

Jobsheet 12 Fungsi Rekursif



Nama : Salsabila Widyadhana

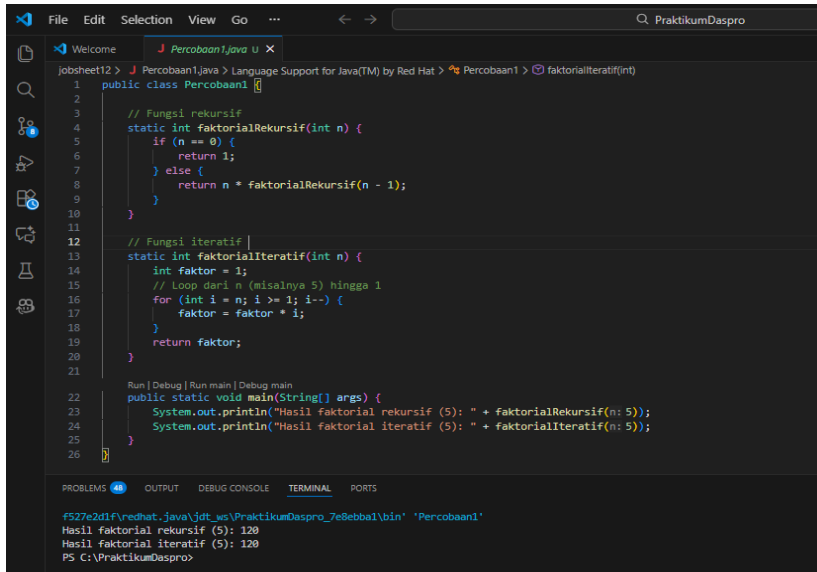
NIM : 254107020200

Kelas : TI-1H

Mata Kuliah : Praktikum Daspro

Pertemuan Ke- : 15

1. Percobaan 1



```
1 public class Percobaan1 {
2
3     // Fungsi rekursif
4     static int faktorialRekursif(int n) {
5         if (n == 0) {
6             return 1;
7         } else {
8             return n * faktorialRekursif(n - 1);
9         }
10    }
11
12    // Fungsi iteratif
13    static int faktorialIteratif(int n) {
14        int faktor = 1;
15        // Loop dari n (misalnya 5) hingga 1
16        for (int i = n; i >= 1; i--) {
17            faktor = faktor * i;
18        }
19        return faktor;
20    }
21
22    public static void main(String[] args) {
23        System.out.println("Hasil faktorial rekursif (5): " + faktorialRekursif(5));
24        System.out.println("Hasil faktorial iteratif (5): " + faktorialIteratif(5));
25    }
26 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

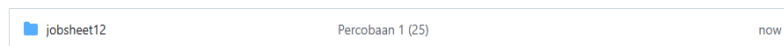
f527e2df:redhat.java:jdt_ws\PraktikumDaspro_7e8ebba1\bin 'Percobaan1'

Hasil faktorial rekursif (5): 120

Hasil faktorial iteratif (5): 120

PS C:\PraktikumDaspro>

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):

- o 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

```

1  import java.util.Scanner;
2
3  public class Percobaan2 {
4
5      /**
6       * Fungsi rekursif untuk menghitung pangkat (x^y).
7       * @param x Bilangan yang akan dihitung pangkatnya.
8       * @param y Bilangan pangkatnya.
9       * @return Hasil dari x^y.
10      */
11
12      static int hitungPangkat(int x, int y) {
13          // Base case: Jika pangkat y = 0, kembalikan 1 (karena x^0 = 1)
14          if (y == 0) {
15              return 1;
16          } else {
17              return x * hitungPangkat(x, y - 1);
18          }
19      }
20
21      public static void main(String[] args) {
22          // Deklarasi Scanner untuk input
23          Scanner sc = new Scanner(System.in); //
24
25          // Deklarasi variabel
26          int bilangan; //
27          int pangkat; //
28
29          // Menerima input dari keyboard
30          System.out.print("Bilangan yang dihitung: "); //
31          bilangan = sc.nextInt(); //
32          System.out.print("Pangkat: "); //
33          pangkat = sc.nextInt(); //
34
35          // Memanggil fungsi hitungPangkat dan menampilkan hasilnya
36          System.out.println("Hasil " + bilangan + "^" + pangkat + " = " + hitungPangkat(bilangan, pangkat)); //
37
38          // Tutup scanner
39          sc.close();
40      }
41  }

```

