Jobsheet 12 Fungsi Rekursif

Nama : **Salsabila Widyadhana**

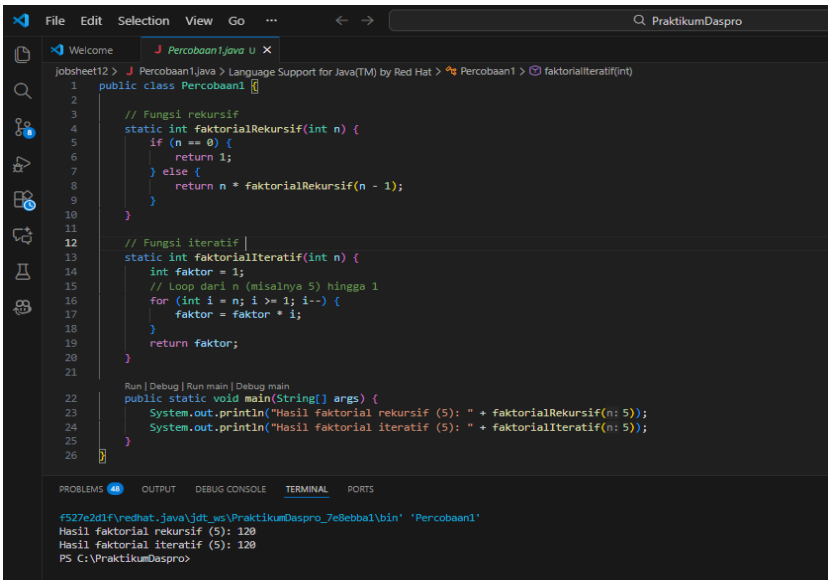
NIM : **254107020200**

Kelas : **TI-1H**

Mata Kuliah : **Praktikum Daspro**

Pertemuan Ke- : **15**

1. Percobaan 1



Commit Github Percobaan 1 :

Pertanyaan :

1. Apa yang dimaksud dengan fungsi rekursif?
2. Bagaimana contoh kasus penggunaan fungsi rekursif ?
3. Pada Percobaan1, apakah hasil yang diberikan fungsi faktorialRekursif() dan fungsi faktorialIteratif() sama? Jelaskan perbedaan alur jalannya program pada penggunaan fungsi rekursif dan fungsi iteratif!

Jawab :

1. Fungsi rekursif adalah sebuah **fungsi yang memanggil dirinya sendiri** secara berulang. Proses pemanggilan ini akan terus berlanjut hingga kondisi berhenti (disebut *base case*) tercapai, yang mana akan menghentikan rekursi dan memulai proses pengembalian nilai.
2. Deret Fibonacci**:** Menghitung suku ke-n di mana setiap suku adalah hasil penjumlahan dua suku sebelumnya, **Struktur Data:** Melakukan penelusuran (traversal) pada struktur data rekursif seperti *Tree* (Pohon) atau *Graph* (Graf).
3. Kesamaan Hasil : Hasil yang diberikan oleh fungsi faktorialRekursif() dan faktorialIteratif() adalah sama**.** Keduanya menghitung n!, menghasilkan 120 untuk input 5.

Perbedaan Alur Program:

Fungsi Rekursif (Menggunakan Stack):

