# LAPORAN PRAKTIKUM
# ALGORITMA DAN PEMROGRAMAN 1
# MODUL 13
# “REPEAT-UNTIL”


**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**:

1. **Pengertian Komposisi**

- **Komposisi dalam algoritma dan pemrograman mengacu pada penggabungan berbagai struktur kontrol untuk menyelesaikan masalah yang lebih kompleks.**
- **Modul ini memfokuskan pada penerapan seluruh konsep yang telah dipelajari sebelumnya, seperti perulangan, seleksi, dan operasi dasar, secara terintegrasi untuk menyelesaikan masalah.**

2. **Tujuan Komposisi**

- **Mengasah kemampuan untuk menggabungkan berbagai struktur kontrol.**
- **Meningkatkan kemampuan berpikir algoritmik untuk menyelesaikan permasalahan yang lebih kompleks dan realistis.**
- **Memahami penggunaan struktur kontrol yang efisien.**

3. **Komponen Utama**

- **Perulangan (Looping): Digunakan untuk pengulangan aksi tertentu.**
- **Seleksi (Conditional Statements): Memilih jalur eksekusi berdasarkan kondisi.**
- **Operasi Dasar: Seperti aritmatika, logika, dan manipulasi data.**

4. **Karakteristik Komposisi**

- **Mencakup lebih dari satu jenis struktur kontrol.**
- **Memerlukan pemahaman mendalam tentang logika dan alur program.**
- **Menuntut efisiensi dalam penggunaan sumber daya komputasi.**

5. **Contoh Implementasi**

- **Menampilkan Bilangan Ganjil:**
    - **Input: Bilangan bulat.**
    - **Output: Deret bilangan ganjil hingga bilangan tersebut.**
- **Menentukan Bilangan Terbesar dan Terkecil:**
    - **Input: Tiga bilangan bulat.**
    - **Output: Bilangan terbesar dan terkecil dari ketiga bilangan tersebut.**
- **Menampilkan Faktor Bilangan:**
    - **Input: Bilangan bulat positif.**
    - **Output: Semua faktor bilangan dari angka yang diberikan.**

6. **Keunggulan Komposisi**

- **Mengoptimalkan logika program dengan memanfaatkan kombinasi struktur kontrol.**
- **Menghasilkan solusi algoritmik yang lebih mendekati kebutuhan nyata.**

**CONTOH SOAL**

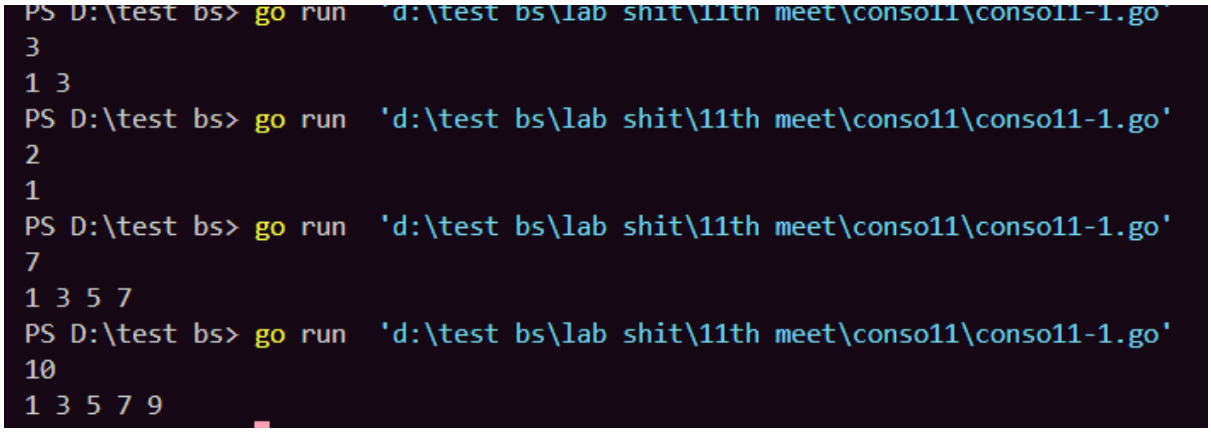
Conso 1:

```
package main

import "fmt"

func main() {
    var bilangan, j int
    fmt.Scan(&bilangan)
    for j = 1; j <= bilangan; j += 1 {
        if j%2 != 0 {
            fmt.Print(j, " ")
        }
    }
}
```

Output:




Deskripsi Program:

Program ini berfungsi untuk mencetak semua bilangan ganjil dari angka 1 hingga bilangan (nilai yang dimasukkan oleh pengguna). Setiap bilangan ganjil dipisahkan dengan spasi.