```

f527e2d1f\redhat.java\jdk_ws\PraktikumDaspro_7eRebbal\bin' 'Percobaan2'
Bilangan yang dihitung: 2
Pangkat: 5
Hasil 2^5: 32
PS C:\PraktikumDaspro>

```

Commit Github Percobaan 2 :

jobsheet12	Percobaan 2 (25)	now
------------	------------------	-----

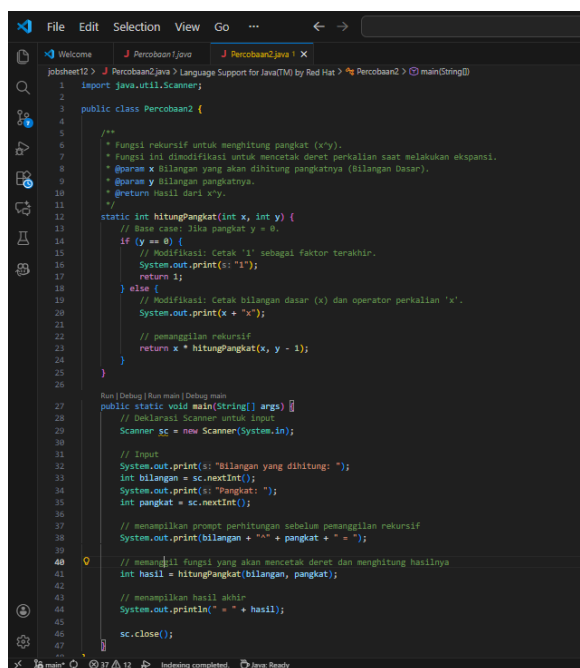
Pertanyaan :

1. Pada Percobaan2, terdapat pemanggilan fungsi rekursif hitungPangkat(bilangan, pangkat) pada fungsi main, kemudian dilakukan pemanggilan fungsi hitungPangkat() secara berulang kali. Jelaskan sampai kapan proses pemanggilan fungsi tersebut akan dijalankan!
2. Tambahkan kode program untuk mencetak deret perhitungan pangkatnya. Contoh : hitungPangkat(2,5) dicetak $2 \times 2 \times 2 \times 2 \times 2 \times 1 = 32$

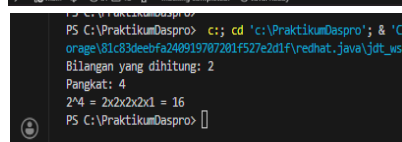
Jawab :

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.

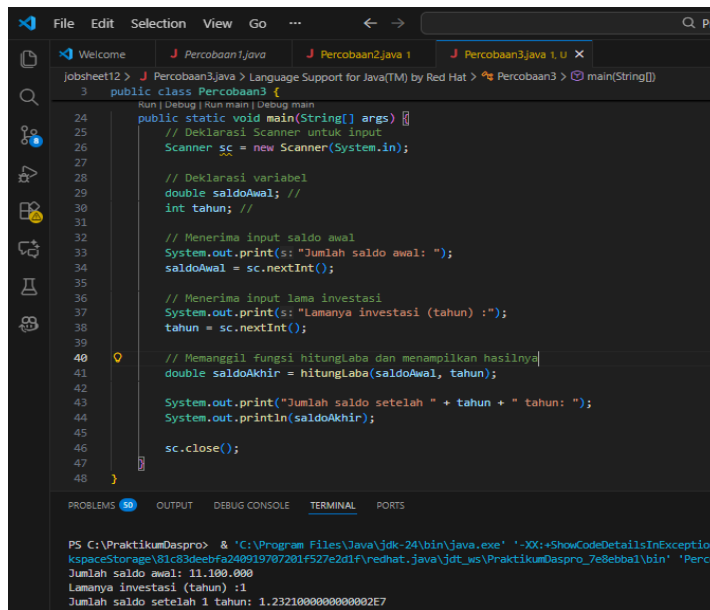


