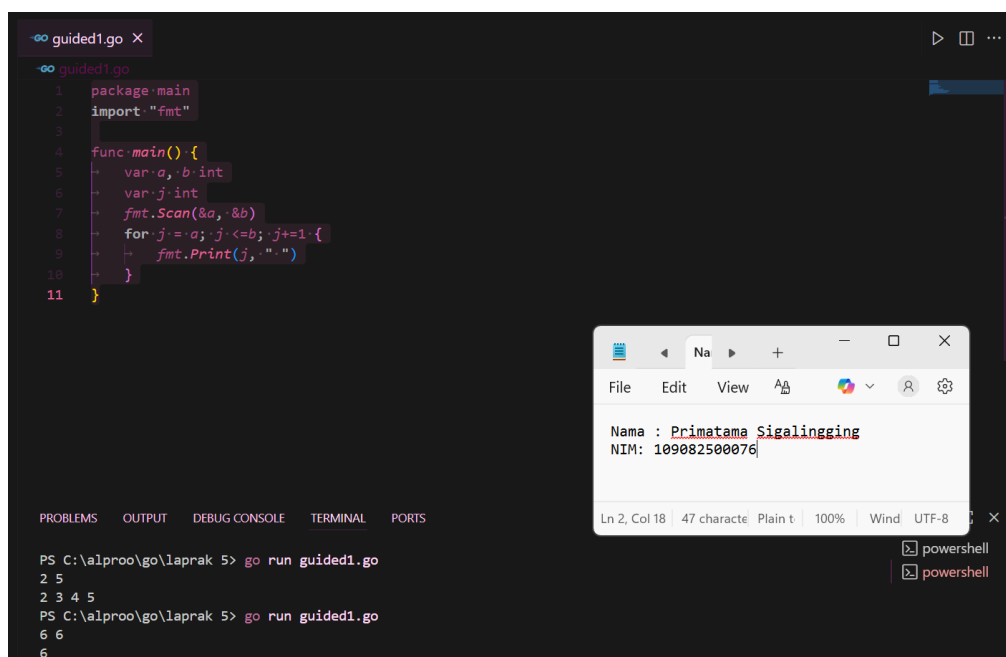
### 1. Guided 1 Source Code

```
package main

import "fmt"

func main() {
    var a, b int
    var j int
    fmt.Scan(&a, &b)
    for j = a; j <=b; j+=1 {
        fmt.Print(j, " ")
    }
}
```