1. **Deklarasi variabel:**

```
var bilangan, j int
```

- bilangan: Batas atas bilangan yang akan diperiksa (input dari pengguna).
- j: Variabel yang digunakan sebagai penghitung dalam perulangan.

2. Input:

```
fmt.Scan(&bilangan)
```

- Program meminta pengguna untuk memasukkan nilai batas atas bilangan.

3. Perulangan dan Pemeriksaan Bilangan Ganjil:

```
for j = 1; j <= bilangan; j += 1 {
    if j%2 != 0 {
        fmt.Print(j, " ")
    }
```

```
}
```

- **Perulangan:**
  - Dimulai dari j = 1 hingga j <= bilangan
  - Nilai j ditambah 1 setiap iterasi
- **Pemeriksaan Ganjil:**
  - Kondisi j%2 != 0 mengecek apakah j adalah bilangan ganjil
  - Jika kondisi terpenuhi, nilai j dicetak di layar diikuti oleh spasi.

Contoh: Jika pengguna memasukkan

```
10 (input)
1 3 5 7 9 (output)
```

2.cosol2

```
package main
import "fmt"
func main() {
    var b1, b2, b3, max, min int
    fmt.Scan(&b1, &b2, &b3)
    if b1 > b2 {
        max = b1
        min = b2
    } else {
        max = b2
        min = b1
    }
    if max < b3 {
        max = b3
    }
    if min > b3 {
        min = b3
    }
    fmt.Println("Terbesar", max)
    fmt.Println("Terkecil", min)
}
```

Output:

```
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\conso11\conso11-2.go'
1 2 3
Terbesar 3
Terkecil 1
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\conso11\conso11-2.go'
5 5 5
Terbesar 5
Terkecil 5
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\conso11\conso11-2.go'
12 31 -43
Terbesar 31
Terkecil -43
```

Deskripsi Program:
Program ini berfungsi untuk menentukan bilangan terbesar dan terkecil dari tiga bilangan bulat yang dimasukkan oleh pengguna.

1. **Deklarasi variabel:**

```
var b1, b2, b3, max, min int
```

- b1, b2, b3: Tiga bilangan yang akan dibandingkan (input dari pengguna).
- max: Variabel untuk menyimpan nilai terbesar.
- min: Variabel untuk menyimpan nilai terkecil.

2. Input Tiga Bilangan:

```
fmt.Scan(&b1, &b2, &b3)
```

- Program meminta pengguna untuk memasukkan tiga bilangan bulat (b1, b2, b3).

3. Menentukan Nilai Awal max dan min:

```
if b1 > b2 {
   max = b1
   min = b2
} else {
   max = b2
   min = b1
}
```

- Jika b1 > b2, maka b1 dianggap sebagai nilai awal max, dan b2 sebagai min.
- Sebaliknya, jika b2 >= b1, maka b2 menjadi max dan b1 menjadi min.

4. Perbandingan dengan Bilangan Ketiga (b3):

```
if max < b3 {
   max = b3
}
if min > b3 {
   min = b3
}
```

- Bandingkan b3 dengan max dan perbarui nilai max jika b3 lebih besar.
- Bandingkan b3 dengan min dan perbarui nilai min jika b3 lebih kecil.

5. Output:

```
fmt.Println("Terbesar", max)
fmt.Println("Terkecil", min)
```

- Program mencetak nilai `max` (terbesar) dan `min` (terkecil).

Contoh: Jika pengguna memasukkan

```
5 12 8 (input)
Terbesar 12 (output)
Terkecil 5 (output)
```

3.cosol3

```
package main
import "fmt"
func main() {
   var bilangan, j int
   fmt.Scan(&bilangan)
   for j = 1; j <= bilangan; j += 1 {
      if bilangan%j == 0 {
         fmt.Print(j, " ")
      }
   }
}
```

Output:

```
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\conso11\conso11-3.go'
5
1 5
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\conso11\conso11-3.go'
12
1 2 3 4 6 12
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\conso11\conso11-3.go'
20
1 2 4 5 10 20
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\conso11\conso11-3.go'
72
1 2 3 4 6 8 9 12 18 24 36 72
```

Deskripsi Program:

Program ini berfungsi untuk mencari dan mencetak semua faktor dari suatu bilangan yang dimasukkan oleh pengguna.