```
File Edit Selection View Go ...
Welcome J Percobaan1.java J Percobaan2.java X
jobsheet12 > J Percobaan2.java > Language Support for Java(TM) by Red Hat > J Percobaan2 > main(String[])
1 import java.util.Scanner;
2
3 public class Percobaan2 {
4
5
6     /**
7      * Fungsi rekursif untuk menghitung pangkat (x^y).
8      * Fungsi ini dimodifikasi untuk mencetak deret perkalian saat melakukan ekspansi.
9      * @param x bilangan yang akan dihitung pangkatnya (bilangan dasar).
10     * @param y Bilangan pangkatnya.
11     * @return Hasil dari x^y.
12     */
13     static int hitungPangkat(int x, int y) {
14         // Base case: jika pangkat y = 0.
15         if (y == 0) {
16             // Modifikasi: Cetak '1' sebagai faktor terakhir.
17             System.out.print("1");
18             return 1;
19         } else {
20             // Modifikasi: Cetak bilangan dasar (x) dan operator perkalian 'x'.
21             System.out.print(x + "x");
22             // pemanggilan rekursif
23             return x * hitungPangkat(x, y - 1);
24         }
25     }
26
27     public static void main(String[] args) {
28         // Deklarasi Scanner untuk input
29         Scanner sc = new Scanner(System.in);
30
31         // Input
32         System.out.print("Bilangan yang dihitung: ");
33         int bilangan = sc.nextInt();
34         System.out.print("Pangkat: ");
35         int pangkat = sc.nextInt();
36
37         // menampilkan prompt perhitungan sebelum pemanggilan rekursif
38         System.out.print(bilangan + "x" + pangkat + " = ");
39
40         // memanggil fungsi yang akan mencetak deret dan menghitung hasilnya
41         int hasil = hitungPangkat(bilangan, pangkat);
42
43         // menampilkan hasil akhir
44         System.out.println(" = " + hasil);
45
46         sc.close();
47     }
48 }
```



```
PS C:\PraktikumDaspro> cd 'C:\PraktikumDaspro'; & 'C:\Program Files\Java\jdk-8.0.60\bin\java.exe' -cp 'C:\PraktikumDaspro\classes;C:\Program Files\Java\jdk-8.0.60\lib\rt.jar;C:\Program Files\Java\jdk-8.0.60\lib\jrt.jar' org.praktikumdaspro.Percobaan2
Bilangan yang dihitung: 2
Pangkat: 4
2^4 = 2x2x2x2x1 = 16
PS C:\PraktikumDaspro>
```

3. Percobaan 3



```
public class Percobaan3 {  
    public static void main(String[] args) {  
        // Deklarasi Scanner untuk input  
        Scanner sc = new Scanner(System.in);  
  
        // Deklarasi variabel  
        double saldoAwal; //  
        int tahun; //  
  
        // Menerima input saldo awal  
        System.out.print(s: "Jumlah saldo awal: ");  
        saldoAwal = sc.nextInt();  
  
        // Menerima input lama investasi  
        System.out.print(s: "Lamanya investasi (tahun) :");  
        tahun = sc.nextInt();  
  
        // Memanggil fungsi hitungLaba dan menampilkan hasilnya  
        double saldoAkhir = hitungLaba(saldoAwal, tahun);  
  