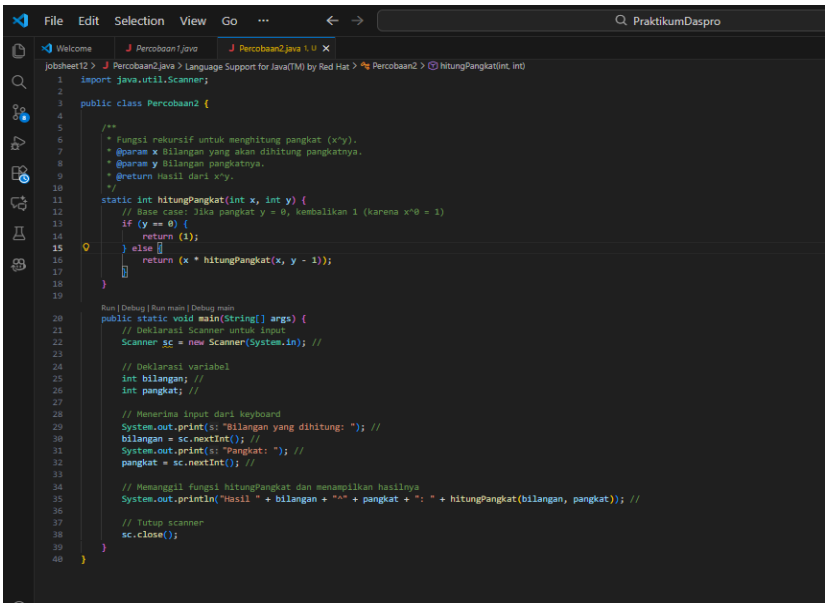
- Mekanisme: Fungsi memanggil dirinya sendiri berulang kali.

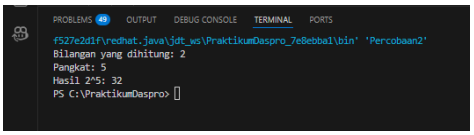
- Alur: Terdapat fase Ekspansi (pemanggilan fungsi ditumpuk ke bawah hingga mencapai *base case*) dan fase Substitusi (nilai dikembalikan ke atas tumpukan untuk perhitungan akhir).
- Efek: Menggunakan lebih banyak memori karena setiap panggilan fungsi baru disimpan di *call stack*.

Fungsi Iteratif (Menggunakan Loop):

- Mekanisme: Menggunakan perulangan (for loop).
- Alur: Perhitungan berjalan secara linier dan berurutan. Variabel hasil (faktor) diperbarui dalam *loop* yang sama dari awal hingga akhir.
- Efek: Lebih efisien dalam penggunaan memori karena semua operasi terjadi dalam satu *frame* tanpa menumpuk panggilan fungsi.

2. Percobaan 2





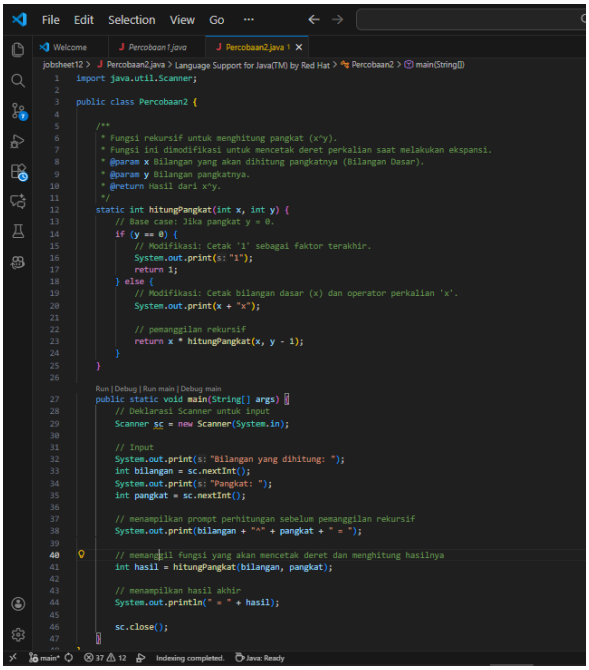
Commit Github Percobaan 2 :

Pertanyaan :

1. Pada Percobaan2, terdapat pemanggilan fungsi rekursif hitungPangkat(bilangan, pangkat) pada fungsi main, kemudian dilakukan pemanggilan fungsi hitungPangkat() secara berulangkali. Jelaskan sampai kapan proses pemanggilan fungsi tersebut akan dijalankan!

