

LAPORAN PRAKTIKUM
Algoritma Pemrograman

Evaluasi



Disusun oleh:

Andra Dwicki Saputra

109082500206

S1IF-13-07

PROGRAM STUDI S1 INFORMATIKA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

SOAL

1. SOAL 1

Source Code

```
package main

import "fmt"

func main() {

    var N int

    fmt.Scan(&N)

    if N >= 8 {

        fmt.Println("Tinggi pohon harus kurang dari 8")

        return

    }

    for i := 1; i <= N; i++ {

        for j := 1; j <= N-i; j++ {

            fmt.Print(" ")

        }

        for k := 1; k <= 2*i-1; k++ {

            fmt.Print("*")

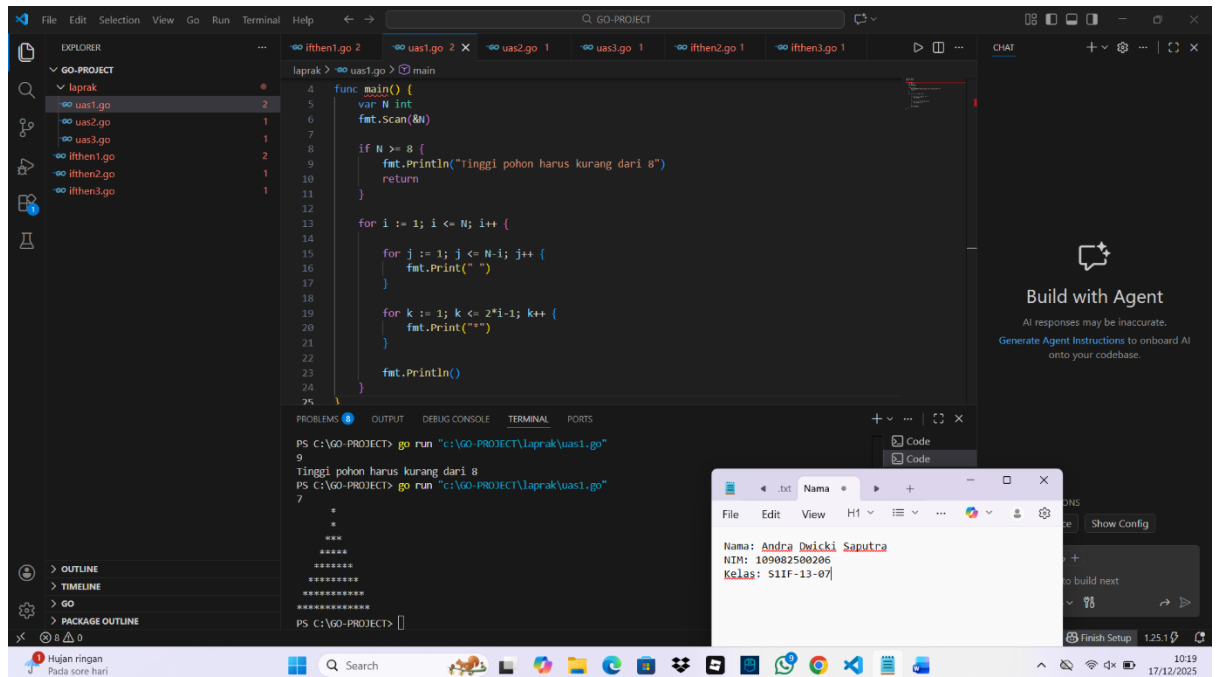
        }

        fmt.Println()

    }

}
```

Screenshoot program



```
func main() {
    var N int
    fmt.Scan(&N)

    if N >= 8 {
        fmt.Println("tinggi pohon harus kurang dari 8")
        return
    }

    for i := 1; i <= N; i++ {
        for j := 1; j <= N-i; j++ {
            fmt.Print(" ")
        }

        for k := 1; k <= 2*i-1; k++ {
            fmt.Print("*")
        }

        fmt.Println()
    }
}
```

```
PS C:\GO-PROJECT> go run "c:\GO-PROJECT\laprak\uas1.go"
9
tinggi pohon harus kurang dari 8
PS C:\GO-PROJECT> go run "c:\GO-PROJECT\laprak\uas1.go"
7
      *
     **
    ***
   ****
  *****
 *****
*****
PS C:\GO-PROJECT>
```