        System.out.print("Jumlah saldo setelah " + tahun + " tahun: ");  
        System.out.println(saldoAkhir);  
  
        sc.close();  
    }  
}
```

PS C:\PraktikumDaspro> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-XG:ShowCodeDetailsInException' kspaceStorage\81c83deebfa240919707201f527e2d1f\redhat.java\jdt_ws\PraktikumDaspro_7e8ebba1\bin\Perco
Jumlah saldo awal: 11.100.000
Lamanya investasi (tahun) :1
Jumlah saldo setelah 1 tahun: 1.2321000000000002E7

Commit Github Percobaan 3 :

jobsheet12	Percobaan 3 (25)	now
------------	------------------	-----

Pertanyaan :

1. Pada Percobaan3, sebutkan blok kode program manakah yang merupakan “base case” dan “recursion call”!
2. Jabarkan trace fase ekspansi dan fase substitusi 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));  
}
```

Fase Ekspansi (Panggilan)

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

- `hitungLaba(100000, 3) → 1.11 × hitungLaba(100000, 2)`
- `hitungLaba(100000, 2) → 1.11 × hitungLaba(100000, 1)`
- `hitungLaba(100000, 1) → 1.11 × hitungLaba(100000, 0)`
- `hitungLaba(100000, 0) → return 100000.0 (Base Case)`

Fase Substitusi (Perhitungan)

Nilai dikembalikan ke atas tumpukan, dan perhitungan laba diselesaikan:

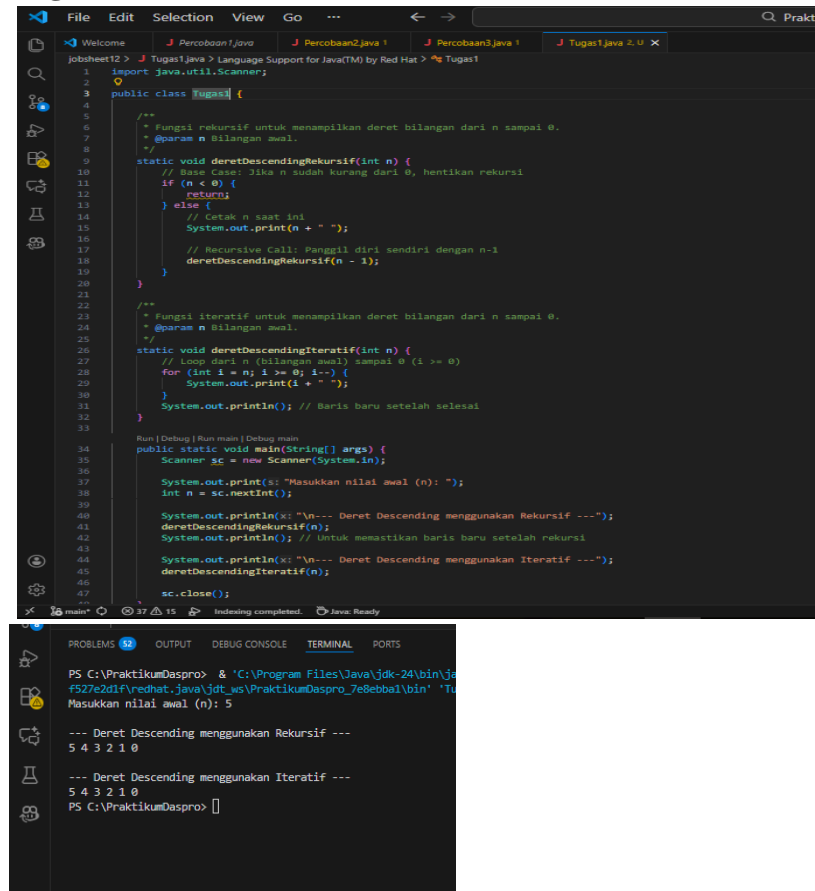
1. Tahun 1: $1.11 \times 100000.0 = 111000.0$
2. Tahun 2: $1.11 \times 111000.0 = 123210.0$
3. Tahun 3: $1.11 \times 123210.0 = 136763.1$

- 2.