1. **Deklarasi variabel:**

```
var bilangan, j int
```

- bilangan: Variabel untuk menyimpan bilangan yang akan dicari faktor-faktornya.
- j: Variabel untuk iterasi dalam mencari faktor dari bilangan.

2. Input Pengguna:

```
fmt.Scan(&bilangan)
```

- Program meminta pengguna untuk memasukkan sebuah bilangan (bilangan).

3. Proses Pencarian Faktor:

```
for j = 1; j <= bilangan; j += 1 {
   if bilangan%j == 0 {
      fmt.Print(j, " ")
```

```
    }
}
```

- Loop for dimulai dari j = 1 hingga j = bilangan.
- Pada setiap iterasi, program memeriksa apakah bilangan % j == 0. Jika benar, maka j adalah faktor dari bilangan.
- Faktor-faktor yang ditemukan akan dicetak satu per satu dengan spasi sebagai pemisah.

Contoh: Jika pengguna memasukkan

```
6 (input)

1 2 3 6 (output)
```

**SOAL LATIHAN**

1. Latihan1

Source Code:

```
package main

import "fmt"

func main() {
   var x, y int
   fmt.Scan(&x)
   for i := 1; i <= x; i++ {
      if i%2 != 0 {
         y++
      }
   }
   fmt.Printf("Terdapat %d bilangan ganjil", y)
}
```

Output:

```
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-1.go'
3
Terdapat 2 bilangan ganjil
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-1.go'
2
Terdapat 1 bilangan ganjil
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-1.go'
7
Terdapat 4 bilangan ganjil
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-1.go'
10
Terdapat 5 bilangan ganjil
```

Deskripsi Program:

Program ini berfungsi untuk menghitung jumlah bilangan ganjil dari 1 hingga bilangan yang dimasukkan oleh pengguna.

1. **Deklarasi variabel:**

```
var x, y int
```

- x: Variabel untuk menyimpan bilangan yang dimasukkan oleh pengguna (batas atas perhitungan).
- y: Variabel untuk menghitung jumlah bilangan ganjil.

2. Input Bilangan:

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

- Program meminta pengguna untuk memasukkan sebuah bilangan (x).

3. Perulangan dan Penghitungan Bilangan Ganjil:

```
for i := 1; i <= x; i++ {
```

```
    if i%2 != 0 {
        y++
    }
}
```

- Loop for dimulai dari i = 1 hingga i = x.
- Pada setiap iterasi, program memeriksa apakah i merupakan bilangan ganjil dengan cara i%2 != 0.
- Jika benar, maka y (jumlah bilangan ganjil) akan bertambah 1.

4. Output Hasil:

```
fmt.Printf("Terdapat %d bilangan ganjil", y)
```

- Setelah perulangan selesai, program mencetak jumlah bilangan ganjil yang ditemukan.

Contoh: Jika pengguna memasukkan

10 (input)

Terdapat 5 bilangan ganjil (output)

2. Latihan2

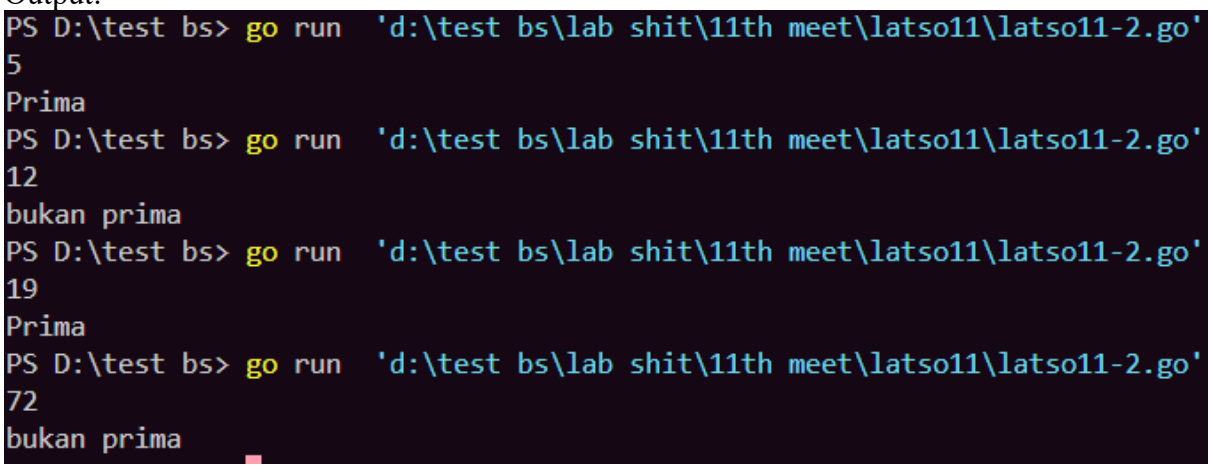
**Source Code:**

```
package main

import "fmt"

func main() {
   var x int
   var y bool
   fmt.Scan(&x)
   for i := 2; i < x; i++ {
      if x%i == 0 {
         y = false
         break
      } else {
         y = true
      }
   }
   if y == true {
      fmt.Print("Prima")
   } else {
      fmt.Print("bukan prima")
   }
}
```