### Screenshoot program



### Deskripsi program

Jelaskan kode yang ada di source code, semakin detal semakin baik nilainya

## 2. Guided 2

### Source Code

```
package main

import "fmt"

func main() {

    var j, alas, tinggi, n int

    var luas float64

    fmt.Scan(&n)

    for j = 1; j <=n; j+=1 {

        fmt.Scan(&alas, &tinggi)

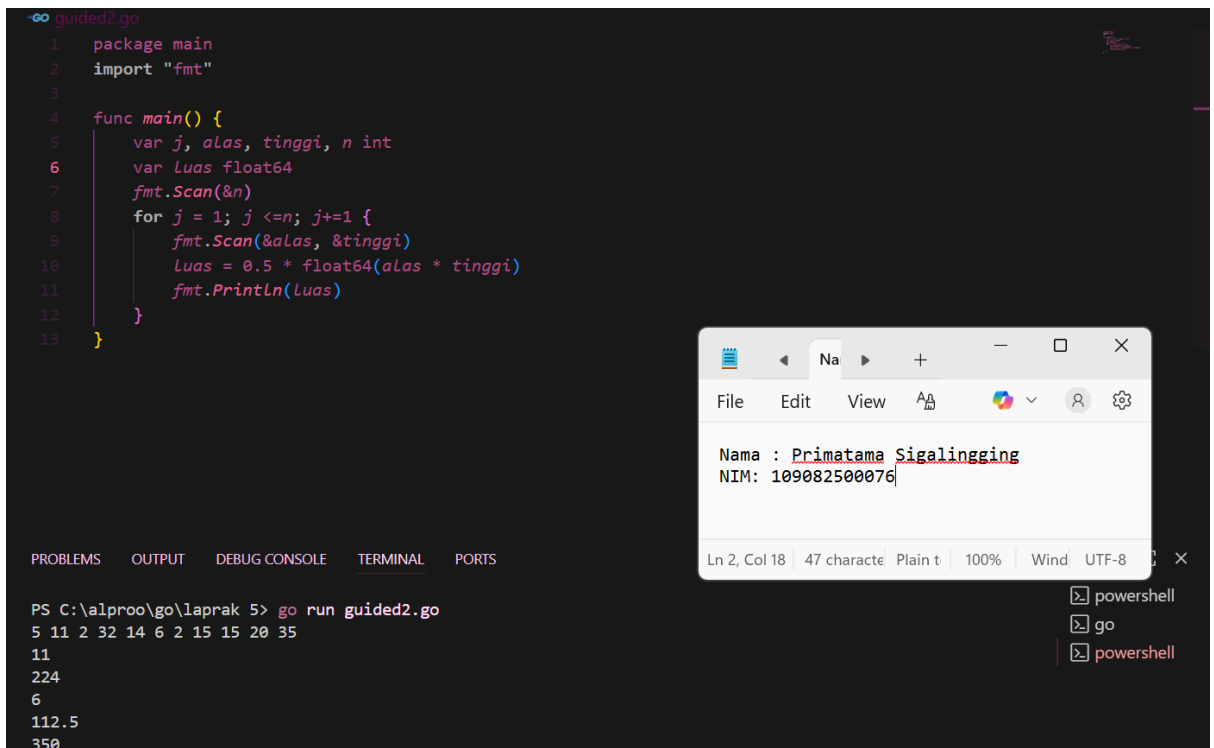
        luas = 0.5 * float64(alas * tinggi)

        fmt.Println(luas)

    }

}
```

### Screenshoot program



```
guided2.go
1 package main
2 import "fmt"
3
4 func main() {
5     var j, alas, tinggi, n int
6     var luas float64
7     fmt.Scan(&n)
8     for j = 1; j <=n; j+=1 {
9         fmt.Scan(&alas, &tinggi)
10        luas = 0.5 * float64(alas * tinggi)
11        fmt.Println(luas)
12    }
13 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\alproo\go\laprak 5> go run guided2.go
5 11 2 32 14 6 2 15 15 20 35
11
224
6
112.5
350
```

Ln 2, Col 18 | 47 character | Plain text | 100% | Window | UTF-8

powerShell  
go  
powerShell

### Deskripsi program

## 3. Guided 3

## Source Code

```
package main

import "fmt"

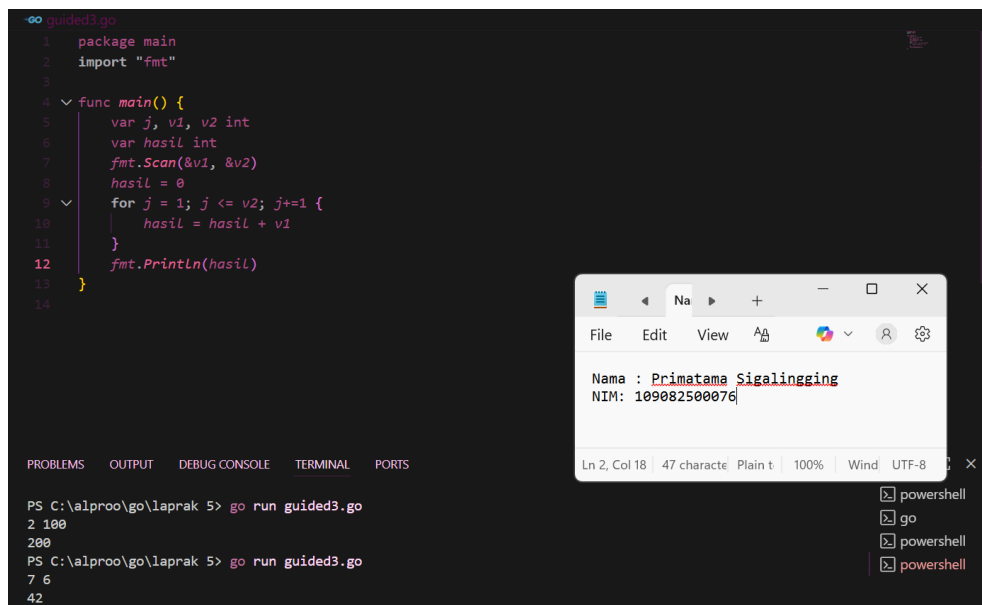
func main() {
    var j, v1, v2 int
    var hasil int
    fmt.Scan(&v1, &v2)

    hasil = 0

    for j = 1; j <= v2; j+=1 {
        hasil = hasil + v1
    }

    fmt.Println(hasil)
}
```