Saldo Akhir: 136,763.10

4. Tugas

1. Tugas 1



```
1 import java.util.Scanner;
2
3 public class Tugas1 {
4
5     /**
6      * Fungsi rekursif untuk menampilkan deret bilangan dari n sampai 0.
7      * @param n Bilangan awal.
8      */
9     static void deretDescendingRekursif(int n) {
10         // Base Case: Jika n sudah kurang dari 0, hentikan rekursi
11         if (n < 0) {
12             return;
13         }
14         // Cetak n saat ini
15         System.out.print(n + " ");
16         // Recursive Call: Panggil diri sendiri dengan n-1
17         deretDescendingRekursif(n - 1);
18     }
19
20     /**
21      * Fungsi iteratif untuk menampilkan deret bilangan dari n sampai 0.
22      * @param n Bilangan awal.
23      */
24     static void deretDescendingIteratif(int n) {
25         // Loop dari n (bilangan awal) sampai 0 (i >= 0)
26         for (int i = n; i >= 0; i--) {
27             System.out.print(i + " ");
28         }
29         System.out.println(); // Baris baru setelah selesai
30     }
31
32     public static void main(String[] args) {
33         Scanner sc = new Scanner(System.in);
34
35         System.out.print("Masukkan nilai awal (n): ");
36         int n = sc.nextInt();
37
38         System.out.println("\n--- Deret Descending menggunakan Rekursif ---");
39         deretDescendingRekursif(n);
40         System.out.println(); // Untuk memastikan baris baru setelah rekursi
41
42         System.out.println("\n--- Deret Descending menggunakan Iteratif ---");
43         deretDescendingIteratif(n);
44
45         sc.close();
46     }
47 }
```

```
PS C:\PraktikumDaspro> & 'C:\Program Files\Java\jdk-24\bin\java.exe' -cp f527e2d1f\redhat.java\jdt_ws\PraktikumDaspro_7e8ebbal\bin 'Tu
Masukkan nilai awal (n): 5

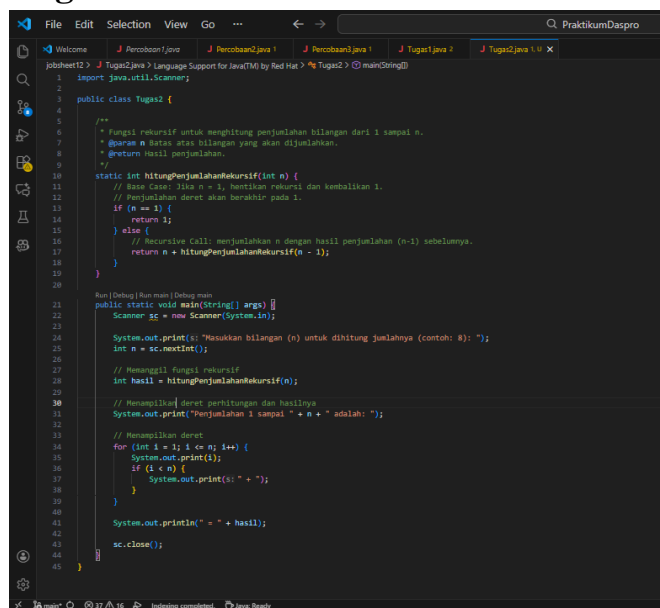
--- Deret Descending menggunakan Rekursif ---
5 4 3 2 1 0

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

Commit Github Tugas 1 :

jobsheet12	Tugas 1 (25)	now
------------	--------------	-----

2. Tugas 2



