LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 1
MODUL 13
REPEAT-UNTIL

DISUSUN OLEH:

MOHAMMAD REYHAN ARETHA FATIN

103112400078

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. Paradigma Perulangan

Perulangan merupakan salah satu struktur kontrol yang memungkinkan suatu instruksi yang sama dilakukan berulang kali dalam waktu atau jumlah yang lama. Tanpa instruksi perulangan, maka suatu instruksi akan ditulis dalam jumlah yang sangat banyak. Pada modul 12 sebelumnya telah dipelajari terkait penggunaan struktur kontrol perulangan dengan while-loop, selanjutnya perulangan juga dapat dilakukan menggunakan repeat-until. Penggunaan **repeat-until** pada dasarnya sama dengan while-loop di mana perulangan berdasarkan kondisi. Perbedaan terletak pada kondisi yang digunakan, pada while-loop kondisi yang harus didefinisikan adalah kondisi perulangannya, atau kapan perulangan itu terjadi, sedangkan pada repeat-until kondisi yang harus didefinisikan merupakan kondisi berhenti, atau kapan perulangan tersebut harus dihentikan.

Kondisi perulangan dan kondisi berhenti memiliki keterhubungan sifat komplemen, sehingga apabila kita mengetahui kondisi perulangannya, maka cukup dengan menambahkan operator negasi atau not untuk mengubah menjadi kondisi berhenti. Hal ini berlaku juga sebaliknya, komplemen dari kondisi berhenti adalah kondisi perulangan.

Pahami beberapa contoh yang diberikan berikut ini:
• Statement while-loop: "Menulis teks tertentu selama tinta pena masih ada".
Statement repeat-until: "Menulis teks tertentu sampai tinta pena habis". Komplemen dari kondisi "tinta pena masih ada" adalah "tinta pena habis".
• Statement while-loop: "Saya makan suap demi suap selama saya masih lapar".
Statement repeat-until: "Saya makan suap demi suap sampai saya merasa kenyang".
Komplemen dari kondisi "saya masih lapar" adalah. "saya merasa kenyang".

2. Karakteristik Repeat-Until
Komponen dari repeat-until sama dengan while-loop, yaitu terdapat kondisi dan aksi, hanya struktur penulisannya saja yang berbeda.
2.1 Aksi, merupakan kumpulan instruksi yang akan dilakukan perulangan. Aksi minimal dijalankan sekali, baru dilakukan pengecekan kondisi berhenti setelahnya. Apabila kondisi bernilai true, maka perulangan dihentikan.
2.2 Kondisi/berhenti, merupakan kondisi berhenti dari perulangan, harus bernilai false selama perulangan dilakukan.

CONTOH SOAL

1. Contoh 1

   Source Code:

```go
coso1 > cososs1.go > ...
1   package main
2
3   import "fmt"
4
5   func main() {
6       var word string
7       var repetitions int
8       fmt.Scan(&word, &repetitions)
9       counter := 0
10      for done := false; !done; {
11          fmt.Println(word)
12          counter++
13          done = (counter >= repetitions)
14      }
15  }
```

   Output:

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTIKUM
pagi 3
pagi
pagi
pagi
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTIKUM
kursi 5
kursi
kursi
kursi
kursi
kursi
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13>
```

   Deskripsi Program:

   Program ini meminta input data dua nilai yaitu sebuah kata (word) dan jumlah perulangan (repetition). Program kemudian mencetak kata tersebut sebanyak perulangan yang di minta.

2. Contoh 2

Source Code:

```
coso2 > coso2.go > main
1   package main
2
3   import "fmt"
4
5   func main() {
6       var number int
7       var continueLoop bool
8       for continueLoop = true; continueLoop; {
9           fmt.Scan(&number)
10          continueLoop = number <= 0
11      }
12      fmt.Printf("%d adalah bilangan bulat positif\n", number)
13  }
```

Output:

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTI
-5
-2
-1
0
5
5 adalah bilangan bulat positif
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTI
17
17 adalah bilangan bulat positif
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13>
```