2. Tambahkan kode program untuk mencetak deret perhitungan pangkatnya. Contoh : hitungPangkat(2,5) dicetak 2x2x2x2x2x1 = 32

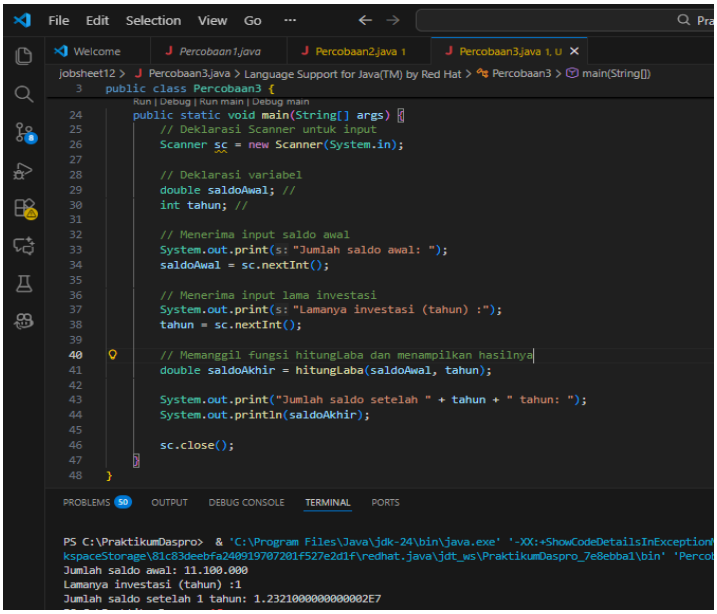
Jawab :

1. Proses pemanggilan fungsi rekursif hitungPangkat(bilangan, pangkat) akan terus dijalankan sampai base case tercapai. Base case untuk perhitungan pangkat adalah ketika nilai pangkat (y) mencapai 0. kondisi berhenti**:** Saat y == 0, fungsi akan mengembalikan nilai 1 (karena bilangan apapun yang dipangkatkan 0 hasilnya adalah 1, x^0 = -1). Ini menghentikan pemanggilan rekursif dan memulai proses pengembalian nilai (substitusi). Jadi, proses akan berhenti ketika parameter pangkat yang dikirimkan ke fungsi hitungPangkat() bernilai 0.
2. 

```
import java.util.Scanner;

public class Percobaan2 {

    /**
     * Fungsi rekursif untuk menghitung pangkat (x^y).
     * Fungsi ini dimodifikasi untuk mencetak deret perkalian saat melakukan ekspansi.
     * @param x Bilangan yang akan dihitung pangkatnya (Bilangan Dasar).
     * @param y Bilangan pangkatnya.
     * @return Hasil dari x^y.
     */
    static int hitungPangkat(int x, int y) {
        // Base case: Jika pangkat y = 0.
        if (y == 0) {
            // Modifikasi: Cetak '1' sebagai faktor terakhir.
            System.out.print("1");
            return 1;
        } else {
            // Modifikasi: Cetak bilangan dasar (x) dan operator perkalian 'x'.
            System.out.print(x + "x");

            // pemanggilan rekursif
            return x * hitungPangkat(x, y - 1);
        }
    }

    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        // Deklarasi Scanner untuk input
        Scanner sc = new Scanner(System.in);

        // Input
        System.out.print("Bilangan yang dihitung: ");
        int bilangan = sc.nextInt();
        System.out.print("Pangkat: ");
        int pangkat = sc.nextInt();

        // menampilkan prompt perhitungan sebelum pemanggilan rekursif
        System.out.print(bilangan + "^" + pangkat + " = ");

        // memanggil fungsi yang akan mencetak deret dan menghitung hasilnya
        int hasil = hitungPangkat(bilangan, pangkat);

        // menampilkan hasil akhir
        System.out.println(" = " + hasil);

        sc.close();
    }
}
```

```
PS C:\PraktikumDaspro>  c:; cd 'c:\PraktikumDaspro'; & 'C:
orage\81c83deebfa240919707201f527e2d1f\redhat.java\jdt_ws\
Bilangan yang dihitung: 2
Pangkat: 4
2^4 = 2x2x2x2x1 = 16
PS C:\PraktikumDaspro>
```

3. Percobaan 3



Commit Github Percobaan 3 :

Pertanyaan :

1. Pada Percobaan3, sebutkan blok kode program manakah yang merupakan “base case” dan “recursion call”!
2. Jabarkan trace fase ekspansi dan fase subtitusi algoritma perhitungan laba di atas jika diberikan nilai hitungLaba(100000,3).