```
1 import java.util.Scanner;
2
3 public class Tugas2 {
4
5     /**
6      * Fungsi rekursif untuk menghitung penjumlahan bilangan dari 1 sampai n.
7      * @param n Bilangan awal yang akan dijumlahkan.
8      * @return Hasil penjumlahan.
9      */
10     static int hitungPenjumlahanRekursif(int n) {
11         // Base Case: Jika n = 1, hentikan rekursi dan kembalikan 1.
12         // Penjumlahan deret akan berakhir pada 1.
13         if (n == 1) {
14             return 1;
15         }
16         // Recursive Call: Menjumlahkan n dengan hasil penjumlahan (n-1) sebelumnya.
17         return n + hitungPenjumlahanRekursif(n - 1);
18     }
19
20     public static void main(String[] args) {
21         Scanner sc = new Scanner(System.in);
22
23         System.out.print("Masukkan bilangan (n) untuk dihitung jumlahnya (contoh: 8): ");
24         int n = sc.nextInt();
25
26         // Memanggil fungsi rekursif
27         int hasil = hitungPenjumlahanRekursif(n);
28
29         // Menampilkan deret perhitungan dan hasilnya
30         System.out.print("Penjumlahan 1 sampai " + n + " adalah: ");
31
32         // Menampilkan deret
33         for (int i = 1; i <= n; i++) {
34             System.out.print(i);
35             if (i < n) {
36                 System.out.print(" + ");
37             }
38         }
39
40         System.out.println(" = " + hasil);
41
42         sc.close();
43     }
44 }
```

```
PS C:\PraktikumDaspro> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-XX:+S
f527e2d1fredhat.java\jdt_ws\PraktikumDaspro_7e8ebbal\bin' 'Tugas2'
Masukkan bilangan (n) untuk dihitung jumlahnya (contoh: 8): 8
Penjumlahan 1 sampai 8 adalah: 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = 36
PS C:\PraktikumDaspro> []
```

Commit Github Tugas 2 :

jobsheet12	Tugas 2 (25)	now
------------	--------------	-----

3. Tugas 3

```
File Edit Selection View Go ...
Welcome Percobaan1.java Percobaan2.java 1 Percobaan3.java 1 Tugas1.java 2 Tugas2.java 1 Tugas3.java 1 U X
jobsheet12 > J Tugas3.java > Language Support for Java(TM) by Red Hat > % Tugas3 > @ hitungMarmut(int)
1 import java.util.Scanner;
2
3 public class Tugas3 {
4
5     /**
6      * @param bulan Bulan ke-n.
7      * @return Jumlah pasangan marmut pada bulan tersebut.
8      */
9     static int hitungMarmut(int bulan) {
10         // Base Case 1: Bulan 1 dan Bulan 2, Jumlah pasangan adalah 1
11         // Karena pasangan baru lahir dan belum produktif (perlu 2 bulan untuk produktif).
12         if (bulan <= 2) {
13             return 1;
14         } else {
15             // Recursive Call: Jumlah pasangan saat ini adalah jumlah pasangan
16             // pada bulan sebelumnya (bulan-1) ditambah jumlah pasangan
17             // pada dua bulan sebelumnya (bulan-2).
18             return hitungMarmut(bulan - 1) + hitungMarmut(bulan - 2);
19         }
20     }
21
22     Run [Debug] [Run main] [Debug main]
23     public static void main(String[] args) {
24         Scanner sc = new Scanner(System.in);
25
26         // Target yang diminta dalam soal adalah bulan ke-12
27         int targetBulan = 12;
28
29         System.out.println(x: "--- Perhitungan Pasangan Marmut (Fibonacci Rekursif) ---");
30         System.out.println("Target: Bulan ke-" + targetBulan);
31
32         // Memanggil fungsi rekursif
33         int totalPasangan = hitungMarmut(targetBulan);
34
35         System.out.println("\nJumlah pasangan marmut pada akhir bulan ke-" + targetBulan + " adalah: " + totalPasangan);
36
37         sc.close();
38     }
39
40
41 PS C:\PraktikumDaspro> & 'C:\Program Files\Java\jdk-24\bin\java.exe' '-X
f527e2d1fredhat.java\jdt_ws\PraktikumDaspro_7e8ebbal\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 :

jobsheet12	Tugas 3 (25)	now
------------	--------------	-----