## Screenshoot program



## Deskripsi program

## TUGAS

### 1. Tugas 1

#### Source code

```
package main

import "fmt"

func main() {
    var n, hasil int

    fmt.Print("Masukkan batas angka n: ")

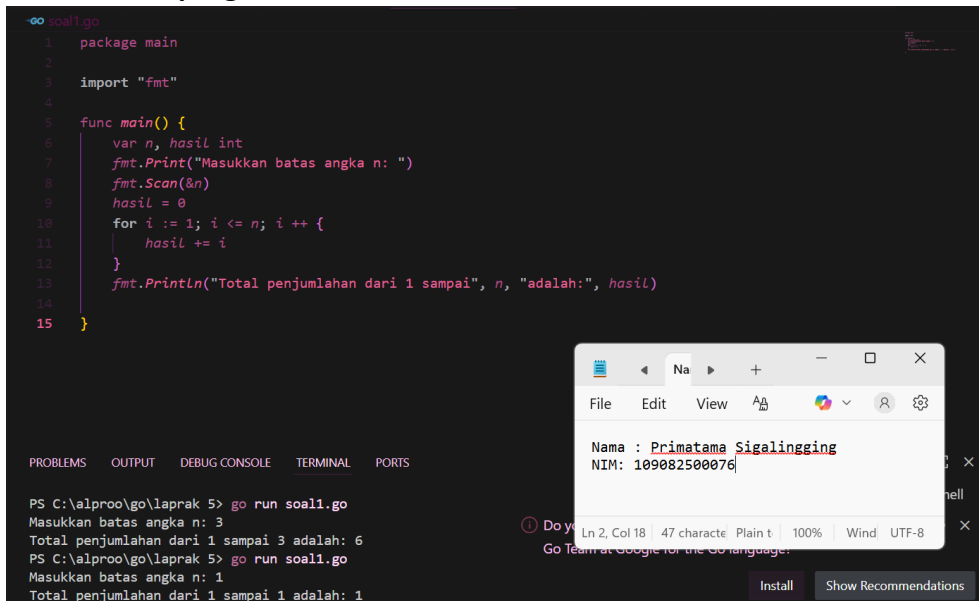
    fmt.Scan(&n)

    hasil = 0

    for i := 1; i <= n; i ++ {
        hasil += i
    }

    fmt.Println("Total penjumlahan dari 1 sampai", n, "adalah:",
hasil)
}
```

#### Screenshoot program



The screenshot displays a Go IDE with a dark theme. The editor shows the source code for a program that calculates the sum of numbers from 1 to a user-defined limit. The code is as follows:

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var n, hasil int
7     fmt.Print("Masukkan batas angka n: ")
8     fmt.Scan(&n)
9     hasil = 0
10    for i := 1; i <= n; i ++ {
11        hasil += i
12    }
13    fmt.Println("Total penjumlahan dari 1 sampai", n, "adalah:", hasil)
14 }
15
```

Below the editor, the terminal output shows the program being run twice. In the first run, the user enters 3, and the output is "Total penjumlahan dari 1 sampai 3 adalah: 6". In the second run, the user enters 1, and the output is "Total penjumlahan dari 1 sampai 1 adalah: 1".

Overlaid on the bottom right of the IDE is a small window titled "Na" with a menu bar (File, Edit, View) and a toolbar. It contains the text:

```
Nama : Primatama Sigalingging
NIM: 109082500076
```

At the bottom of the IDE, there are buttons for "Install" and "Show Recommendations".

## **Deskripsi program**

### ➤ **Tujuan**

Program ini dibuat untuk menghitung total penjumlahan dari angka 1 sampai angka tertentu yang dimasukkan oleh pengguna. Tujuannya adalah memahami bagaimana perulangan bekerja dalam menghitung nilai yang terus bertambah secara otomatis, tanpa perlu menulis banyak baris penjumlahan secara manual.

### ➤ **Proses**

Ketika dijalankan, program akan meminta pengguna untuk memasukkan batas angka, misalnya 10. Setelah itu, sistem akan menambahkan setiap angka dari 1 hingga 10 satu per satu secara berurutan.

Prosesnya berjalan cepat karena dilakukan oleh komputer menggunakan perulangan. Pengguna tidak perlu menghitung secara manual, karena hasil akhirnya langsung ditampilkan di layar.

Program ini juga bisa dikembangkan agar menampilkan proses perhitungannya, misalnya menampilkan urutan " $1 + 2 + 3 + \dots + n$ " sebelum menampilkan hasil akhirnya. Dengan begitu, pengguna bisa melihat secara langsung bagaimana perulangan bekerja di balik layar.

### ➤ **Kesimpulan**

Program ini membantu memahami konsep dasar looping dan logika penjumlahan otomatis dalam pemrograman. Meskipun sederhana, konsepnya sangat penting karena hampir semua program komputasi melibatkan proses berulang seperti ini.

## 2. Tugas 2

### Source code

```
package main

import (
    "fmt"
    "math"
)

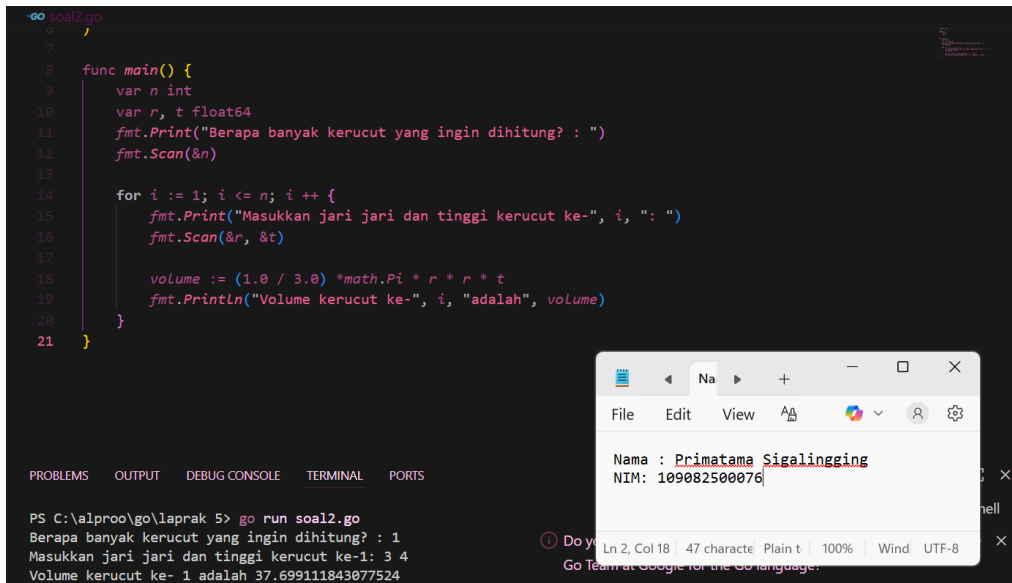
func main() {
    var n int
    var r, t float64

    fmt.Print("Berapa banyak kerucut yang ingin dihitung? : ")
    fmt.Scan(&n)

    for i := 1; i <= n; i ++ {
        fmt.Print("Masukkan jari jari dan tinggi kerucut ke-", i, ": ")
        fmt.Scan(&r, &t)

        volume := (1.0 / 3.0) * math.Pi * r * r * t
        fmt.Println("Volume kerucut ke-", i, "adalah", volume)
    }
}
```

## Screenshoot program



