LAPORAN PRAKTIKUM

PRAKTIKUM DASAR PEMROGRAMAN

JOBSHEET 12

Disusun oleh:

NEVITA TRIYA YULIANA (21)

244107020208

PROGRAM STUDI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

PERCOBAAN 1:

1. Buat project baru bernama Rekursif, dan buat file Java dengan nama Percobaan1
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.

```
public static int faktorialRekursif(int n) {
    if (n == 0) {
        return (1);
    } else {
        return(n * faktorialRekursif(n-1));
    }
}
```

3. Buat lagi 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.

```
public static int faktorialIteratif (int n) {
    int faktor = 1;
    for (int i = n; i < 1; i++) {
        faktor = faktor * i;
    }
    return faktor;
}
```

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

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

5. Jalankan program tersebut. Amati apa yang terjadi!

```
120
1
PS C:\Users\HP\daspro-jobsheet12>
```

6. Jika ditelusuri, pada saat pemanggilan fungsi faktorialRekursif(5),maka proses yang terjadi dapat diilustrasikan sebagai berikut:

PERTANYAAN:

1. Apa yang dimaksud dengan fungsi rekursif?
   ⇨ Fungsi rekursif adalah suatu fungsi yang memanggil dirinya sendiri dalam proses eksekusinya.
2. Bagaimana contoh kasus penggunaan fungsi rekursif ?
   ⇨ Penggunaan fungsi rekursif pada kasus menghitung faktorial n, menghitung angka ke-n dalam deret fibonacci

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!
   - ⇨ Berbeda,
   - ⇨ Fungsi rekursif
     1. Panggil faktorialRekursif(5).
     2. Karena 5 tidak sama dengan 0, maka fungsi memanggil faktorialRekursif(4).
     3. Proses berulang hingga faktorialRekursif(0) dipanggil dan mengembalikan 1.
     4. Hasil dari setiap pemanggilan kemudian dikalikan secara berurutan hingga kembali ke pemanggilan awal.
   - ⇨ Fungsi iterative
     1. Inisialisasi faktor dengan 1.
     2. Melakukan perulangan dari n hingga 1.
     3. Pada setiap iterasi, faktor dikalikan dengan nilai i.
     4. Setelah perulangan selesai, nilai faktor yang dihasilkan adalah nilai faktorial.

PERCOBAAN 2:

1. Pada project Rekursif, dan buat file Java dengan nama Percobaan2
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.

```
public static int hitungPangkat(int x, int y) {
    if (y == 0) {
        return (1);
    }else {
        return (x * hitungPangkat(x, y- 1));
    }
}
```

3. Buatlah fungsi main dan deklarasikan Scanner dengan nama sc
4. Buatlah dua buah variabel bertipa int dengan nama bilangan dan pangkat

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

5. Tambahkan kode berikut ini untuk menerima input dari keyboard

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

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

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

7. Jalankan program tersebut. Amati apa yang terjadi!

```
Bilangan yang dihitung: 4
Pangkat: 3
64
```

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!
   - ⇨ Proses pemanggilan fungsi akan berhenti ketika y mencapai 0. Ini berarti bahwa jumlah total pemanggilan fungsi bergantung pada nilai y yang diberikan. Jika y adalah 5, maka akan ada 6 pemanggilan fungsi (termasuk panggilan awal hitungPangkat(x, y) dan 5 pemanggilan rekursif hingga hitungPangkat(x, 0)).
2. Tambahkan kode program untuk mencetak deret perhitungan pangkatnya. Contoh : hitungPangkat(2,5) dicetak 2x2x2x2x2x1 = 32

```java
import java.util.Scanner;
public class Percobaan2 {
    public static int hitungPangkat(int x, int y) {
        if (y == 0) {
            System.out.print(s:"1");
            return 1 ;
        }else {
            if (y == 1) {
                System.out.print(x + "x");
            } else {
                System.out.print(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();

        int hasil = hitungPangkat(bilangan, pangkat);
        System.out.println("=" + hasil);
    }
}
```

PERCOBAAN 3:

1. Pada project Rekursif, dan buat file Java dengan nama Percobaan3
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. Pada kasus ini dianggap laba yang ditentukan adalah 11% per tahun. Karena perhitungan laba adalah laba * saldo, sehingga untuk menghitung besarnya uang setelah ditambah laba adalah saldo + laba * saldo. Dalam hal ini, besarnya laba adalah 0.11 * saldo, dan saldo dianggap 1 * saldo, sehingga 1 * saldo + 0.11 * saldo dapat diringkas menjadi 1.11 * saldo untuk perhitungan saldo setelah ditambah laba (dalam setahun).

```
public static double hitungLaba(double saldo, int tahun) {
    if (tahun == 0) {
        return (saldo);
    } else {
        return (1.11 * hitungLaba(saldo, tahun - 1));
    }

}
```

3. Buatlah fungsi main dan deklarasikan Scanner dengan nama sc
4. Buatlah sebuah variabel bertipa double dengan nama saldoAwal dan sebuah variabel bertipe int bernama tahun