Output:




Deskripsi Program:

Program ini berfungsi untuk menentukan apakah sebuah bilangan yang dimasukkan oleh pengguna adalah bilangan prima atau bukan.

1. **Deklarasi variabel:**

```
var x int
var y bool
```

- x: Variabel untuk menyimpan bilangan yang dimasukkan oleh pengguna.
- y: Variabel boolean yang digunakan untuk menandakan apakah bilangan tersebut prima atau tidak.

2. Input:

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

- Program meminta pengguna untuk memasukkan bilangan yang ingin diperiksa apakah prima atau bukan.

3. Pemeriksaan Bilangan Prima:

```
for i := 2; i < x; i++ {
   if x%i == 0 {
      y = false
      break
   } else {
      y = true
   }
}
```

- Program memulai perulangan for dari i = 2 hingga i < x.
- Pada setiap iterasi, program memeriksa apakah x habis dibagi oleh i. Jika x%i == 0, artinya x memiliki pembagi selain 1 dan dirinya sendiri, sehingga x bukan bilangan prima dan program akan menghentikan perulangan (break).
- Jika tidak ada pembagi yang ditemukan, y di-set menjadi true yang menunjukkan bahwa bilangan tersebut adalah prima.

4. Output:

```
if y == true {
   fmt.Print("Prima")
} else {
   fmt.Print("bukan prima")
}
```

- Setelah perulangan selesai, program akan memeriksa nilai `y`:
    - Jika `y == true`, artinya bilangan tersebut prima, dan program mencetak "Prima".
    - Jika `y == false`, artinya bilangan tersebut bukan prima, dan program mencetak "bukan prima".

Contoh: Jika pengguna memasukkan

| |
|---|
| 7 (input)<br>Prima (output) |

3. Latihan3

**Source Code:**

```
package main

import "fmt"

func main() {
   var m, k, h, u,t string
   var x bool
   m = "merah"
   k = "kuning"
   h = "hijau"
   u = "ungu"
   x = true
   for i := 0; i < 5; i++ {
      fmt.Printf("percobaan %d: ",i+1)
      fmt.Scan(&t)
      if t!=m{
         x=false
      }
      fmt.Scan(&t)
      if t!=k{
         x=false
      }
      fmt.Scan(&t)
      if t!=h{
         x=false
      }
      fmt.Scan(&t)
      if t!=u{
         x=false
      }
   }
   fmt.Print("Berhasil : ",x)
}
```

Output:

```
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-3.go'
percobaan 1: merah kuning hijau ungu
percobaan 2: merah kuning hijau ungu
percobaan 3: merah kuning hijau ungu
percobaan 4: merah kuning hijau ungu
percobaan 5: merah kuning hijau ungu
Berhasil : true
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-3.go'
percobaan 1: merah kuning hijau ungu
percobaan 2: merah kuning hijau ungu
percobaan 3: merah kuning hijau ungu
percobaan 4: ungu kuning hijau merah
percobaan 5: merah kuning hijau ungu
Berhasil : false
```

Deskripsi Program:

Program ini berfungsi untuk memeriksa apakah input warna yang diberikan oleh pengguna selama 5 percobaan sesuai dengan urutan yang benar: "merah", "kuning", "hijau", dan "ungu". Jika semua percobaan menghasilkan urutan yang benar, maka program mencetak "Berhasil: true". Jika salah satu input tidak sesuai urutan, maka hasilnya adalah "Berhasil: false".

