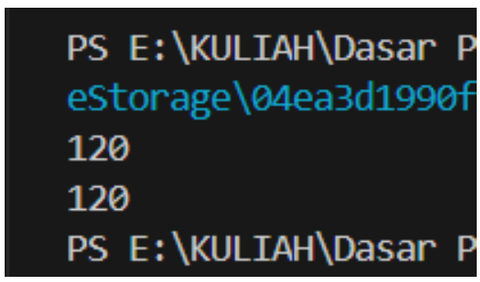
Nama : Wahyudi Satriawan Hamid

Nim : 244107020137

Percobaan 1

```java
Percobaan1.java > Percobaan1
public class Percobaan1 {
    static int faktorialRekursif(int n) {
        if (n == 0) {
            return (1);
        } else {
            return (n * faktorialRekursif(n - 1));
        }
    }

    static int faktorialIteratif(int n) {
        int faktor = 1;
        for (int i = n; i >= 1; i--) {
            faktor = faktor * i;
        }
        return faktor;
    }

    Run | Debug
    public static void main(String[] args) {
        System.out.println(faktorialRekursif(n:5));
        System.out.println(faktorialIteratif(n:5));
    }
}
```

Verifikasi Hasil Percobaan 1



```
PS E:\KULIAH\Dasar P
eStorage\04ea3d1990f
120
120
PS E:\KULIAH\Dasar P
```

Pertanyaan 1

1. Fungsi rekursif adalah fungsi yang memanggil dirinya sendiri selama eksekusi hingga mencapai kondisi tertentu (disebut kasus dasar) yang akan menghentikan rekursi. Fungsi ini biasanya digunakan untuk menyelesaikan masalah yang dapat dipecah menjadi submasalah yang lebih kecil dan serupa dengan masalah awal.
2. Contoh kasus penggunaan fungsi rekursif:
    - Menghitung faktorial: seperti pada kode yang Anda berikan.
    - Fibonacci sequence: Menghitung nilai Fibonacci ke-n.
    - Traversal struktur data tree: Rekursi sering digunakan untuk menelusuri pohon biner.
    - Permutasi dan kombinasi: Untuk menghitung semua kemungkinan urutan elemen.
3. Ya, hasil yang diberikan oleh kedua fungsi adalah sama, karena keduanya menghitung faktorial bilangan dengan cara yang benar.
    - **Fungsi Rekursif:** Fungsi ini memecah masalah menjadi submasalah yang lebih kecil melalui pemanggilan fungsi dirinya sendiri.
    - **Fungsi Iteratif:** Fungsi ini menggunakan loop untuk menghitung faktorial secara langsung.

Percobaan 2

```java
import java.util.Scanner;
public class Percobaan2 {
    static int hitungPangkat (int x, int y) {
        if (y == 0) {
            return (1);
        } else {
            return (x * hitungPangkat(x, y - 1));
        }
    }
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int bilangan, pangkat;

        System.out.print("Bilangan yang dihitung: ");
        bilangan = sc.nextInt();
        System.out.print("Pangkat: ");
        pangkat = sc.nextInt();
        System.out.println(hitungPangkat(bilangan, pangkat));
    }
}

```

Verifikasi Hasil Percobaan 2

```
Bilangan yang dihitung: 10
Pangkat: 2
100
```

Pertanyaan 2

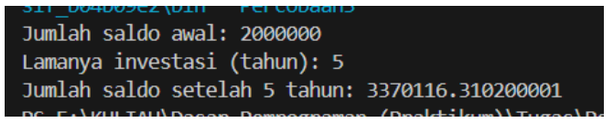
1. Proses pemanggilan fungsi rekursif hitungPangkat akan dijalankan hingga mencapai basis kasus. Dalam program ini, basis kasus adalah ketika y == 0, di mana fungsi akan langsung mengembalikan nilai 1 tanpa memanggil dirinya sendiri lagi.
2.

```
Bilangan yang dihitung: 2
Pangkat: 5
2x2x2x2x2x1 = 32
PS E:\KULIAH\Dasar Pemrogram
```

Percobaan 3

```java
import java.util.Scanner;

public class Percobaan3 {
    static double hitungLaba(double saldo, int tahun) {
        if (tahun == 0) {
            return (saldo);
        } else {
            return (1.11 * hitungLaba(saldo, tahun - 1));
        }
    }
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double saldoAwal;
        int tahun;

        System.out.print("Jumlah saldo awal: ");
        saldoAwal = sc.nextDouble();
        System.out.print("Lamanya investasi (tahun): ");
        tahun = sc.nextInt();

        System.out.print("Jumlah saldo setelah " + tahun + " tahun: ");
        System.out.print(hitungLaba(saldoAwal, tahun));
    }
}

```

Verifikasi Hasil Percobaan 3



Pertanyaan 3

1. Pada Percobaan3, sebutkan blok kode program manakah yang merupakan “base case” dan “recursion call”!
    - Base Case: Basis kasus adalah kondisi yang menghentikan rekursi.

      if (tahun == 0) {

      return (saldo);

      }
    - Recursion Call: Pemanggilan rekursif terjadi ketika fungsi memanggil dirinya sendiri untuk mengulangi proses.

      return (1.11 * hitungLaba(saldo, tahun - 1));
2. Jabarkan trace fase ekspansi dan fase subtitusi algoritma perhitungan laba di atas jika diberikan nilai hitungLaba(100000,3)
    a. Fase Ekspansi: Fungsi memecah permasalahan besar menjadi bagian-bagian yang lebih kecil dengan memanggil dirinya sendiri.

    hitungLaba(100000, 3)

    = 1.11 * hitungLaba(100000, 2)

    = 1.11 * (1.11 * hitungLaba(100000, 1))

    = 1.11 * (1.11 * (1.11 * hitungLaba(100000, 0)))

    = 1.11 * (1.11 * (1.11 * 100000)) // Base case terpenuhi

    b. Fase Substitusi: Nilai mulai dihitung dari hasil base case dan dikembalikan ke level rekursi sebelumnya.

    = 1.11 * (1.11 * (1.11 * 100000))

    = 1.11 * (1.11 * 111000) // Menghitung 1.11 * 100000

    = 1.11 * 123210 // Menghitung 1.11 * 111000

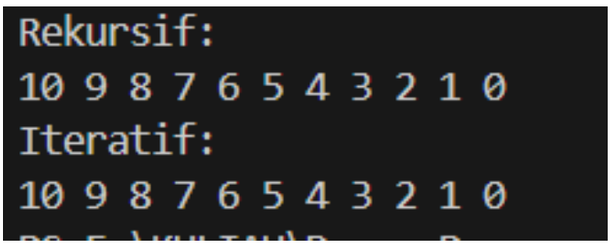
    = 136743 // Menghitung 1.11 * 123210

    c. Hasil Akhir: **hitungLaba(100000, 3) = 136743**

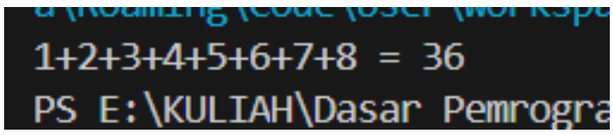
Tugas

https://github.com/Wahyudi-Satriawan-1B-TI/daspro-jobsheet12-Rekursif

1.



2.



3.