LAPORAN PRAKTIKUM

ALGORITMA DAN PEMROGRAMAN 1

MODUL 13

“REPAT-UNTIL”

DISUSUN OLEH:

RAIHAN ADI ARBA

103112400071

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

Dalam bahasa pemrograman Go, struktur Repeat-Until merupakan salah satu konsep perulangan yang meskipun tidak tersedia secara built-in, namun dapat diimplementasikan dengan menggunakan fitur-fitur yang ada di Go. Konsep dasarnya adalah melakukan eksekusi blok kode minimal satu kali sebelum melakukan pengecekan kondisi, yang membedakannya dari struktur perulangan while yang melakukan pengecekan di awal. Implementasi ini sangat berguna ketika kita membutuhkan kepastian bahwa suatu blok kode harus dijalankan setidaknya satu kali, terlepas dari kondisi yang akan dicek nantinya.

Dalam implementasinya menggunakan Go, kita dapat menggunakan perulangan for tanpa kondisi awal yang akan bertindak sebagai infinite loop. Di dalam blok perulangan tersebut, kita menempatkan statement atau perintah yang ingin dieksekusi, kemudian diikuti dengan sebuah kondisi pengecekan menggunakan statement if. Jika kondisi yang ditentukan terpenuhi (bernilai true), maka perulangan akan dihentikan menggunakan keyword break. Struktur ini memungkinkan program untuk menjalankan blok kode terlebih dahulu sebelum melakukan evaluasi terhadap kondisi yang ditetapkan.

Keuntungan menggunakan pendekatan ini dalam Go adalah fleksibilitas yang ditawarkan oleh bahasa pemrograman Go itu sendiri. Meskipun tidak memiliki syntax khusus untuk Repeat-Until seperti bahasa pemrograman Pascal atau Basic, Go menyediakan cara yang elegant untuk mengimplementasikan logika yang sama. Penggunaan for dan break memberikan kontrol yang lebih granular terhadap alur program, dan programmer dapat dengan mudah menambahkan logika tambahan di dalam blok perulangan jika diperlukan.

Dalam praktiknya, implementasi Repeat-Until dalam Go sering digunakan dalam berbagai skenario pemrograman, seperti validasi input user, pemrosesan data yang memerlukan minimal satu kali iterasi, atau dalam kasus-kasus dimana kita perlu memastikan suatu kondisi terpenuhi sebelum program dapat melanjutkan ke tahap berikutnya. Struktur ini juga sangat berguna dalam pengembangan aplikasi yang memerlukan interaksi dengan pengguna, dimana program perlu memastikan bahwa input yang diberikan valid sebelum melanjutkan ke proses selanjutnya. Dengan pemahaman yang baik tentang konsep Repeat-Until dan implementasinya dalam Go, programmer dapat membuat kode yang lebih efisien dan mudah dipahami.

A. UNGUIDED

1. Latihan 1

   Buatlah program yang digunakan untuk menghitung banyaknya digit dari suatu bilangan. Masukan berupa bilangan bulat positif. Keluaran berupa bilangan bulat yang menyatakan banyaknya digit dari bilangan yang diberikan pada masukan.
   Contoh Masukan dan Keluaran:

No	Masukan	Keluaran
1	5	1
2	234	3
3	78787	5
4	1894256	7

Source Code:

```
 package main

import "fmt"

func main() {
	var bilangan, digit int

	digit = 0
	fmt.Scan(&bilangan)

	for bilangan > 0 {
		bilangan = bilangan / 10
		digit++
	}
	fmt.Println(digit)
}
```

Output :

```
raihan@Raihans-MacBook-Pro minggu 13 % go run l1.go
5
1
raihan@Raihans-MacBook-Pro minggu 13 % go run l1.go
234
3
```

Penjelasan Program:

Program ini dirancang untuk menghitung jumlah digit dalam sebuah bilangan bulat. Program dimulai dengan mendeklarasikan dua variabel integer: 'bilangan' untuk menyimpan input dari pengguna dan 'digit' sebagai penghitung jumlah digit. Variabel digit diinisialisasi dengan nilai 0 sebelum program menerima input. Setelah menerima input bilangan, program menggunakan perulangan for yang akan terus berjalan selama bilangan lebih besar dari 0. Di dalam loop, program membagi bilangan dengan 10 (menghapus digit terakhir) dan menambah nilai digit dengan 1. Proses ini berulang sampai semua digit terhitung. Seperti yang terlihat pada contoh eksekusi, ketika input adalah 5, program menampilkan output 1 (karena 5 memiliki satu digit), dan ketika input adalah 234, program menampilkan output 3 (karena 234 memiliki tiga digit). Program ini efektif untuk menentukan panjang atau jumlah digit dari suatu bilangan bulat positif.

