TUGAS MATA KULIAH

PRAKTIKUM DASAR PEMROGRAMAN

JOBSHEET 12

FUNGSI REKURSIF

OLEH:

MARTA PRAMA DANISWARA

244107020205

PROGRAM STUDI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

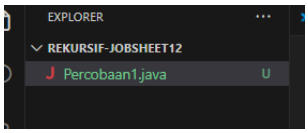
POLITEKNIK NEGERI MALANG

MALANG

2024

2.1 PERCOBAAN 1

1. Buat repository baru dengan nama **Rekursif-jobsheet12**. Buat file Java dengan nama **Percobaan1.java.**



2. Buat fungsi static dengan nama **faktorialRekursif()** dengan tipe data kembalian fungsi *int* dan memiliki 1 parameter dengan tipe data *int* berupa bilangan yang akan dihitung nilai faktorialnya.

```
Percobaan1.java > Percobaan1
1   public class Percobaan1 {
2       static int faktorialRekursif(int n) {
3           if (n==0) {
4               return (1);
5           } else {
6               return (n * faktorialRekursif(n - 1));
7           }
```

3. Buat fungsi static dengan nama **faktorialIteratif()** dengan tipe data kembalian fungsi *int* dan memiliki 1 parameter dengan tipe data *int* berupa bilangan yang akan dihitung nilai faktorialnya.

```
10      static int faktorialIteratif(int n) {
11          int faktor = 1;
12          for (int i = n; i >= 1; i--) {
13              faktor = faktor * i;
14          }
15          return faktor;
16      }
17
```

4. Buat fungsi main dan lakukan pemanggilan terhadap kedua fungsi yang telah dibuat sebelumnya, dan tampilkan hasil yang didapatkan.

```
        Run | Debug
18      public static void main(String[] args) {
19          System.out.println(faktorialRekursif(n:5));
20          System.out.println(faktorialIteratif(n:5));
21      }
22  }
```

5. Run program.

```
120
120
```

JAWAB

1. Fungsi rekursif adalah fungsi yang memanggil dirinya sendiri secara langsung atau tidak langsung untuk menyelesaikan suatu masalah. Setiap pemanggilan rekursif

memiliki *base case* (kasus dasar) yang menentukan kapan rekursi berhenti, serta *recursive case* yang memecah masalah menjadi sub-masalah yang lebih kecil.

2. Contoh kasus penggunaan fungsi rekursif:
   - Faktorial
   - Pencarian dalam struktur data pohon (*tree traversal*)
   - Pembagian dan penaklukan (divide and conquer), seperti dalam algoritma merge sort atau quick sort.
   - Perhitungan deret Fibonacci.
   - Menghitung pangkat
3. Ya, hasil yang diberikan oleh faktorialRekursif(5) dan faktorialIteratif(5) adalah **sama**, yaitu **120**. Kedua fungsi menghitung faktorial dari angka 5 dengan cara yang berbeda tetapi memberikan hasil akhir yang sama.

   Perbedaan alur program:
   - **Fungsi Rekursif:**
     - Fungsi memanggil dirinya sendiri sampai mencapai *base case*.
     - Setiap pemanggilan fungsi menyimpan konteks (variabel lokal dan alamat fungsi) di *call stack.*
     - Setelah mencapai *base case*, nilai faktorial dihitung dengan melakukan *unwinding* (kembali ke pemanggilan sebelumnya) dan mengalikan hasil dari setiap tahap.
   - **Fungsi Iteratif:**
     - Menggunakan perulangan `for` untuk menghitung faktorial.
     - Nilai faktorial dihitung dalam satu langkah dengan mengalikan nilai `faktor` pada setiap iterasi.
     - Tidak ada pemanggilan fungsi berulang, sehingga lebih efisien dalam penggunaan memori.