1. **Deklarasi variabel:**

```
var m, k, h, u, t string
var x bool
```

- m, k, h, dan u: Variabel string yang menyimpan warna yang benar: "merah", "kuning", "hijau", dan "ungu".
- t: Variabel string untuk menyimpan input dari pengguna pada setiap percobaan.
- x: Variabel boolean untuk menandakan apakah urutan warna yang dimasukkan oleh pengguna sesuai dengan urutan yang benar.

2. Inisialisasi Warna:

```
m = "merah"
k = "kuning"
h = "hijau"
u = "ungu"
x = true
```

- Program menginisialisasi nilai untuk m, k, h, dan u yang masing-masing berisi warna yang benar.
- x diinisialisasi dengan true untuk menandakan bahwa percobaan dimulai dengan kondisi sukses

3. Perulangan Percobaan:

```
for i := 0; i < 5; i++ {
    fmt.Printf("percobaan %d: ", i+1)
    fmt.Scan(&t)
    if t != m {
        x = false
    }
```

```
    fmt.Scan(&t)
    if t != k {
        x = false
    }
    fmt.Scan(&t)
    if t != h {
        x = false
    }
    fmt.Scan(&t)
    if t != u {
        x = false
    }
}
```

- Program melakukan perulangan sebanyak 5 kali.
- Pada setiap iterasi, program meminta pengguna untuk memasukkan 4 warna.
- Setiap warna yang dimasukkan dibandingkan dengan urutan yang benar:
    - Jika warna yang dimasukkan tidak sesuai dengan `m`, `k`, `h`, atau `u`, maka `x` di-set menjadi `false`, menandakan bahwa percobaan tersebut gagal.

4. Output:

```
fmt.Print("Berhasil : ", x)
```

- Setelah 5 percobaan, program mencetak apakah seluruh percobaan berhasil atau tidak, berdasarkan nilai dari x.

Contoh: Jika pengguna memasukkan

```
percobaan 1: merah kuning hijau ungu (input)
percobaan 2: merah kuning hijau ungu (input)
percobaan 3: merah kuning hijau ungu (input)
percobaan 4: merah kuning hijau ungu (input)
percobaan 5: merah kuning hijau ungu (input)
Berhasil : true (output)
```

4. Latihan4

Source Code:

```
package main

import "fmt"

func main() {
   var x, y string
   var z, i int
   i = 1
   for {
      fmt.Printf("Bunga %d:", i)
      fmt.Scan(&x)
      if x == "SELESAI" {
         break
      }
      y = y + x + " - "
      i++
      z++
   }
   fmt.Print("Pita: ", y, "\nBunga: ", z)
}
```

Output:

```
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-4.go'
Bunga 1:Kertas
Bunga 2:Mawar
Bunga 3:Tulip
Bunga 4:SELESAI
Pita: Kertas - Mawar - Tulip -
Bunga: 3
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-4.go'
Bunga 1:SELESAI
Pita:
Bunga: 0
```

Deskripsi Program:

Program ini digunakan untuk meminta input berupa nama bunga dari pengguna secara berulang. Pengguna akan diminta untuk memasukkan nama bunga hingga memasukkan kata "SELESAI". Program kemudian akan mencetak daftar bunga yang telah dimasukkan, dipisahkan oleh tanda " - ", serta menampilkan jumlah bunga yang dimasukkan.

1. **Deklarasi variabel:**

```
var x, y string
var z, i int
```

- x: Variabel string untuk menyimpan nama bunga yang dimasukkan oleh pengguna.
- y: Variabel string untuk menyimpan daftar bunga yang dimasukkan, dipisahkan oleh " - ".

- z: Variabel integer untuk menghitung jumlah bunga yang dimasukkan.
- i: Variabel integer untuk mencatat nomor bunga (urutan).

2. Inisialisasi:

```
i = 1
```

- i diinisialisasi dengan 1, untuk memulai penomoran bunga pertama.

