LAPORAN PRAKTIKUM

Algoritma Pemrograman

EVALUASI

Disusun oleh:

Abdillahtama rifdy gumilang

109082500054

S1IF-13-07

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

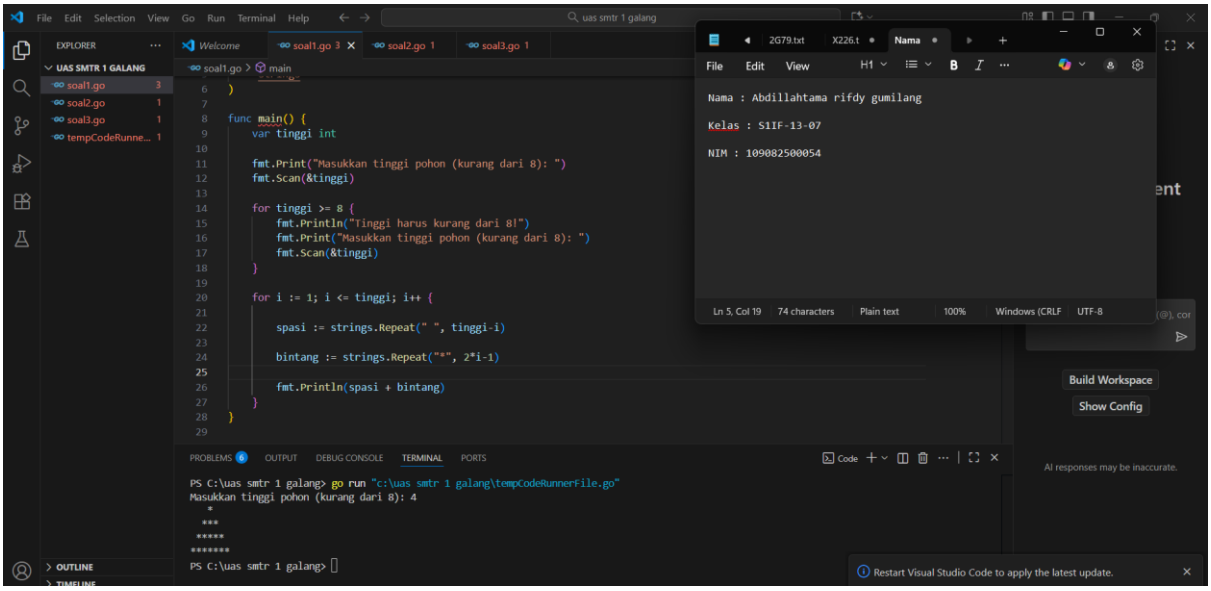
2025

SOAL

1. **SOAL 1**
   Source Code

```
package main

import (
    "fmt"
    "strings"
)

func main() {
    var tinggi int

    fmt.Print("Masukkan tinggi pohon (kurang dari 8): ")
    fmt.Scan(&tinggi)

    for tinggi >= 8 {
        fmt.Println("Tinggi harus kurang dari 8!")
        fmt.Print("Masukkan tinggi pohon (kurang dari 8): ")
        fmt.Scan(&tinggi)
    }

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

        spasi := strings.Repeat(" ", tinggi-i)

        bintang := strings.Repeat("*", 2*i-1)

        fmt.Println(spasi + bintang)
    }
}
```