```
1 // soal2.go
2
3
4
5
6
7
8 func main() {
9     var n int
10    var r, t float64
11    fmt.Println("Berapa banyak kerucut yang ingin dihitung? : ")
12    fmt.Scan(&n)
13
14    for i := 1; i <= n; i++ {
15        fmt.Print("Masukkan jari jari dan tinggi kerucut ke-", i, ": ")
16        fmt.Scan(&r, &t)
17
18        volume := (1.0 / 3.0) * math.Pi * r * r * t
19        fmt.Println("Volume kerucut ke-", i, "adalah", volume)
20    }
21 }
```

PS C:\alproo\go\laprak 5> go run soal2.go  
Berapa banyak kerucut yang ingin dihitung? : 1  
Masukkan jari jari dan tinggi kerucut ke-1: 3 4  
Volume kerucut ke- 1 adalah 37.699111843077524

Nama : Primatama Sigalingging  
NIM: 109082500076

## Deskripsi program

### ➤ Tujuan

Tujuan utama program ini adalah untuk menghitung volume beberapa kerucut sekaligus dengan cara yang praktis. Selain itu, program ini juga melatih kemampuan pengguna dalam mengelola input yang lebih dari satu, serta memahami bagaimana data diproses berulang kali.

### ➤ Proses

Pertama, pengguna akan diminta untuk menentukan berapa banyak kerucut yang ingin dihitung volumenya. Setelah itu, untuk setiap kerucut, pengguna memasukkan dua nilai ukuran alas dan tinggi.

Program kemudian memproses setiap data satu per satu dan langsung menampilkan hasil perhitungannya di layar. Proses ini dilakukan secara berulang hingga semua data selesai dihitung.

Program ini cukup interaktif karena menampilkan hasil setiap kali satu kerucut selesai dihitung. Selain itu, pengguna juga bisa mengetahui bila ada input yang salah, seperti angka negatif atau huruf, dan mengulanginya dengan benar.

### ➤ Kesimpulan

Melalui program ini, kita bisa memahami bagaimana cara komputer mengolah banyak data secara efisien menggunakan perulangan. Walaupun hasil akhirnya hanya berupa angka, logika di balik program ini mencerminkan cara berpikir sistematis dan runtut dalam menyelesaikan masalah.



### 3. Tugas 3

#### Source code

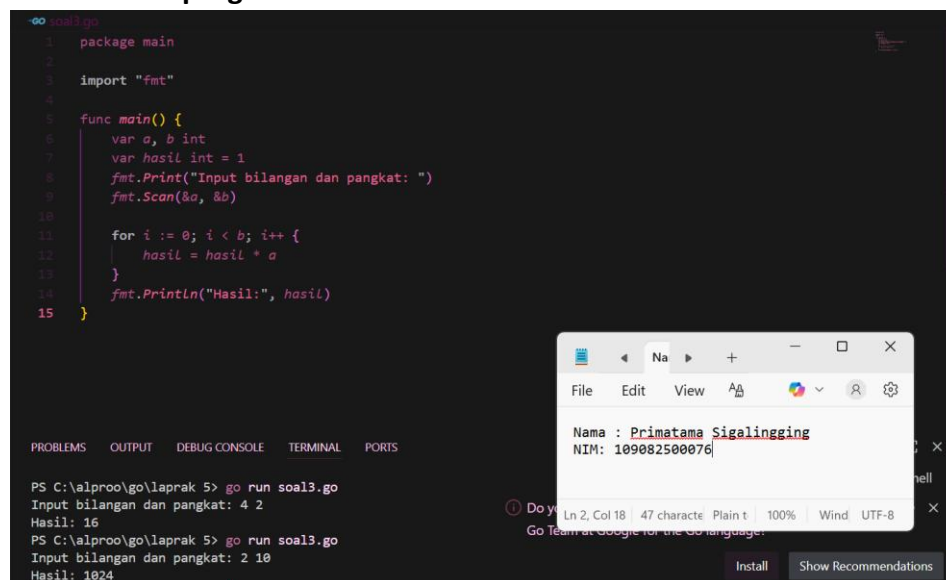
```
package main

import "fmt"

func main() {
    var a, b int
    var hasil int = 1
    fmt.Print("Input bilangan dan pangkat: ")
    fmt.Scan(&a, &b)