Jawab :

1. Base case :

```
if (tahun <= 0) {
    return (saldo);
} else {
```

Recursion call :

```
    return (1.11 * hitungLaba(saldo, tahun - 1));
}
```

2. **Fase Ekspansi (Panggilan)**

Pada fase ini, fungsi memanggil dirinya sendiri secara berulang, menunda perhitungan:

- $\text{hitungLaba}(100000, 3) \rightarrow 1.11 \times \text{hitungLaba}(100000, 2)$
- $\text{hitungLaba}(100000, 2) \rightarrow 1.11 \times \text{hitungLaba}(100000, 1)$
- $\text{hitungLaba}(100000, 1) \rightarrow 1.11 \times \text{hitungLaba}(100000, 0)$
- $\text{hitungLaba}(100000, 0) \rightarrow \text{return } 100000.0$ (Base Case)

Fase Substitusi (Perhitungan)

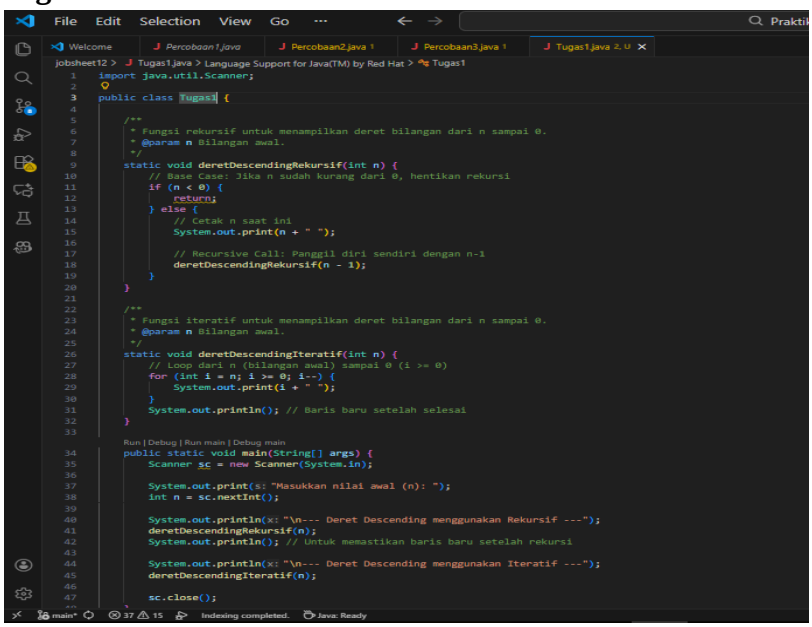
Nilai dikembalikan ke atas tumpukan, dan perhitungan laba diselesaikan:

1. **Tahun 1:** $1.11 \times 100000.0 = 111000.0$
2. **Tahun 2:** $1.11 \times 111000.0 = 123210.0$
3. **Tahun 3:** $1.11 \times 123210.0 = 136763.1$