Screenshoot program



Deskripsi program

Program ini adalah aplikasi sederhana menggambar pohon natal berbentuk piramida menggunakan karakter asterisk (*). Program dimulai dengan meminta pengguna memasukkan tinggi pohon melalui keyboard, yang akan disimpan dalam variabel integer. Namun, sebelum memproses lebih lanjut, program melakukan validasi ketat terhadap input pengguna dengan memastikan nilai tinggi tersebut kurang dari 8, sesuai dengan cerita awal yang menyebutkan karakter Towel yang rendah hati. Jika pengguna memasukkan angka 8 atau lebih, program akan menampilkan pesan peringatan dan meminta input ulang secara berulang hingga syarat terpenuhi.
Setelah tinggi pohon yang valid diperoleh, program masuk ke fase pembentukan pola piramida. Inti dari program ini terletak pada sebuah loop for yang akan dieksekusi sebanyak tinggi pohon yang dimasukkan. Pada setiap iterasi loop, program menghitung jumlah spasi dan bintang yang diperlukan untuk membentuk baris piramida yang proporsional. Jumlah spasi dihitung dengan rumus 'tinggi-i', yang membuat spasi berkurang secara bertahap dari baris atas ke baris bawah, sementara jumlah bintang mengikuti rumus '2*i-1' untuk menghasilkan deret bilangan ganjil yang membentuk efek melebar.
Untuk implementasi program memanfaatkan fungsi strings.Repeat() yang berfungsi mengulang sebuah karakter sebanyak jumlah tertentu. Fungsi ini digunakan dua kali dalam setiap iterasi: pertama untuk menghasilkan sejumlah spasi sebagai padding kiri, dan kedua untuk membuat rangkaian bintang sebagai tubuh pohon. Kombinasi antara spasi yang semakin berkurang dan bintang yang semakin bertambah inilah yang menciptakan ilusi visual piramida rapi di layar output.

Output program ini sepenuhnya berbasis teks, di mana setiap baris piramida dicetak ke layar menggunakan fmt.Println(). Hasil akhirnya adalah representasi pohon natal simetris dengan puncak runcing di bagian atas dan dasar yang lebar di bagian bawah.

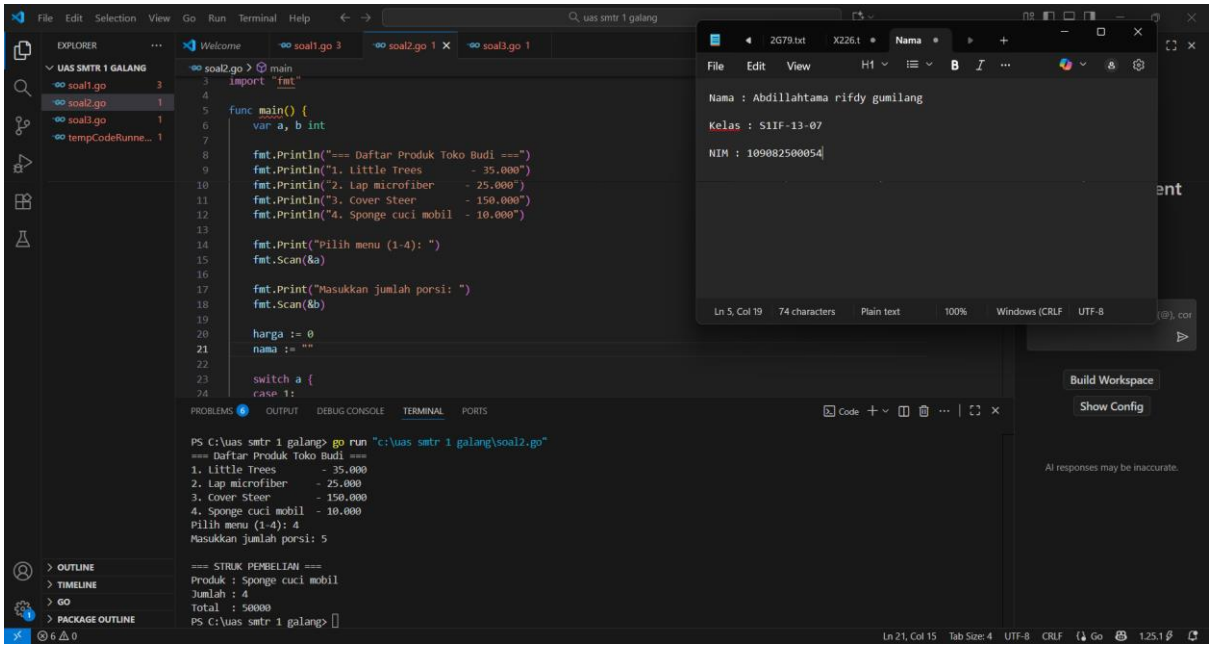
2. **SOAL 2**

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



Deskripsi program

Program ini adalah sistem kasir sederhana Program dimulai dengan menampilkan daftar produk yang tersedia beserta harganya, kemudian meminta pengguna untuk memilih menu produk (1-4) dan memasukkan jumlah porsi yang ingin dibeli. Input dari pengguna dibaca menggunakan `fmt.Scan()` dan disimpan dalam variabel `a` untuk pilihan menu dan `b` untuk jumlah porsi.