    for i := 0; i < b; i++ {
        hasil = hasil * a
    }
    fmt.Println("Hasil:", hasil)
}
```

#### Screenshoot program



#### Deskripsi program

##### ➤ Tujuan

Program ini dibuat untuk menghitung hasil perpangkatan dari sebuah bilangan dengan pangkat tertentu. Tujuan pembuatannya adalah untuk melatih logika dalam proses perkalian berulang serta memahami bagaimana komputer bisa menjalankan operasi matematis yang sama secara berulang tanpa kesalahan.

##### ➤ Proses

Ketika program dijalankan, pengguna cukup memasukkan dua angka — angka dasar dan pangkatnya. Misalnya pengguna ingin tahu berapa hasil dari 2 pangkat 4, maka setelah input dimasukkan, program akan mengalikan angka 2 dengan dirinya sendiri sebanyak empat kali.

Proses ini menggunakan struktur perulangan sehingga bisa menyesuaikan dengan berapa pun pangkat yang dimasukkan pengguna.

Program juga bisa dikembangkan lebih lanjut agar mendukung perhitungan pangkat negatif atau menampilkan hasil dalam bentuk langkah-langkah, misalnya “ $2 \times 2 \times 2 \times 2 = 16$ ”.

➤ **Kesimpulan**

Program ini mengajarkan bahwa operasi matematika seperti perpangkatan bisa disusun menggunakan logika perulangan. Dari program sederhana ini, kita bisa memahami dasar dari bagaimana komputer melakukan komputasi matematis secara berulang dan konsisten.

#### 4. Tugas 4

##### Source Code

```
package main

import "fmt"

func main() {
    var n int
    fmt.Print("Masukkan angka: ")
    fmt.Scan(&n)
    var hasil int = 1
    for i := 1; i <= n; i++ {
        hasil = hasil * i
    }
    fmt.Println("Hasil faktorial dari", n, "adalah:", hasil)
}
```

## Screenshoot Program

```
14 soal4.go
2
3 import "fmt"
4
5 func main() {
6     var n int
7     fmt.Print("Masukkan angka: ")
8     fmt.Scan(&n)
9     var hasil int = 1
10    for i := 1; i <= n; i++ {
11        hasil = hasil * i
12    }
13    fmt.Println("Hasil faktorial dari", n, "adalah:", hasil)
14 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\alproo\go\laprak 5> go run soal4.go  
Masukkan angka: 5  
Hasil faktorial dari 5 adalah: 120  
PS C:\alproo\go\laprak 5> go run soal4.go  
Masukkan angka: 10  
Hasil faktorial dari 10 adalah: 3628800

File Edit View A A 100% Wind UTF-8

Nama : Primatama Sigalingging  
NIM: 109082500076

Ln 2, Col 18 | 47 character Plain t 100% Wind UTF-8

Go Team at Google for the Go language.

Install Show Recommendations

## Deskripsi Program

### ➤ Tujuan

Program ini bertujuan untuk menghitung faktorial dari suatu bilangan yang dimasukkan oleh pengguna, sekaligus melatih pemahaman terhadap konsep pengulangan dan operasi perkalian bertahap.

### ➤ Proses

Saat dijalankan, program meminta pengguna untuk memasukkan satu angka.

Misalnya pengguna memasukkan angka 5, maka program akan menghitung hasil kali dari 1 sampai 5. Semua proses dilakukan secara otomatis menggunakan perulangan, sehingga hasil akhir bisa langsung tampil di layar.

Selain itu, program juga memperhatikan kondisi khusus. Misalnya, jika pengguna memasukkan angka negatif, sistem akan menolak input tersebut karena faktorial tidak bisa dihitung untuk bilangan negatif. Sementara jika pengguna memasukkan angka 0, hasil yang muncul adalah 1, sesuai aturan dasar matematika.

### ➤ Kesimpulan

Program faktorial ini menggambarkan penerapan logika berulang yang terstruktur dan terkontrol. Selain berguna untuk latihan logika dasar, program ini juga sering digunakan dalam konteks algoritma lanjutan seperti kombinasi, permutasi, dan analisis komputasi.