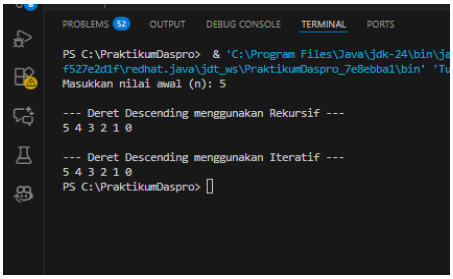
Saldo Akhir: $136,763.10$

4.Tugas

1. Tugas 1



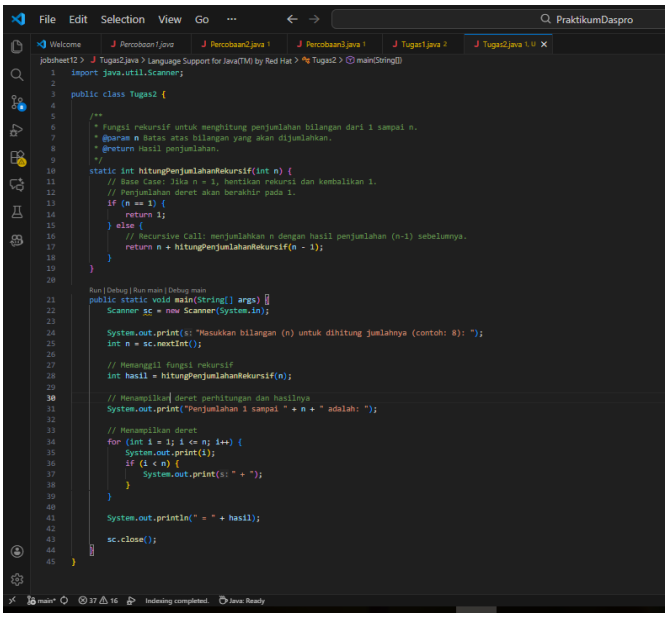
```java
import java.util.Scanner;

public class Tugas1 {

    /**
     * Fungsi rekursif untuk menampilkan deret bilangan dari n sampai 0.
     * @param n Bilangan awal.
     */
    static void deretDescendingRekursif(int n) {
        // Base Case: Jika n sudah kurang dari 0, hentikan rekursi
        if (n < 0) {
            return;
        } else {
            // Cetak n saat ini
            System.out.print(n + " ");

            // Recursive Call: Panggil diri sendiri dengan n-1
            deretDescendingRekursif(n - 1);
        }
    }

    /**
     * Fungsi iteratif untuk menampilkan deret bilangan dari n sampai 0.
     * @param n Bilangan awal.
     */
    static void deretDescendingIteratif(int n) {
        // Loop dari n (bilangan awal) sampai 0 (i >= 0)
        for (int i = n; i >= 0; i--) {
            System.out.print(i + " ");
        }
        System.out.println(); // Baris baru setelah selesai
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Masukkan nilai awal (n): ");
        int n = sc.nextInt();

        System.out.println("\n--- Deret Descending menggunakan Rekursif ---");
        deretDescendingRekursif(n);
        System.out.println(); // Untuk memastikan baris baru setelah rekursi

        System.out.println("\n--- Deret Descending menggunakan Iteratif ---");
        deretDescendingIteratif(n);

        sc.close();
    }
}
```



```
PS C:\PraktikumDaspro> & 'C:\Program Files\Java\jdk-24\bin\ja
f527e2d1f\redhat.java\jdt_ws\PraktikumDaspro_7e8ebba1\bin' 'Tu
Masukkan nilai awal (n): 5

--- Deret Descending menggunakan Rekursif ---
5 4 3 2 1 0

--- Deret Descending menggunakan Iteratif ---
5 4 3 2 1 0
PS C:\PraktikumDaspro>
```

Commit Github Tugas 1 :