2. Latihan 2

Buatlah program yang digunakan untuk mendapatkan bilangan bulat optimal dari bilangan yang telah diinputkan. Keluaran terdiri dari bilangan hasil penjumlahan tiap perulangannya sampai pembulatan keatas dari bilangan yang diinputkansimal. Keluaran terdiri dari bilangan hasil penjumlahan tiap perulangannya sampai pembulatan keatas dari bilangan yang diinputkan.

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	0.2	0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
2	2.7	2.8 2.9 3

Source Code :

```
package main

import "fmt"

func main() {

	var angka, xy float64

	fmt.Scan(&angka)
	xy = float64(int(angka)) + 1

	for selesai := false; !selesai; {
		angka += 0.1
		angka = float64(int(angka*10+0.5)) / 10
		if angka >= xy {
			fmt.Printf("%.0f\n", angka)
			selesai = true
		} else {
			fmt.Printf("%.1f\n", angka)
		}

	}

}
```

Output :

```
raihan@Raihans-MacBook-Pro minggu 13 % go run l2.go
2.7
2.8
2.9
3
raihan@Raihans-MacBook-Pro minggu 13 % go run l2.go
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
```

Pembahasan :

Program ini dirancang untuk menampilkan bilangan desimal dengan increment 0.1 sampai mencapai bilangan bulat terdekat di atasnya. Program dimulai dengan mendeklarasikan dua variabel bertipe float64: 'angka' untuk menyimpan input dari pengguna dan 'xy' untuk menyimpan bilangan bulat terdekat di atas input. Setelah menerima input, program menggunakan perulangan for dengan kondisi selesai yang diinisialisasi false. Di dalam loop, program menambahkan 0.1 ke variabel angka, kemudian melakukan pembulatan untuk menghindari masalah presisi floating-point dengan mengalikan angka dengan 10, menambahkan 0.5 untuk pembulatan, mengkonversi ke integer, dan membagi kembali dengan 10. Jika angka mencapai atau melebihi nilai xy (bilangan bulat terdekat di atas input), program akan menampilkan angka tanpa desimal dan mengakhiri loop dengan mengubah selesai menjadi true. Jika belum mencapai xy, program akan menampilkan angka dengan satu angka desimal. Program ini berguna untuk menampilkan sequence bilangan desimal dengan increment tetap hingga mencapai bilangan bulat tertentu.

3. Latihan 3

Sebuah organisasi amal sedang mengumpulkan donasi untuk mendukung kegiatan sosial mereka. Setiap donatur dapat memberikan sumbangan dalam jumlah tertentu. Program ini akan terus meminta input dari pengguna untuk jumlah donasi hingga total donasi mencapai atau melebihi target yang telah ditentukan. Masukan pada baris pertama berupa bilangan bulat yang merupakan target donasi yang harus dicapai. Masukan pada baris berikut dan seterusnya merupakan bilangan bulat yang menyatakan donasi oleh setiap donatur, masukan terus diterima hingga target tercapai. Keluaran berupa bilangan hasil total penjumlahan tiap perulangannya serta jumlah donatur.

Contoh masukan dan keluaran:

No	Masukan	Keluaran
1	300 100 50 200	Donatur 1: Menyumbang 100. Total terkumpul: 100 Donatur 2: Menyumbang 50. Total terkumpul: 150 Donatur 3: Menyumbang 200. Total terkumpul: 350 Target tercapai! Total donasi: 350 dari 3 donatur.
2	500 150 100 50 300	Donatur 1: Menyumbang 150. Total terkumpul: 150 Donatur 2: Menyumbang 100. Total terkumpul: 250 Donatur 3: Menyumbang 50. Total terkumpul: 300 Donatur 4: Menyumbang 300. Total terkumpul: 600 Target tercapai! Total donasi: 600 dari 4 donatur.
3	200 300	Donatur 1: Menyumbang 300. Total terkumpul: 300 Target tercapai! Total donasi: 300 dari 1 donatur.