Deksripsi Program:

Program ini meminta untuk memasukkan sebuah bilangan bulat (number). Selama bilangan yang dimasukkan kurang dari atau sama dengan nol, program akan meminta input ulang dengan menggunakan loop for yang dikontrol oleh variabel boolean continueLoop. Loop ini akan berhenti ketika pengguna memasukkan bilangan positif (lebih besar dari nol). Setelah itu, program mencetak bilangan positif tersebut dengan format yang menyatakan bahwa itu adalah bilangan bulat positif.

3. Contoh 3

Source Code:

```go
package main

import "fmt"

func main() {
    var x, y int
    var selesai bool
    fmt.Scan(&x, &y)
    for selesai = false; !selesai; {
        x = x - y
        fmt.Println(x)
        selesai = x <= 0
    }
    fmt.Println(x == 0)
}

```

Output:

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGO
5
2
3
1
-1
false
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGO
15
3
12
9
6
-1
false
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGO
15
3
12
9
6
9
6
6
3
0
true
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGO
25
5
20
15
10
5
0
true
```

Deksripsi Program:

Program ini meminta input dua bilangan bulat, x dan y. Kemudian, program mengurangi nilai x secara berulang dengan nilai y dalam sebuah loop for. Loop tersebut akan terus berjalan selama kondisi selesai bernilai false. Di dalam loop, nilai x yang baru akan dicetak setiap iterasi, dan kondisi selesai akan berubah menjadi true jika nilai x menjadi kurang dari atau sama dengan 0. Setelah loop selesai, program memeriksa apakah x sama dengan 0 dan mencetak hasilnya (true atau false). Program ini pada dasarnya melakukan pengurangan berulang hingga x menjadi nol atau negatif.

SOAL LATIHAN

1.

Source Code:

```
latsol1 > latsol1.go > main
  1  package main
  2
  3  import "fmt"
  4
  5  func main() {
  6  var angka, hasil int
  7  fmt.Scan(&angka)
  8  for angka > 0 {
  9  hasil++
 10  angka = angka/10
 11  }
 12  fmt.Print(hasil)
 13  }
```

Output:

```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGO
5
1
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGO
234
3
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGO
78787
5
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGO
1894256
7
```

Deskripsi Program:

Program ini menghitung jumlah digit dari bilangan bulat positif yang dimasukkan oleh pengguna. Variabel angka digunakan untuk menyimpan input bilangan, sedangkan variabel hasil digunakan untuk menghitung jumlah digit. Dalam loop for, selama nilai angka lebih besar dari nol, bilangan tersebut dibagi 10 (menghilangkan satu digit paling kanan), dan variabel hasil ditambahkan 1 di setiap iterasi. Loop berlanjut hingga angka menjadi nol. Terakhir, program mencetak nilai hasil, yang menunjukkan jumlah digit dari bilangan yang dimasukkan.

2.

Source Code:

```
latsol2 > latsol2.go > main
  1  package main
  2
  3  import "fmt"
  4
  5  func main() {
  6  var angka float64
  7  var temp int
  8  fmt.Scan(&angka)
  9  temp = int(angka)+1
 10  cek := false
 11  for !cek {
 12  angka = angka + 0.1
 13  fmt.Printf("%.1f\n", angka)
 14  cek = angka > float64(temp)-0.11
 15  }
 16  fmt.Println(temp)
 17  }
```

Output:

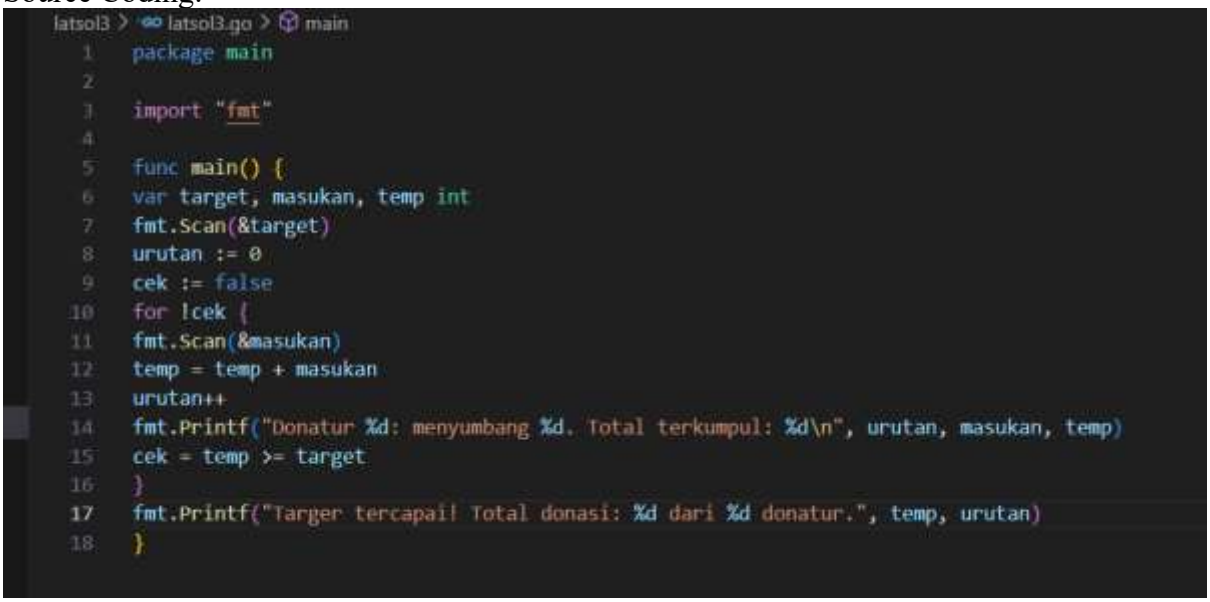
```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTIKUM
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTIKUM
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTIKUM
2.7
2.8
2.9
3
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13>
```

Deksripsi Program:

Program ini membaca input angka dengan tipe data float64 dari pengguna. Variabel temp digunakan untuk menyimpan nilai integer dari angka yang dibulatkan ke atas dengan penambahan 1. Program kemudian menggunakan loop for untuk menaikkan nilai angka sebesar 0.1 pada setiap iterasi, dan mencetak angka tersebut dengan format satu desimal. Kondisi loop berhenti jika nilai angka melebihi nilai temp dikonversi ke float64 dikurangi 0.11. Setelah loop selesai, nilai temp dicetak. Program ini pada dasarnya menaikkan angka secara bertahap hingga hampir mencapai nilai yang dibulatkan ke atas dari input awal.

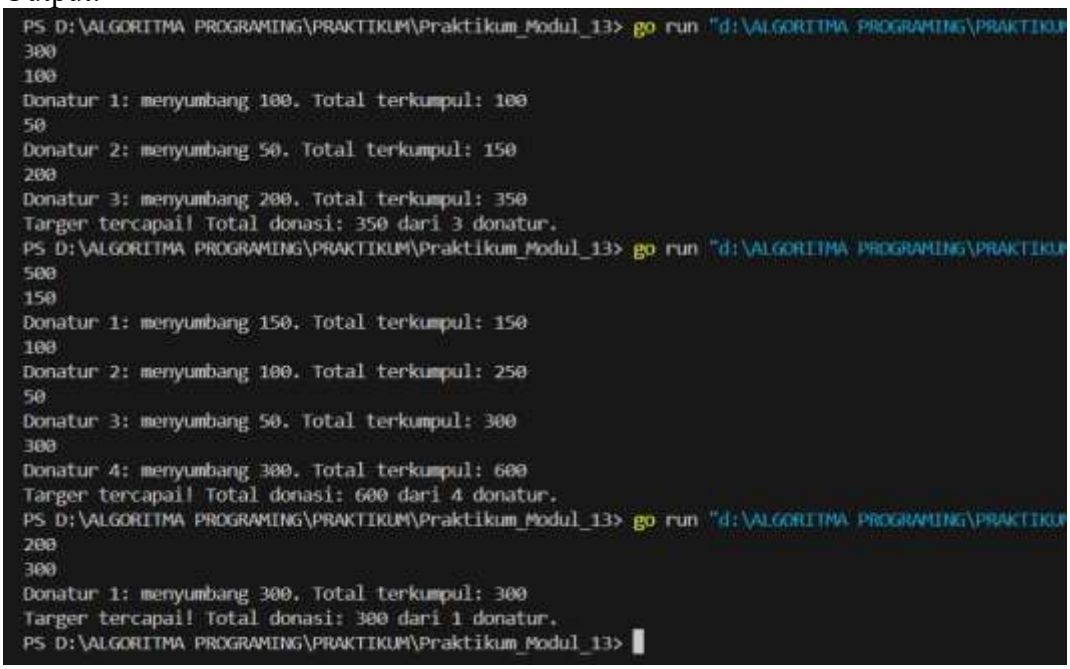
3.

Source Coding:



```
latsol3 > latsol3.go > main
1   package main
2
3   import "fmt"
4
5   func main() {
6   var target, masukan, temp int
7   fmt.Scan(&target)
8   urutan := 0
9   cek := false
10  for !cek {
11  fmt.Scan(&masukan)
12  temp = temp + masukan
13  urutan++
14  fmt.Printf("Donatur %d: menyumbang %d. Total terkumpul: %d\n", urutan, masukan, temp)
15  cek = temp >= target
16  }
17  fmt.Printf("Targer tercapai! Total donasi: %d dari %d donatur.", temp, urutan)
18  }
```

Output:



```
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTIKU
300
100
Donatur 1: menyumbang 100. Total terkumpul: 100
50
Donatur 2: menyumbang 50. Total terkumpul: 150
200
Donatur 3: menyumbang 200. Total terkumpul: 350
Targer tercapai! Total donasi: 350 dari 3 donatur.
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTIKU
500
150
Donatur 1: menyumbang 150. Total terkumpul: 150
100
Donatur 2: menyumbang 100. Total terkumpul: 250
50
Donatur 3: menyumbang 50. Total terkumpul: 300
300
Donatur 4: menyumbang 300. Total terkumpul: 600
Targer tercapai! Total donasi: 600 dari 4 donatur.
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13> go run "d:\ALGORITMA PROGRAMING\PRAKTIKU
200
300
Donatur 1: menyumbang 300. Total terkumpul: 300
Targer tercapai! Total donasi: 300 dari 1 donatur.
PS D:\ALGORITMA PROGRAMING\PRAKTIKUM\Praktikum_Modul_13>
```

Deksripsi Program:

Program ini berfungsi untuk menghitung total donasi hingga mencapai target yang ditentukan. Pertama, pengguna memasukkan nilai target donasi. Program kemudian meminta input nilai donasi secara berulang melalui loop for yang berhenti ketika total donasi (temp) mencapai atau melebihi target. Setiap kali donasi dimasukkan, program menambahkan nilai donasi ke total sementara (temp), mencatat jumlah donatur (urutan), dan mencetak rincian donatur ke-n, jumlah yang disumbangkan, dan total donasi saat ini. Setelah target tercapai, program mencetak pesan "Target tercapai!" bersama dengan total donasi yang dikumpulkan dan jumlah donatur yang berkontribusi.

DAFTAR PUSTAKA

Prasti Eko Yunanto, S.T., M.Kom.

MODUL PRAKTIKUM 13-REPEAT-UNTIL ALGORITMA DAN PEMOGRAMAN 1 S1 INFORMATIKA