2. Tugas 2

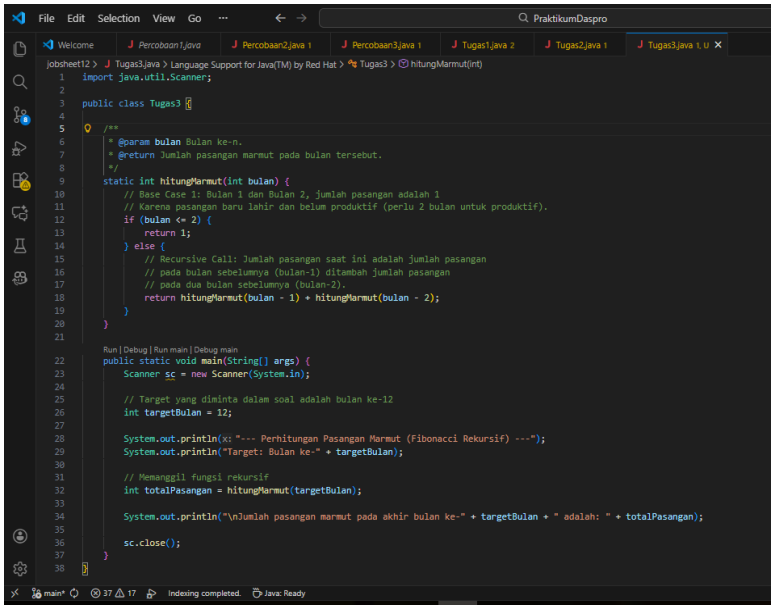


```java
import java.util.Scanner;

public class Tugas2 {

    /**
     * Fungsi rekursif untuk menghitung penjumlahan bilangan dari 1 sampai n.
     * @param n Batas atas bilangan yang akan dijumlahkan.
     * @return Hasil penjumlahan.
     */
    static int hitungPenjumlahanRekursif(int n) {
        // Base Case: Jika n = 1, hentikan rekursi dan kembalikan 1.
        // Penjumlahan deret akan berakhir pada 1.
        if (n == 1) {
            return 1;
        } else {
            // Recursive Call: menjumlahkan n dengan hasil penjumlahan (n-1) sebelumnya.
            return n + hitungPenjumlahanRekursif(n - 1);
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Masukkan bilangan (n) untuk dihitung jumlahnya (contoh: 8): ");
        int n = sc.nextInt();

        // Memanggil fungsi rekursif
        int hasil = hitungPenjumlahanRekursif(n);

        // Menampilkan deret perhitungan dan hasilnya
        System.out.print("Penjumlahan 1 sampai " + n + " adalah: ");

        // Menampilkan deret
        for (int i = 1; i <= n; i++) {
            System.out.print(i);
            if (i < n) {
                System.out.print(" + ");
            }
        }

        System.out.println(" = " + hasil);

        sc.close();
    }
}
```

```
PS C:\PraktikumDaspro> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-XX:+S
f527e2d1f\redhat.java\jdt_ws\PraktikumDaspro_7e8ebba1\bin' 'Tugas2'
Masukkan bilangan (n) untuk dihitung jumlahnya (contoh: 8): 8
Penjumlahan 1 sampai 8 adalah: 1 + 2 + 3 + 4 + 4 + 5 + 6 + 7 + 8 = 36
PS C:\PraktikumDaspro>
```

Commit Github Tugas 2 :

3. Tugas 3



```java
import java.util.Scanner;

public class Tugas3 {

    /**
     * @param bulan Bulan ke-n.
     * @return Jumlah pasangan marmut pada bulan tersebut.
     */
    static int hitungMarmut(int bulan) {
        // Base Case 1: Bulan 1 dan Bulan 2, jumlah pasangan adalah 1
        // Karena pasangan baru lahir dan belum produktif (perlu 2 bulan untuk produktif).
        if (bulan <= 2) {
            return 1;
        } else {
            // Recursive Call: Jumlah pasangan saat ini adalah jumlah pasangan
            // pada bulan sebelumnya (bulan-1) ditambah jumlah pasangan
            // pada dua bulan sebelumnya (bulan-2).
            return hitungMarmut(bulan - 1) + hitungMarmut(bulan - 2);
        }
    }

    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Target yang diminta dalam soal adalah bulan ke-12
        int targetBulan = 12;

        System.out.println(x: "--- Perhitungan Pasangan Marmut (Fibonacci Rekursif) ---");
        System.out.println("Target: Bulan ke-" + targetBulan);

        // Memanggil fungsi rekursif
        int totalPasangan = hitungMarmut(targetBulan);

        System.out.println("\nJumlah pasangan marmut pada akhir bulan ke-" + targetBulan + " adalah: " + totalPasangan);

        sc.close();
    }
}
```

```
PS C:\PraktikumDaspro> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-X
f527e2d1f\redhat.java\jdt_ws\PraktikumDaspro_7e8ebba1\bin' 'Tugas3'
--- Perhitungan Pasangan Marmut (Fibonacci Rekursif) ---
Target: Bulan ke-12

Jumlah pasangan marmut pada akhir bulan ke-12 adalah: 144
PS C:\PraktikumDaspro>
```

Commit Github Tugas 3 :