Source Code:

```
package main
```

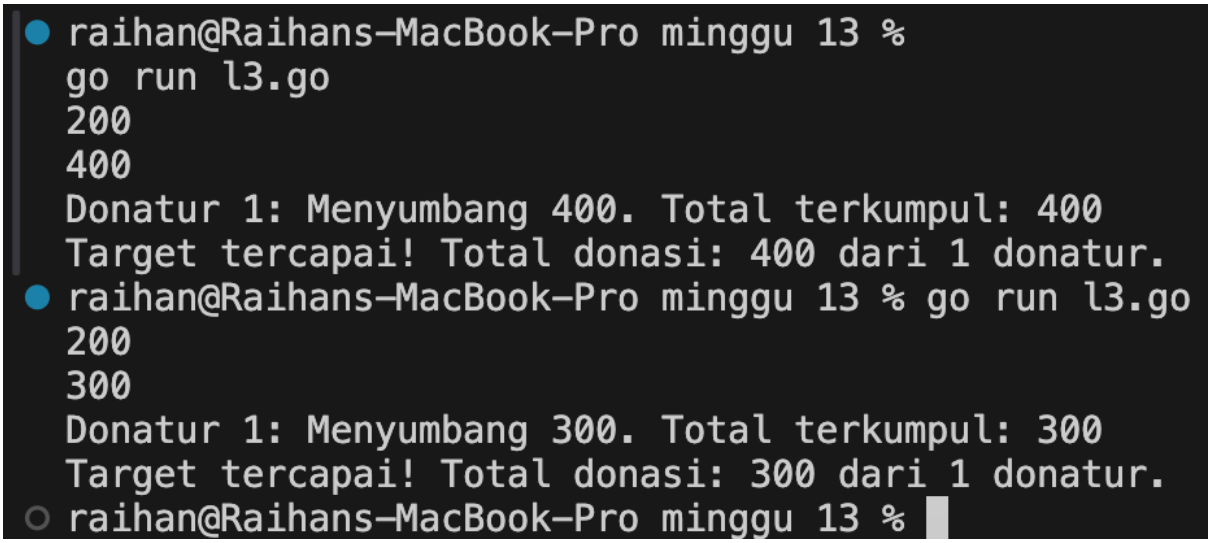
```
import "fmt"

func main() {
	var target, donasi, total, donatur int
	fmt.Scan(&target)
	donatur = 0
	total = 0

	for {
		fmt.Scan(&donasi)
		total += donasi
		donatur++
		fmt.Printf("Donatur %d: Menyumbang %d. Total terkumpul: %d\n",
donatur, donasi, total)

		if total >= target {
			break
		}
	}
	fmt.Printf("Target tercapai! Total donasi: %d dari %d donatur.\n", total,
donatur)
}
```

Output:



Deskripsi Program:

Program ini merupakan sistem pengumpulan donasi yang dibuat menggunakan bahasa pemrograman Go. Program dimulai dengan mengimpor package "fmt" untuk menangani operasi input dan output. Di dalam fungsi main, program mendeklarasikan beberapa variabel yaitu target (jumlah target donasi yang ingin dicapai), donasi (jumlah sumbangan per donatur), total (jumlah keseluruhan donasi), dan donatur (penghitung

jumlah donatur). Program akan meminta input target donasi yang ingin dicapai melalui fmt.Scan(). Selanjutnya, program akan masuk ke dalam perulangan yang akan terus berjalan untuk menerima input donasi dari para donatur. Setiap kali ada donasi masuk, jumlahnya akan ditambahkan ke total dan jumlah donatur akan bertambah satu. Program akan menampilkan informasi setiap ada donasi masuk, yang mencakup nomor donatur, jumlah sumbangannya, dan total donasi yang sudah terkumpul. Perulangan akan terus berlanjut sampai total donasi mencapai atau melebihi target yang telah ditentukan. Ketika target tercapai, program akan menampilkan pesan sukses yang menunjukkan total donasi yang terkumpul dan jumlah donatur yang berpartisipasi.

DAFTAR PUSTAKA

Prayogo, N. A. (2021). ***Dasar Pemrograman Go. Ebook***