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

5. Tambahkan kode berikut ini untuk menerima input dari keyboard

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

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

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

7. Jalankan program tersebut. Amati apa yang terjadi!

```
Jumlah saldo awal : 5.000.000
Lamanya investasi (tahun) :1
Jumlah saldo setelah 1 tahun : 5550000.000000001
```

PERTANYAAN:

1. Pada Percobaan3, sebutkan blok kode program manakah yang merupakan “base case” dan “recursion call”!
   ⇨ Base case terdapat pada kondisi *if (tahun == 0)*. Ketika kondisi ini terpenuhi, fungsi *hitungLaba* akan mengembalikan nilai saldo tanpa memanggil dirinya sendiri lagi.
   ⇨ Recursion call terjadi di bagian *return (1.11 * hitungLaba(saldo, tahun - 1));*. Di sini, fungsi *hitungLaba* memanggil dirinya sendiri dengan *tahun - 1* sebagai argumen, sehingga mendekati kondisi base case.
2. Jabarkan trace fase ekspansi dan fase subtitusi algoritma perhitungan laba di atas jika diberikan nilai hitungLaba(100000,3)
   ⇨ Fase Ekspansi
      1. Panggilan pertama: *hitungLaba(100000,3)*
         - Karena *tahun* tidak sama dengan 0, fungsi memanggil dirinya sendiri dengan *tahun - 1: hitungLaba(100000, 2)*
      2. Panggilan kedua: *hitungLaba(100000, 2)*
         - Karena *tahun* tidak sama dengan 0, fungsi memanggil dirinya sendiri lagi dengan *tahun - 1: hitungLaba(100000, 1)*
      3. Panggilan ketiga: *hitungLaba(100000, 1)*
         - Karena *tahun* tidak sama dengan 0, fungsi memanggil dirinya sendiri lagi dengan *tahun - 1: hitungLaba(100000, 0)*
      4. Panggilan keempat (base case): *hitungLaba(100000, 0)*
         - Karena *tahun* sama dengan 0, fungsi mengembalikan *saldo (100000).* Ini adalah kondisi base case.
   ⇨ Fase Substitusi
      1. Base case mengembalikan nilai:
         - *hitungLaba(100000, 0)* mengembalikan 100000.
      2. Substitusi pada panggilan ketiga:
         - *hitungLaba(100000, 1)* menggantikan hasil *1.11 * hitungLaba(100000, 0)* dengan *1.11 * 100000*.
         - Hasilnya: *hitungLaba(100000, 1) = 111000.0*
      3. Substitusi pada panggilan kedua:
         - *hitungLaba(100000, 2)* menggantikan hasil *1.11 * hitungLaba(100000, 1)* dengan *1.11 * 111000.0.*
         - Hasilnya: *hitungLaba(100000, 2) = 123210.0*
      4. Substitusi pada panggilan pertama:
         - *hitungLaba(100000, 3)* menggantikan hasil *1.11 * hitungLaba(100000, 2)* dengan *1.11 * 123210.0.*
         - Hasilnya: *hitungLaba(100000, 3) = 136962.1*

TUGAS:

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

```java
import java.util.Scanner;
public class Tugas1 {
    public static void tampilkanDeretRekursif(int n) {
        if (n < 0) {
            return;
        }
        System.out.print(n + " ");
        tampilkanDeretRekursif(n - 1);
    }
    public static void tampilkanDeretIteratif(int n) {
        for (int i = n; i >= 0; i--) {
            System.out.print(i + " ");
        }
    }
    Run | Debug
    public static void main(String[] args) {
        int n = 10;
        System.out.println("Deret bilangan dari " + n + " sampai 0 (rekursif):");
        tampilkanDeretRekursif(n);
        System.out.println();
        System.out.println("Deret bilangan dari " + n + " sampai 0 (iteratif):");
        tampilkanDeretIteratif(n);
    }
}
```

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).

```java
import java.util.Scanner;
public class Tugas2 {
    public static int penjumlahanRekursif(int n) {
        if (n == 1) {
            return 1;
        }
        return n + penjumlahanRekursif(n - 1);
    }
    Run | Debug
    public static void main(String[] args) {
        int f = 8;
        int hasil = penjumlahanRekursif(f);
        System.out.println("Hasil penjumlahan dari 1 sampai " + f + " adalah: " + hasil);
    }
}
```

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). Berikut ini adalah ilustrasinya dalam bentuk tabel.

Bulan ke-	Jumlah Pasangan		Total Pasangan
	Produktif	Belum Produktif	
1	0	1	1
2	0	1	1
3	1	1	2
4	1	2	3
5	2	3	5
6	3	5	8
7	5	8	13
8	8	13	21
9	13	21	34
10	21	34	55
11	34	55	89
12	55	89	144

```java
import java.util.Scanner;
public class Tugas3 {
    public static int hitungPasanganMarmut(int bulan) {
        if (bulan <= 2) {
            return 1;
        } else {
            return hitungPasanganMarmut(bulan - 1) + hitungPasanganMarmut(bulan - 2);
        }
    }
    Run | Debug
    public static void main(String[] args) {
        int bulan = 12;
        int jumlahPasangan = hitungPasanganMarmut(bulan);
        System.out.println("Jumlah pasangan marmut pada bulan ke-" + bulan + " adalah: " + jumlahPasangan);
    }
}
```