LAPORAN PRAKTIKUM

ALGORITMA DAN PEMROGRAMAN 2

MODUL

“REKURSIF”

DISUSUN OLEH:

SHEILA STEPHANIE ANINDYA

103112400086

S1 IF-12-01

S1 TEKNIK INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

I. DASAR TEORI

A. Penjelasan

Rekursif dalam pemrograman adalah teknik di mana sebuah fungsi memanggil dirinya sendiri untuk menyelesaikan masalah dengan membagi masalah tersebut menjadi submasalah yang lebih kecil.

Rekursi dapat membantu mengurangi jumlah kode yang dibutuhkan, namun dalam beberapa kasus, penggunaan rekursi bisa lebih mahal secara memori dan performa dibandingkan dengan pengulangan (loop). Namun, ada beberapa kasus di mana rekursi bisa digunakan untuk menyelesaikan masalah yang lebih kompleks, yang tidak bisa atau sulit diselesaikan dengan loop biasa. Dalam pemrograman, rekursif sering digunakan untuk menyelesaikan permasalahan yang membutuhkan pemecahan matematika dan ilmu komputer . Namun, perlu diingat bahwa penggunaan rekursi harus dilakukan dengan bijak dan memperhatikan efisiensi dan memori, serta memahami algoritma yang menggunakan rekursi secara tidak langsung. Rekursif adalah teknik pemrograman yang memecah suatu masalah menjadi masalah yang lebih kecil yang serupa dengan masalah semula. Dalam konteks ini, rekursif melibatkan dua komponen penting:

- Base Case: Kondisi yang menentukan kapan fungsi rekursif harus berhenti memanggil dirinya sendiri.
- Recursive Call: Pemanggilan fungsi rekursif di dalam fungsi itu sendiri, yang mengurangi masalah menjadi submasalah yang lebih kecil.

B. Penerapan Rekursif

Fungsi rekursif digunakan dalam berbagai jenis masalah, seperti pengurutan, pencarian, pemecahan masalah kombinatorial, dan lainnya. Beberapa jenis rekursif, seperti rekursif langsung, rekursif

tidak langsung, rekursif tail, rekursif head, rekursif nested, dan rekursif tree.

Dalam pemrograman, rekursif sering digunakan untuk menyelesaikan permasalahan yang membutuhkan pemecahan matematika dan ilmu komputer. Contoh penerapan rekursif yang populer adalah menghitung faktorial dan deret Fibonacci.

1. Faktorial: Menghitung faktorial dari sebuah bilangan n dapat didefinisikan secara rekursif sebagai:
   - n! = n * (n-1)! untuk n > 0
   - 0! = 1
2. Bilangan Fibonacci: Deret Fibonacci juga dapat didefinisikan secara rekursif:
   - Fib (n) = Fib (n - 1) + Fib (n - 2) untuk n $\geq$ 2
   - Fib (0) = 0 dan Fib (1) = 1

C. Kelebihan dan Kekurangan Rekursif

a) Kelebihan:
   1. Kode lebih ringkas dan mudah dipahami.
   2. Memudahkan pemecahan masalah yang kompleks dengan membaginya menjadi bagian yang lebih kecil.

b) Kekurangan:
   1. Penggunaan memori yang lebih tinggi karena setiap panggilan fungsi menambah tumpukan panggilan.
   2. Potensi untuk menyebabkan stack overflow jika kedalaman rekursi terlalu besar.

II. GUIDED

1. Source Code:

```
package main

import "fmt"

//fungsi iteratif untk menghitung pangkat (base ^ exp)
func pangkatIteratif(base, exp int) int {
    hasil := 1
    for i := 0; i < exp; i++ {
        hasil *= base

    }

    return hasil
}

// fungsi untk menghitung faktorial (n!)
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("bilangan: ")
    fmt.Scanln(&base)
    fmt.Print("pangkat: ")
    fmt.Scanln(&exp)

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

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

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

}
```

Output:

```
PS C:\Sheila Stephanie Anindya\SMT 2\Alpro 2\code> go run "c:\Sheila

bilangan: 4
pangkat: 2
4^2 = 16
angka untuk faktorial: 5
5! = 120
```

Penjelasan :

Program ini menghitung pangkat dari sebuah bilangan dan factorial daru sebuah angka, dengan menggunakan metode iterative di mana menggunakan loop untuk melakukan suatu proses berulang kali sampai kondisi tertentu terpenuhi.

2. Source Code:

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

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

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

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

Output:

```
PS C:\Sheila Stephanie Anindya\SMT 2\Alpro 2\code> go run "c:\Sheila

bilangan: 4
pangkat: 2
4^2 = 16
angka untuk faktorial: 5
5! = 120
```

Penjelasan :

Program ini menghitung pangkat dari sebuah bilangan dan factorial daru sebuah angka, dengan menggunakan metode rekursif di mana sebuah fungsi memanggil dirinya sendiri untuk menyelesaikan suatu masalah.