Deskripsi program

Program tersebut dibuat untuk mencetak bentuk pohon natal menggunakan simbol asterisk (*) berdasarkan tinggi yang diinputkan oleh pengguna. Pada awal program, paket `fmt` digunakan untuk keperluan input dan output. Di dalam fungsi `main`, program terlebih dahulu membaca sebuah bilangan bulat `N` yang merepresentasikan tinggi pohon. Setelah itu dilakukan pengecekan kondisi: jika nilai `N` lebih besar atau sama dengan 8, maka program langsung menampilkan pesan bahwa tinggi pohon harus kurang dari 8 dan program dihentikan. Pengecekan ini sesuai dengan ketentuan soal yang membatasi tinggi pohon agar tidak melebihi jumlah tertentu.

Jika nilai `N` memenuhi syarat (kurang dari 8), program kemudian masuk ke proses pencetakan pohon natal. Proses ini menggunakan perulangan `for` yang berjalan dari 1 sampai `N`, di mana setiap iterasi merepresentasikan satu baris pohon. Pada setiap baris, program terlebih dahulu mencetak spasi sebanyak `N - i` untuk membuat posisi asterisk berada di tengah. Setelah itu, program mencetak simbol asterisk sebanyak `2*i - 1` sehingga terbentuk pola segitiga yang semakin lebar ke bawah. Setelah satu baris selesai dicetak, program berpindah ke baris berikutnya.

Setelah seluruh bagian daun pohon tercetak, program mencetak batang pohon. Batang ini dibuat dengan mencetak spasi sebanyak `N - 1` agar posisinya sejajar dengan tengah pohon, lalu diakhiri dengan satu karakter `|`

2. SOAL 2

Source Code

```
package main

import "fmt"

func main() {

    var a, b int

    fmt.Println("=== Daftar Produk Toko Budi ===")

    fmt.Println("1. Little Trees          - 35.000")
    fmt.Println("2. Lap microfiber          - 25.000")
    fmt.Println("3. Cover Steer            - 150.000")
    fmt.Println("4. Sponge cuci mobil    - 10.000")

    fmt.Print("Pilih menu (1-4): ")

    fmt.Scan(&a)

    fmt.Print("Masukkan jumlah porsi: ")

    fmt.Scan(&b)

    harga := 0

    nama := ""

    switch a {

    case 1:

        harga = 35000

        nama = "Little Trees"

    case 2:

        harga = 25000

        nama = "Lap microfiber"
```

```
        case 3:

            harga = 150000

            nama = "Cover Steer"

        case 4:

            harga = 10000

            nama = "Sponge cuci mobil"

        default:

            fmt.Println("Menu tidak tersedia")

            return

    }

    total := harga * b

    fmt.Println("\n=== STRUK PEMBELIAN ===")

    fmt.Println("Produk :", nama)

    fmt.Println("Jumlah :", a)

    fmt.Println("Total  :", total)

}
```

Screenshoot program

The screenshot shows the Visual Studio Code editor with a Go project named 'GO-PROJECT'. The Explorer pane on the left shows the file structure. The main editor displays the file 'uas2.go' with the following code:

```
func main() {  
    var a, b int  
  
    fmt.Println("=== Daftar Produk Toko Budi ===")  
    fmt.Println("1. Little Trees      - 35.000")  
    fmt.Println("2. Lap microfiber      - 25.000")  
    fmt.Println("3. Cover Steer        - 150.000")  
    fmt.Println("4. Sponge cuci mobil  - 10.000")  
  
    fmt.Print("Pilih menu (1-4): ")  
    fmt.Scan(&a)  
  
    fmt.Print("Masukkan jumlah porsi: ")  
    fmt.Scan(&b)  
  
    harga := 0  
    nama := ""  
  
    switch a {  
    case 1:  
        harga = 35000  
        nama = "Little Trees"  
    }  
}
```

The terminal window at the bottom shows the output of the program:

```
PS C:\GO-PROJECT> go run ".\GO-PROJECT\laprak\uas2.go"  
=== Daftar Produk Toko Budi ===  
1. Little Trees      - 35.000  
2. Lap microfiber      - 25.000  
3. Cover Steer        - 150.000  
4. Sponge cuci mobil  - 10.000  
Pilih menu (1-4): 1  
Masukkan jumlah porsi: 5  
  
=== STRUK PEMBELIAN ===  
Produk : Little Trees  
Jumlah : 1  
Total : 175000  
PS C:\GO-PROJECT>
```