2.2 PERCOBAAN 2

1. Pada project yang sama, buat file Java dengan nama **Percobaan2.java**.

2. Buat fungsi static dengan nama hitungPangkat(), dengan tipe data kembalian fungsi *int* dan memiliki 2 parameter dengan tipe data *int* berupa bilangan yang akan dihitung pangkatnya dan bilangan pangkatnya.

```java
Percobaan2.java > Percobaan2
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));
        }
    }
```

3. Buat fungsi main dan deklarasikan Scanner sc. Kemudian, buat 2 variabel bertipe *int* dengan nama **bilangan** dan **pangkat**.

```java
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int bilangan, pangkat;
```

4. Tambahkan kode program seperti dibawah.

```java
        System.out.print(s:"Bilangan yang dihitung: ");
        bilangan = sc.nextInt();
        System.out.print(s:"Pangkat: ");
        pangkat = sc.nextInt();
```

5. Lakukan pemanggilan fungsi **hitungPangkat** yang telah dibuat sebelumnya dengan mengirimkan dua nilai parameter.

```java
        System.out.println(hitungPangkat(bilangan, pangkat));
    }
}
```

6. Run program.

```
Bilangan yang dihitung: 7
Pangkat: 3
343
```

JAWAB

1. Proses pemanggilan rekursif akan dijalankan **berulang kali** sampai **nilai y mencapai 0**, yang merupakan *base case* dan titik di mana rekursi berhenti. Jumlah pemanggilan fungsi rekursif adalah sebanyak nilai y, dan setiap pemanggilan tersebut menurunkan nilai y hingga mencapai 0.
2. Tambahkan kode program untuk mencetak deret perhitungan pangkatnya.

```java
import java.util.Scanner;
public class Percobaan2 {
    static String simbol = " ";
    static int hitungPangkat(int x, int y) {
        if (y==0) {
            simbol += "1";
            return (1);
        } else {
            simbol += x + "x";
            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(s:"Bilangan yang dihitung: ");
        bilangan = sc.nextInt();
        System.out.print(s:"Pangkat: ");
        pangkat = sc.nextInt();

        System.out.print(s:"Perhitungan: ");
        int hasil = hitungPangkat(bilangan, pangkat);
        System.out.println(simbol + " = " + hasil);
    }
}
```

Output:

```
Bilangan yang dihitung: 5
Pangkat: 5
Perhitungan: 5x5x5x5x5x1 = 13125
```

2.3 PERCOBAAN 3

1. Pada project yang sama, buat file Java dengan nama **Percobaan3.java**.

2. Buat fungsi static dengan nama **hitungLaba(),** dengan tipe data kembalian fungsi *double* dan memiliki 2 parameter dengan tipe data *int* berupa saldo investor dan lamanya investasi.

```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));
        }
    }
```

3. Buat fungsi main dan deklarasikan Scanner sc. Kemudian, buat variabel bertipe *double* dengan nama **saldoAwal** dan bertipe *int* dengan nama **tahun**.

```java
    Run | Debug
    public static void main(String[] args) {
        Scanner sc =  new Scanner(System.in);
        double saldoAwal;
        int tahun;
```

4. Tambahkan kode program seperti dibawah.

```
16          System.out.print(s:"Jumlah saldo awal: ");
17          saldoAwal = sc.nextDouble();
18          System.out.print(s:"Lamanya investasi (tahun): ");
19          tahun = sc.nextInt();
20
```

5. Lakukan pemanggilan fungsi **hitungLaba** yang telah dibuat sebelumnya dengan mengirimkan dua nilai parameter.

```
21          System.out.print("Jumlah saldo setelah " + tahun + " tahun: ");
22          System.out.println(hitungLaba(saldoAwal, tahun));
23      }
24
25  }
```

6. Run program.

```
Jumlah saldo awal: 12000000
Lamanya investasi (tahun): 2
Jumlah saldo setelah 2 tahun: 1.4785200000000004E7
```

JAWAB

1. Base case:
```
if (tahun == 0) {
    return (saldo);
```
Recursion call:
```
return (1.11 * hitungLaba(saldo, tahun - 1));
```
2. Analisis Rekursi dan Trace untuk **hitungLaba(100000, 3)**. Fungsi **hitungLaba** menggunakan rekursi untuk menghitung saldo akhir dengan kenaikan 11% setiap tahunnya. Dijabarkan menjadi 2 fase rekursi:
    a. **Fase Ekspansi (Memperluas Panggilan Rekursi)**

    Fungsi terus memanggil dirinya sendiri (mengurangi nilai tahun secara bertahap) sampai mencapai *base case* (tahun == 0).