3. Perulangan:

```
for {
   fmt.Printf("Bunga %d:", i)
   fmt.Scan(&x)
   if x == "SELESAI" {
      break
   }
   y = y + x + " - "
   i++
   z++
}
```

- Program menggunakan perulangan `for` tanpa kondisi penghentian, yang artinya perulangan ini akan terus berlangsung hingga mencapai `break`.
- Dalam setiap iterasi:
    - **Program meminta pengguna untuk memasukkan nama bunga dengan mencetak "Bunga {i}:", di mana {i} adalah nomor bunga.**
    - **Jika nama bunga yang dimasukkan adalah "SELESAI", program akan keluar dari perulangan dengan break.**
    - **Jika nama bunga bukan "SELESAI", bunga tersebut akan ditambahkan ke string y yang berfungsi sebagai daftar bunga, dengan pemisah " - ".**
    - **Variabel i akan bertambah untuk melanjutkan nomor bunga berikutnya.**
    - **Variabel z dihitung untuk melacak jumlah bunga yang dimasukkan**

4. Output Hasil:

```
fmt.Print("Pita: ", y, "\nBunga: ", z)
```

- Setelah keluar dari perulangan, program mencetak hasil:
    - **y: Daftar bunga yang dimasukkan, dipisahkan oleh " - ".**
    - **z: Jumlah bunga yang dimasukkan.**

Contoh: Jika pengguna memasukkan

```
Bunga 1: Melati (input)
Bunga 2: Mawar (input)
Bunga 3: Anggrek (input)
Bunga 4: SELESAI (input)
Pita: Melati - Mawar - Anggrek -  (output)
Bunga: 3 (output)
```

5. Latihan5

Source Code:

```
package main

import "fmt"

func main() {
   var x int
   var y, z, t float64
   t = 1
   fmt.Print("Nilai K = ")
   fmt.Scan(&x)
   for i := 0; i <= x; i++ {
      y = float64((4*i)+2) * float64((4*i)+2)
      z = float64((4*i)+1) * float64((4*i)+3)
      t = t * (y / z)
   }
   fmt.Printf("Nilai akar 2 = %.10f", t)
}
```

Output:

```
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-5.go'
Nilai K = 10
Nilai akar 2 = 1.4062058441
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-5.go'
Nilai K = 100
Nilai akar 2 = 1.4133387072
PS D:\test bs> go run  'd:\test bs\lab shit\11th meet\latso11\latso11-5.go'
Nilai K = 1000
Nilai akar 2 = 1.4141252651
```

Deskripsi Program:

Program ini digunakan untuk menghitung nilai akar 2 (√2) menggunakan rumus deret tertentu. Rumus yang digunakan adalah:

$$\sqrt{2} = \prod_{k=0}^{\infty} \frac{(4k+2)^2}{(4k+1)(4k+3)}$$

Dimana:

- Program mengalikan setiap hasil dari rumus tersebut untuk menghitung nilai mendekati akar 2.

1. **Deklarasi variabel:**

```
var x int
var y, z, t float64
```

```
t = 1
```

- x: Variabel integer untuk menyimpan input dari pengguna yang menentukan jumlah iterasi.
- y, z: Variabel float64 untuk menyimpan bagian dari perhitungan deret dalam rumus.
- t: Variabel float64 untuk menyimpan hasil perhitungan hasil deret akar 2. Diinisialisasi dengan nilai 1.

2. Input Nilai K:

```
fmt.Print("Nilai K = ")
fmt.Scan(&x)
```

- Program meminta pengguna untuk memasukkan nilai K yang menentukan jumlah iterasi dalam perhitungan.

3. Perulangan untuk Menghitung Nilai Akar 2:

```
for i := 0; i <= x; i++ {
   y = float64((4*i)+2) * float64((4*i)+2)
   z = float64((4*i)+1) * float64((4*i)+3)
   t = t * (y / z)
}
```

- Program menggunakan perulangan untuk menghitung nilai dari rumus deret dengan iterasi hingga nilai x.
- Di setiap iterasi:
    - **y dihitung sebagai (4i+2)2(4i + 2)^2(4i+2)2.**
    - **z dihitung sebagai (4i+1)(4i+3)(4i + 1)(4i + 3)(4i+1)(4i+3).**
    - **t dikalikan dengan hasil pembagian yz\frac{y}{z}zy di setiap iterasi.**

4. Output Hasil:

```
fmt.Printf("Nilai akar 2 = %.10f", t)
```

- Setelah perulangan selesai, program mencetak nilai dari t, yang merupakan nilai akar 2 yang dihitung dari deret, dengan presisi hingga 10 angka desimal.

Contoh: Jika pengguna memasukkan

```
Nilai K = 5 (input)

Nilai akar 2 = 1.4142135624 (output)
```