A small text editor window titled 'Nama' is open in the foreground, containing the following text:

```
Nama: Andra Dwicki Saputra  
NIM: 109002500206  
Kelas: SIIF-13-07
```

The screenshot shows the Visual Studio Code editor with a Go project named 'GO-PROJECT'. The Explorer pane on the left shows the file structure. The main editor displays the file 'uas2.go' with the following code:

```
func main() {  
    case 2:  
        harga = 25000  
        nama = "Lap microfiber"  
    case 3:  
        harga = 150000  
        nama = "Cover Steer"  
    case 4:  
        harga = 10000  
        nama = "Sponge cuci mobil"  
    default:  
        fmt.Println("Menu tidak tersedia")  
        return  
    }  
  
    total := harga * b  
  
    fmt.Println("\n=== STRUK PEMBELIAN ===")  
    fmt.Println("Produk :", nama)  
    fmt.Println("Jumlah :", a)  
    fmt.Println("Total :", total)  
}
```

The terminal window at the bottom shows the output of the program:

```
PS C:\GO-PROJECT> go run ".\GO-PROJECT\laprak\uas2.go"  
=== Daftar Produk Toko Budi ===  
1. Little Trees      - 35.000  
2. Lap microfiber      - 25.000  
3. Cover Steer        - 150.000  
4. Sponge cuci mobil  - 10.000  
Pilih menu (1-4): 1  
Masukkan jumlah porsi: 5  
  
=== STRUK PEMBELIAN ===  
Produk : Little Trees  
Jumlah : 1  
Total : 175000  
PS C:\GO-PROJECT>
```

A small text editor window titled 'Nama' is open in the foreground, containing the following text:

```
Nama: Andra Dwicki Saputra  
NIM: 109002500206  
Kelas: SIIF-13-07
```

Deskripsi program

Program tersebut merupakan program POS (Point of Sale) sederhana yang digunakan untuk menghitung total pembayaran berdasarkan produk yang dipilih dan jumlah pembelian. Program diawali dengan deklarasi package main dan import paket fmt yang berfungsi untuk menangani proses input dan output. Di dalam fungsi main, terdapat dua variabel integer, yaitu a untuk menyimpan pilihan menu produk dan b untuk menyimpan jumlah porsi atau jumlah barang yang dibeli oleh pengguna. Selanjutnya, program menampilkan daftar produk Toko Budi beserta harganya ke layar menggunakan beberapa perintah fmt.Println. Setelah daftar produk ditampilkan, program meminta pengguna untuk memilih menu dengan memasukkan angka 1 sampai 4, yang kemudian disimpan ke dalam variabel a. Program juga meminta pengguna memasukkan jumlah porsi atau jumlah barang yang ingin dibeli, dan nilai tersebut disimpan ke dalam variabel b.

Setelah menerima input, program mendeklarasikan variabel harga dan nama yang masing-masing digunakan untuk menyimpan harga satuan produk dan nama produk yang dipilih. Pemilihan produk dilakukan menggunakan struktur kontrol switch berdasarkan nilai variabel a. Jika pengguna memilih menu 1 sampai 4, program akan mengisi variabel harga dan nama sesuai dengan produk yang dipilih. Jika pengguna memasukkan pilihan di luar menu yang tersedia, program akan menampilkan pesan "Menu tidak tersedia" dan menghentikan eksekusi program menggunakan perintah return.

