Tugas 6 Desain Analisis Algoritma

Nama: Ahmad

Kelas: Tekom B 23

NIM: 230210501020

1. Membuat Ringkasan

A. Pemahaman dan Aplikasi Rekursi dalam Python

Di dunia pemrograman, rekursi adalah metode di mana sebuah fungsi akan memanggil dirinya sendiri. Teknik ini berguna untuk menyederhanakan masalah yang rumit dengan membaginya menjadi masalah-masalah yang lebih kecil.

B. Pengertian dan Komponen Rekursi

terdiri dari dua elemen utama yitu :

1. Recurrence : Bagian dalam fungsi yang memanggil ulang fungsi itu sendiri, misalnya, faktorial(n) yang memanggil faktorial(n-1).

2. Base Case : Kondisi dimana untuk menghentikan pemanggilan berulang. Dalam kasus faktorial(n), kondisi ini adalah ketika n bernilai 1.

C. Contoh Penggunaan Rekursi: Fungsi Faktorial

Faktorial dari suatu bilangan n adalah hasil perkalian berturut-turut dari bilangan n hingga 1. Contoh: 5*4*3*2*1. Berikut implementasi rekursifnya dalam Python:

python

```
def faktorial(n):
    if n == 1:  # Kondisi Berhenti
        return 1
    else:
        return n * faktorial(n-1)  # Pemanggilan Ulang
```

Jika kita memanggil faktorial(5), prosesnya adalah sebagai berikut:
- faktorial(5) memanggil 5 * faktorial(4)
- faktorial(4) memanggil 4 * faktorial(3), dan seterusnya
- Ketika faktorial(1) dipanggil, fungsi berhenti dan mulai mengembalikan hasil untuk operasi sebelumnya.

Contoh Implementasi Lain: Deret Fibonacci
Rekursi juga sangat berguna untuk menghitung angka dalam deret Fibonacci, di mana setiap angka merupakan jumlah dari dua angka sebelumnya. Berikut implementasi rekursifnya:

python

```
def fibonacci(n):
```

```
    if n <= 1:
        return n
    else:
        return fibonacci(n-1) + fibonacci(n-2)
```

Misalnya, fibonacci(5) akan mengembalikan angka kelima dalam deret Fibonacci menggunakan pola rekursif di atas.

Kelebihan dan Kekurangan Rekursi
Fungsi rekursif memiliki beberapa kelebihan dan kekurangan:

Kelebihan
- Kode Lebih Ringkas: Rekursi memungkinkan kode lebih singkat dan mudah dipahami, terutama untuk masalah dengan sub-masalah berulang.
- Cocok untuk Struktur Hierarkis: Banyak struktur data seperti pohon atau grafik cocok diproses dengan rekursi.
- Pembagian Masalah: Masalah kompleks dapat dipecah menjadi sub-masalah yang lebih sederhana.

Kekurangan
- Penggunaan Memori Lebih Besar: Setiap pemanggilan fungsi rekursif membutuhkan memori baru, yang bisa memakan banyak memori.
- Menyulitkan untuk Dibaca dan Debugging: Sifat rekursifnya bisa menyulitkan dalam melacak kesalahan.
- Tidak Efisien untuk Kasus Besar: Pada perhitungan besar, seperti deret Fibonacci, rekursi bisa lambat karena banyaknya pemanggilan ulang.

Penyelesaian Nilai Rekurens

1. an = 2an-1; a0 = 3
- Persamaan Karakteristik: r - 2 = 0
- Akar Karakteristik: r = 2
- Solusinya adalah: an = a1 . 2n
- Menggunakan kondisi awal a0 = 3: a0 = 3 = a1 . 20 = a1
- Jadi, solusi relasi rekurens ini adalah an = 3 . 2n.

2. an = 5an-1 - 6an-2; a0 = 1 dan a1 = 0
- Persamaan Karakteristik: r^2 - 5r + 6 = 0
- Akar Karakteristik: (r - 2)(r - 3) = 0 sehingga r1 = 2 dan r2 = 3
- Maka, an = a1 . 2n + a2 . 3n
- Dari kondisi awal a0 = 1: a0 = 1 = a1 + a2
- Dari kondisi a1 = 0: 0 = 2a1 + 3a2
- Memecahkan kedua persamaan ini, didapatkan a1 = 3 dan a2 = -2
- Jadi, solusi relasi rekurensnya adalah an = 3 . 2n - 2 . 3n.

Pertanyaan tentang Fungsi Rekursif
Mengapa kondisi dasar (base case) penting dalam rekursi, dan bagaimana kita memastikan kondisi ini berfungsi dengan benar untuk menghentikan pemanggilan fungsi secara berulang?