Struktur logika program menggunakan statement `switch` untuk memproses pilihan menu yang dimasukkan pengguna. Setiap `case` dalam `switch` menetapkan harga dan nama produk yang sesuai berdasarkan pilihan: Little Trees (35.000), Lap microfiber (25.000), Cover Steer (150.000), atau Sponge cuci mobil (10.000). Jika pengguna memasukkan angka di luar rentang 1-4, program akan menampilkan pesan "Menu tidak tersedia" dan berhenti eksekusi menggunakan `return`.

Setelah menentukan harga satuan dan nama produk, program menghitung total biaya dengan mengalikan harga satuan dengan jumlah porsi yang diminta. Perhitungan ini menghasilkan nilai yang disimpan dalam variabel `total`, yang kemudian akan ditampilkan dalam struk pembelian. Program tidak menerapkan pengecekan tambahan seperti validasi input negatif atau kapitalisasi error handling untuk input non-angka.

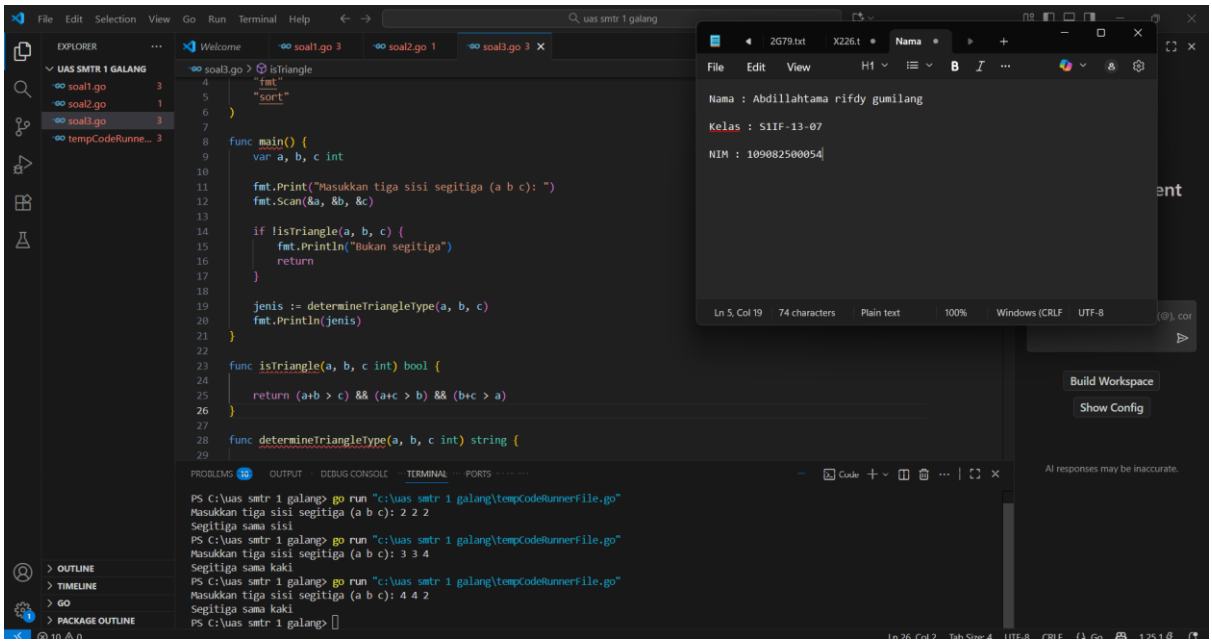
Terakhir, program mencetak struk pembelian yang berisi detail transaksi: nama produk, jumlah porsi, dan total harga yang harus dibayar.