Pada bagian akhir, program menghitung total pembayaran dengan cara mengalikan harga satuan produk (harga) dengan jumlah pembelian (b). Hasil perhitungan tersebut disimpan dalam variabel total, yang nantinya dapat digunakan untuk ditampilkan sebagai total biaya yang harus dibayar oleh pelanggan. Dengan alur ini, program mampu mensimulasikan proses pemilihan produk dan perhitungan total harga secara sederhana

3. SOAL 3

Source Code

```
package main

import "fmt"

func main() {

    var a, b, c int

    fmt.Scan(&a, &b, &c)
```

```
if a+b <= c || a+c <= b || b+c <= a {  
  
    fmt.Println("Bukan segitiga")  
  
    return  
  
}  
  
if a == b && b == c {  
  
    fmt.Println("Segitiga sama sisi")  
  
    return  
  
}  
  
x, y, z := a, b, c  
  
if x > y {  
  
    x, y = y, x  
  
}  
  
if y > z {  
  
    y, z = z, y  
  
}  
  
if x > y {  
  
    x, y = y, x  
  
}  
  
if x*x+y*y == z*z {  
  
    fmt.Println("Segitiga siku-siku")  
  
    return  
  
}  
  
if a == b || a == c || b == c {
```



```

        fmt.Println("Segitiga sama kaki")

    }

    return

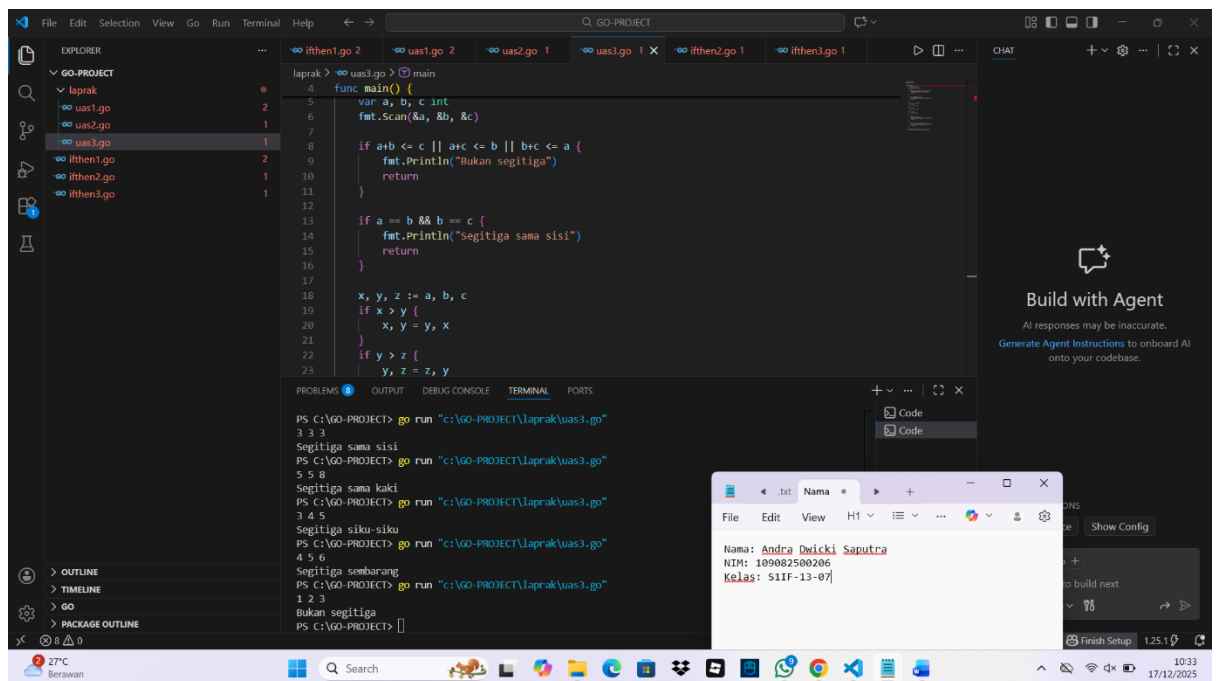
}

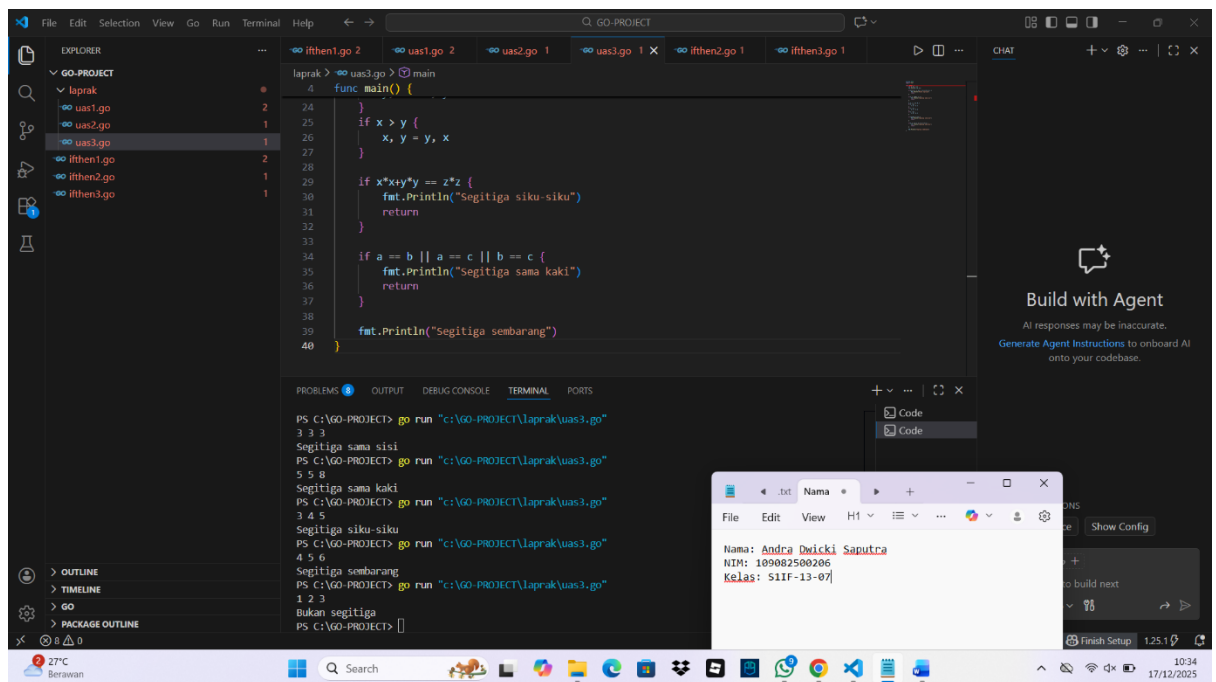
fmt.Println("Segitiga sembarang")

}