    1) **Pemanggilan Awal:**

    hitungLaba(100000,3)

    Belum bisa dihitung langsung, karena tahun != 0, maka dilakukan ekspansi:

    1.11×hitungLaba(100000,2)

    2) **Panggilan Kedua:**

    1.11×hitungLaba(100000,2)

    Belum bisa dihitung, dilakukan ekspansi lagi:

    1.11×(1.11×hitungLaba(100000,1))

3) **Panggilan Ketiga:**

   1.11×(1.11×hitungLaba(100000,1))

   Dilakukan ekspansi terakhir:

   1.11×(1.11×(1.11×hitungLaba(100000,0)))

4) **Base Case (Akhir Ekspansi):**

   hitungLaba(100000,0)=100000

b. Fase Substitusi (Menghitung Hasil dari Ekspansi)

Pada fase ini, nilai dikembalikan dari fungsi terdalam ke fungsi terluar.

1) **Langkah 1 (Substitusi Base Case):**

   hitungLaba(100000,0)=100000

2) **Langkah 2 (Menghitung Tahun ke-1):**

   hitungLaba(100000,1)=1.11×100000=111000

3) **Langkah 3 (Menghitung Tahun ke-2):**

   hitungLaba(100000,2)=1.11×111000=123210

4) **Langkah 4 (Menghitung Tahun ke-3):**

   hitungLaba(100000,3)=1.11×123210=136763.1

Hasil Akhir:

Jumlah saldo setelah 3 tahun=136763.1

TUGAS

1. Buatlah program untuk menampilkan bilangan n sampai 0 dengan menggunakan fungsi rekursif dan fungsi iteratif. (DeretDescendingRekursif).
   - Kode program

```java
import java.util.Scanner;

public class DeretDescending {
    static void deretDescendingRekursif(int n){
        if (n < 0) {
            return;
        } else {
            System.out.print(n + " ");
            deretDescendingRekursif(n - 1);
        }
    }
    static void deretDescendingIteratif(int n){
        for (int i = n; i >= 0; i--) {
            System.out.print(i + " ");
        }
    }
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Masukkkan bilangan n: ");
        int n = sc.nextInt();

        System.out.print("Deret Rekursif: ");
        deretDescendingRekursif(n);
        System.out.println();

        System.out.print("Deret Iteratif: ");
        deretDescendingIteratif(n);
    }
}
```

- Output

```
Masukkkan bilangan n: 9
Deret Rekursif: 9 8 7 6 5 4 3 2 1 0
Deret Iteratif: 9 8 7 6 5 4 3 2 1 0
```

2. Buatlah program yang di dalamnya terdapat fungsi rekursif untuk menghitung penjumlahan bilangan. Misalnya f = 8, maka akan dihasilkan 1+2+3+4+5+6+7+8 = 36 (PenjumlahanRekursif).

- Kode program:

```java
import java.util.Scanner;

public class PenjumlahanRekursif {
    static int hitungPenjumlahan(int n){
        if (n == 1) {
            return (1);
        } else {
            return (n + hitungPenjumlahan(n - 1));
        }
    }
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Masukkan angka: ");
        int bil = sc.nextInt();

        int hasil = hitungPenjumlahan(bil);
        System.out.println("Hasil penjumlahan rekursif angka " + bil + " = " + hasil);
    }
}
```

- Output:

```
Masukkan angka: 5
Hasil penjumlahan rekursif angka 5 = 15
```

3. Sepasang marmut yang baru lahir (jantan dan betina) ditempatkan pada suatu pembiakan. Setelah dua bulan pasangan marmut tersebut melahirkan sepasang marmut kembar (jantan dan betina). Setiap pasangan marmut yang lahir juga akan melahirkan sepasang marmut juga setiap 2 bulan. Berapa pasangan marmut yang ada pada akhir bulan ke-12? Buatlah programnya menggunakan fungsi rekursif! (Fibonacci).

- Kode program:

```java
import java.util.Scanner;

public class Fibonacci {
    static int hitungMarmut(int bulan) {
        if (bulan == 1 || bulan == 2) {
            return (1);
        }
        return (hitungMarmut(bulan - 1) + hitungMarmut(bulan - 2));
    }

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

        System.out.print("Masukkan bulan ke-n: ");
        int bulan = sc.nextInt();

        int hasil = hitungMarmut(bulan);
        System.out.println("Jumlah pasangan marmut pada bulan ke-" + bulan + ": " + hasil);
    }
}
```

- Output:

```
Masukkan bulan ke-n: 11
Jumlah pasangan marmut pada bulan ke-11: 89
```