3. **SOAL 3**
   Source Code

```
package main

import (
    "fmt"
    "sort"
)

func main() {
    var a, b, c int

    fmt.Print("Masukkan tiga sisi segitiga (a b c): ")
    fmt.Scan(&a, &b, &c)

    if !isTriangle(a, b, c) {
        fmt.Println("Bukan segitiga")
        return
    }

    jenis := determineTriangleType(a, b, c)
    fmt.Println(jenis)
}

func isTriangle(a, b, c int) bool {

    return (a+b > c) && (a+c > b) && (b+c > a)
}

func determineTriangleType(a, b, c int) string {

    if a == b && b == c {
        return "Segitiga sama sisi"
    }
```

```
        if a == b || a == c || b == c {
            return "Segitiga sama kaki"
        }

        sides := []int{a, b, c}
        sort.Ints(sides)

        if sides[2]*sides[2] ==
    sides[0]*sides[0]+sides[1]*sides[1] {
            return "Segitiga siku-siku"
        }

        return "Segitiga sembarang"
    }
```

Screenshoot program



Deskripsi program

Program ini memulai eksekusi dengan meminta pengguna memasukkan tiga bilangan bulat yang merepresentasikan panjang sisi segitiga melalui perintah `fmt.Print("Masukkan tiga sisi segitiga (a b c): ")`. Input dari pengguna dibaca menggunakan `fmt.Scan(&a, &b, &c)` yang secara otomatis memisahkan tiga nilai yang dimasukkan berdasarkan spasi dan menyimpannya dalam variabel `a`, `b`, dan `c`. Proses input ini menjadi dasar untuk seluruh analisis geometri yang akan dilakukan program, di mana ketiga bilangan ini akan ditentukan apakah dapat membentuk suatu segitiga dan termasuk jenis segitiga apa.

Setelah input diperoleh, program memanggil fungsi `isTriangle(a, b, c)` untuk melakukan validasi dasar. Fungsi ini menerapkan prinsip ketidaksetamaan segitiga dengan mengecek apakah jumlah dua sisi selalu lebih besar dari sisi ketiga melalui ekspresi boolean `(a+b > c) && (a+c > b) && (b+c > a)`. Jika salah satu kondisi ini tidak terpenuhi, program langsung mencetak "Bukan segitiga" dan berhenti eksekusi menggunakan `return`, yang mengindikasikan bahwa tiga bilangan tersebut secara matematis tidak mungkin membentuk bangun segitiga.

Jika validasi berhasil, program melanjutkan ke fungsi `determineTriangleType(a, b, c)` yang melakukan serangkaian pengecekan hierarkis. Pertama, fungsi mengecek apakah ketiga sisi sama panjang untuk mengidentifikasi segitiga sama sisi. Jika tidak, dilanjutkan dengan mengecek apakah dua sisi sama panjang untuk segitiga sama kaki. Untuk deteksi segitiga siku-siku, fungsi mengurutkan sisi-sisi menggunakan `sort.Ints(sides)` lalu menerapkan teorema Pythagoras `sides[2]*sides[2] == sides[0]*sides[0]+sides[1]*sides[1]` pada hipotenusa (sisi terpanjang). Jika tidak ada kondisi yang terpenuhi, segitiga diklasifikasikan sebagai sembarang.

Hasil akhir program berupa string deskriptif yang dicetak ke layar sesuai klasifikasi yang telah ditentukan. Output ini memberikan informasi langsung kepada pengguna tentang jenis segitiga yang dibentuk oleh tiga sisi input, mulai dari "Segitiga sama sisi", "Segitiga sama kaki", "Segitiga siku-siku", "Segitiga sembarang", atau "Bukan segitiga" jika gagal validasi awal. Program ini dengan efisien menggabungkan proses input, validasi matematis, analisis geometri, dan output informatif dalam alur logis yang terstruktur dengan baik.