LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN 2

MODUL 5

Rekursif

Oleh:

Achmad Zulvan Nur Hakim

103112400070

IF-12-01

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

Rekursif secara sederhana dapat diartikan sebagai cara menyelesaikan suatu masalah dengan cara menyelesaikan sub-masalah yang identik dari masalah utama. Oleh karena itu bisanya ditambahkan struktur kontrol percabangan (if-then) untuk menghentikan proses rekursif ini. Kondisi ini disebut juga dengan base-case, artinya apabila kondisi base-case bernilai true maka proses rekursif akan berhenti.

• Teknik rekursif ini merupakan salah satu alternatif untuk mengganti struktur kontrol perulangan dengan memanfaatkan subprogram (bisa fungsi ataupun prosedure).

• Untuk menghentikan proses rekursif digunakan percabangan (if-then).

• Base-case adalah kondisi proses rekursif berhenti. Base-case merupakan hal terpenting dan pertama yang harus diketahui ketika akan membuat program rekursif. Mustahil membuat program rekursif tanpa mengetahui base-case terlebih dahulu.

• Recursive-case adalah kondisi dimana proses pemanggilan dirinya sendiri dilakukan. Kondisi recursive-case adalah komplemen atau negasi dari base-case.

• Setiap algoritma rekursif selalu memiliki padanan dalam bentuk algoritma interatif.

Algoritma rekursif terdiri dari dua komponen utama:

• Base-case (Basis), yaitu bagian untuk menghentikan proses rekursif dan menjadi komponen terpenting di dalam sebuah rekursif.

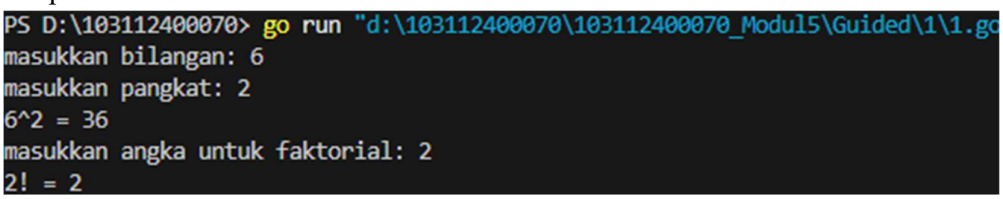
• Recursive-case, yaitu bagian pemanggilan subprogramnya.

II. GUIDED

Code 1:

```
package main

import "fmt"

func pangkatIteratif(base, exp int) int {
	hasil := 1
	for i := 0; i < exp; i++ {
		hasil *= base
	}
	return hasil
}

func faktorialIteratif(n int) int {
	hasil := 1
	for i := 2; i <= n; i++ {
		hasil *= i
	}
	return hasil
}

func main() {
	var base, exp, n int

	fmt.Print("masukkan bilangan: ")
	fmt.Scanln(&base)
	fmt.Print("masukkan pangkat: ")
	fmt.Scanln(&exp)

	fmt.Printf("%d^%d = %d\n", base, exp, pangkatIteratif(base, exp))

	fmt.Print("masukkan angka untuk faktorial: ")
	fmt.Scanln(&n)

	fmt.Printf("%d! = %d\n", n, faktorialIteratif(n))
}
```

Output:



Penjelasan:

Program ini menghitung hasil perpangkatan dan faktorial menggunakan metode iteratif. Pengguna memasukkan bilangan dan pangkat untuk dihitung, serta angka untuk faktorial.