III. UNGUIDED

1. Source Code:

```
package main

import "fmt"
func fibonacci(n int) int {
    if n == 0 {
        return 0
    } else if n == 1 {
        return 1
    }
    return fibonacci(n-1) + fibonacci(n-2)
}
func main() {
    var n int

    fmt.Print("n : ")
    fmt.Scanln(&n)

    fmt.Print("Sn: ")
    for i := 0; i <= n; i++ {
        fmt.Print(fibonacci(i))
        if i < n {
            fmt.Print(" ")
        }
    }
}
```

Output:

```
PS C:\Sheila Stephanie Anindya\SMT 2\Alpro 2\code> go run "c:\Sheila S
"
n : 10
Sn: 0 1 1 2 3 5 8 13 21 34 55
PS C:\Sheila Stephanie Anindya\SMT 2\Alpro 2\code>
```

Penjelasan :

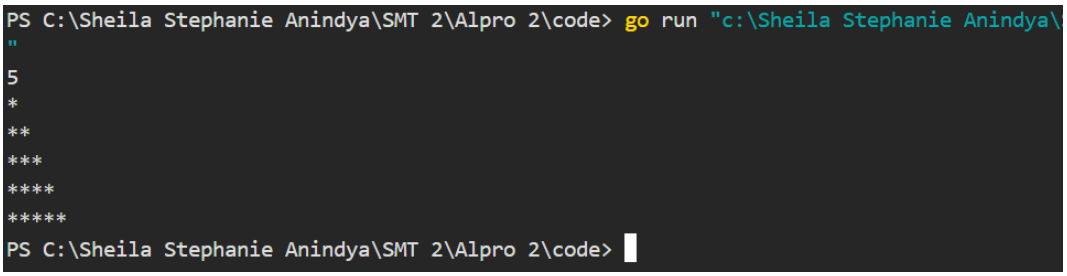
Program ini menghitung deret Fibonacci menggunakan metode rekursif. Deret Fibonacci adalah urutan angka di mana setiap angka adalah hasil penjumlahan dari dua angka sebelumnya.

2. Source Code:

```
package main

import "fmt"

// rekursif cetak bintang
func cetakBintang(n int) {
    if n == 0 {
        return
    }

    cetakBintang(n - 1) //buat baris sblmnya

    for i := 0; i < n; i++ {
        fmt.Print("*")
    }
    fmt.Println() // baris baru
}
```

```
func main() {
    var n int

    fmt.Scanln(&n)

    cetakBintang(n)
}
```

Output:



Penjelasan :

Program ini mencetak bintang setitiga siku – siku berdasarkan angka yang diinput. Dalam program ini, setiap panggilan rekursif menangani satu baris bintang, dan panggilan tersebut berkurang hingga mencapai basis kasus (n = 0).

3. Source Code:

```
package main

import "fmt"

func faktorBilangan(n, i int) {
    if i > n { // baris rekursif jika i lebih besar dari n, berenti
```

```
        return
    }

    if n%i == 0 { // jika i adalah faktor dari n, cetak
        fmt.Print(i, " ")
    }

    faktorBilangan(n, i+1) // rekursif dengan nilai i bertambah
}

func main() {
    var n int

    fmt.Scanln(&n)

    faktorBilangan(n, 1) // pengecekan dari 1
    fmt.Println()        // baris baru
}
```

Output:

```
PS C:\Sheila Stephanie Anindya\SMT 2\Alpro 2\code> go run "c:\Sheila Stephanie Anindya\
"
5
1 5
PS C:\Sheila Stephanie Anindya\SMT 2\Alpro 2\code> go run "c:\Sheila Stephanie Anindya\
"
12
1 2 3 4 6 12
PS C:\Sheila Stephanie Anindya\SMT 2\Alpro 2\code>
```

Penjelasan :

Program ini mencetak factor dari bilangan bulat yang diinput. Faktor dari sebuah bilangan adalah angka-angka yang dapat membagi bilangan tersebut tanpa menyisakan sisa.

IV. KESIMPULAN

Rekursif adalah teknik pemrograman yang memecah suatu masalah menjadi masalah yang lebih kecil yang serupa dengan masalah semula. Dasar-dasar rekursif terdiri dari basiscase dan rekursif case. Rekursif dapat digunakan untuk menyelesaikan berbagai macam masalah, termasuk faktorial, fibonacci, pencarian, dan pembagian. Rekursif memiliki kelebihan dan kekurangan, yaitu: elegan dan mudah dipahami, efisien untuk masalah yangdapat dibagi menjadi masalah yang lebih kecil yang serupa, tetapi sulit untuk didebug dan dapat menghabiskan memori.

V. REFERENSI

1. Fungsi Rekursif. Academia. https://www.academia.edu/31869854/Fungsi_Rekursif
2. Algoritma Rekursif Menggunakan Bahasa Assembly dan Bahasa C. Academia. https://www.academia.edu/43388938/Algoritma_Rekursif_Menggunakan_Bahasa_Assembly_dan_Bahasa_C
3. Jurnal Rekursif. Universitas Bengkulu. https://ejournal.unib.ac.id/rekursif