```

Screenshoot program





Deskripsi program

Program tersebut digunakan untuk menentukan jenis segitiga berdasarkan tiga buah bilangan bulat positif yang diinputkan oleh pengguna. Di dalam fungsi main, program membaca tiga buah nilai, yaitu a, b, dan c, yang masing-masing merepresentasikan panjang sisi-sisi segitiga.

Langkah pertama yang dilakukan program adalah mengecek syarat terbentuknya segitiga. Program memastikan bahwa jumlah dua sisi selalu lebih besar dari sisi yang lain. Jika salah satu kondisi tersebut tidak terpenuhi, maka program menyimpulkan bahwa ketiga nilai tersebut tidak dapat membentuk segitiga dan langsung menampilkan output "Bukan segitiga", kemudian menghentikan eksekusi program dengan perintah return.

Jika syarat segitiga terpenuhi, program kemudian mengecek apakah ketiga sisi memiliki panjang yang sama. Apabila a, b, dan c bernilai sama, program menyimpulkan bahwa segitiga tersebut adalah segitiga sama sisi, lalu mencetak hasilnya dan menghentikan program. Setelah itu, program menyalin nilai sisi ke dalam variabel x, y, dan z, kemudian mengurutkannya dari yang terkecil hingga terbesar. Proses pengurutan ini bertujuan untuk mempermudah pengecekan jenis segitiga siku-siku menggunakan teorema Pythagoras.

Setelah sisi diurutkan, program memeriksa apakah kuadrat sisi terkecil ditambah kuadrat sisi tengah sama dengan kuadrat sisi terbesar. Jika kondisi ini terpenuhi, maka segitiga tersebut dikategorikan sebagai segitiga siku-siku dan hasilnya langsung ditampilkan. Jika bukan segitiga siku-siku, program selanjutnya mengecek apakah terdapat dua sisi yang sama panjang. Jika ada dua sisi yang sama, maka segitiga tersebut diklasifikasikan sebagai segitiga sama kaki. Apabila seluruh kondisi sebelumnya tidak terpenuhi, maka program menyimpulkan bahwa segitiga tersebut

adalah segitiga sembarang, yaitu segitiga dengan tiga sisi yang berbeda panjang dan tidak siku-siku