Code 2:

```
package main

import "fmt"

func pangkatRekursif(base, exp int) int {

	if exp == 0 {
		return 1
	}
	return base * pangkatRekursif(base, exp-1)
}

func faktorialRekursif(n int) int {

	if n == 0 || n == 1 {
		return 1
	}
	return n * faktorialRekursif(n-1)
}

func main() {
	var base, exp, n int
	fmt.Print("masukkan bilangan: ")
	fmt.Scanln(&base)
	fmt.Print("masukkan pangkat: ")
	fmt.Scanln(&exp)

	fmt.Printf("%d^%d = %d\n", base, exp, pangkatRekursif(base, exp))
	fmt.Print("masukkan angka untuk faktorial: ")
	fmt.Scanln(&n)

	fmt.Printf("%d! = %d\n", n, faktorialRekursif(n))
}
```

Output:

```
PS D:\103112400070> go run "d:\103112400070\103112400070_Modul5\Guided\2\2.go"
masukkan bilangan: 6
masukkan pangkat: 2
6^2 = 36
masukkan angka untuk faktorial: 4
4! = 24
```

Penjelasan:

Program ini menghitung hasil perpangkatan dan faktorial menggunakan metode rekursif. Pengguna memasukkan bilangan dan pangkat untuk dihitung, serta angka untuk factorial.

III. UNGUIDED

Code 1:

```
// Achmad Zulvan Nur Hakim 103112400070
package main

import "fmt"


func deret(n int) int {
	if n == 0 {
		return 0
	} else if n == 1 {
		return 1
	}
	return deret(n-1) + deret(n-2)
}

func main() {
	var x int
	fmt.Scan(&x)
	fmt.Print("Deret :")
	for i := 0; i <= x; i++ {
		fmt.Print(deret(i), " ")
	}
}
```

Output:

```
PS D:\103112400070> go run "d:\103112400070\103112400070_Modul5\Unguided\1\1.go
10
Deret :0 1 1 2 3 5 8 13 21 34 55
```

Penjelasan:

Program ini menggunakan metode rekursf untuk menghitung deret Fibonacci hingga suku ke-x yang dimasukkan oleh pengguna. Hasilnya kemudian dicetak dalam urutan dari suku pertama hingga suku ke-x.

Code 2:

```
// Achmad Zulvan Nur Hakim 103112400070
package main

import "fmt"

func bintang(n, x int) {
	if x > n {
		return
	}
	for i := 0; i < x; i++ {
		fmt.Print("*")
	}
	fmt.Println()

	bintang(n, x+1)
}

func main() {
	var n int
	fmt.Scan(&n)

	bintang(n, 1)
}
```

Output:

```
PS D:\103112400070> go run "d:\103112400070\103112400070_Modul5\Unguided\2\2.go"
3
*
**
***
PS D:\103112400070> go run "d:\103112400070\103112400070_Modul5\Unguided\2\2.go"
5
*
**
***
****
*****
```

Penjelasan:
Program ini menggunakan metode rekursif untuk mencetak pola segitiga bintang dengan jumlah baris n yang dimasukkan oleh pengguna. Setiap baris memiliki jumlah bintang yang bertambah secara bertahap dari 1 hingga n.

Code 3:

```
//Achmad Zulvan Nur Hakim 103112400070
package main

import "fmt"

func faktor(x, i int) {
	if i > x {
		return
	}
	if x%i == 0 {
		fmt.Print(i, " ")
	}
	faktor(x, i+1)
}

func main() {
	var x int
	fmt.Scan(&x)
	fmt.Print("Faktor ", x, " : ")
	faktor(x, 1)
}
```

Output:

```
PS D:\103112400070> go run "d:\103112400070\103112400070_Modul5\Unguided\3\3.go"
12
Faktor 12 : 1 2 3 4 6 12
```

Penjeasan:

Program ini menggunakan metode rekursif untuk mencari dan mencetak semua faktor dari bilangan x yang dimasukkan oleh pengguna. Rekursif berjalan dari 1 hingga x, mencetak setiap bilangan yang habis membagi x.

IV. KESIMPULAN

Praktikum ini menunjukkan bahwa rekursi merupakan teknik pemrograman yang efektif dalam menyelesaikan berbagai permasalahan komputasi. Dengan menggunakan rekursif, kode menjadi lebih sederhana dan mudah dipahami, meskipun perlu mempertimbangkan optimasi agar tidak mengalami stack overflow akibat pemanggilan fungsi yang terlalu dalam.

V. REFERENSI

MODUL 5. Rekursif Algoritma dan Pemrograman 2