

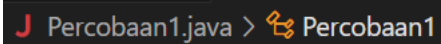
Nama : Muhammad Hikmal Al-Ghifary

Prodi : TI – 1B

Matkul : Praktikum Dasar Pemrograman

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.

```
2 static int faktorialRekursif (int n) {
3     if (n == 0) {
4         return (1);
5     } else {
6         return (n * faktorialRekursif(n - 1));
7     }
8 }
```

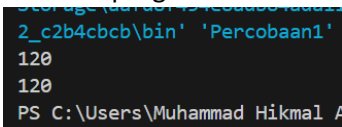
3. Buat lagi fungsi static dengan nama faktorialIteratif(), dengan tipe data kembalian fungsi int dan memiliki 1 parameter dengan tipe data int berupa bilangan yang akan dihitung nilai faktorialnya.

```
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 }
```

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

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

5. Jalankan program tersebut.



PERTANYAAN

1. Fungsi rekursif merupakan sebuah fungsi yang memanggil dirinya sendiri dalam proses eksekusinya. Fungsi ini digunakan untuk menyelesaikan masalah yang dapat dibagi menjadi sub-masalah yang lebih kecil dengan cara yang serupa. Rekursi sering digunakan dalam masalah yang memiliki struktur berulang.
2. Contoh kasus penggunaan fungsi rekursif adalah penghitungan faktorial, deret Fibonacci, atau traversing struktur data seperti pohon dan graf.
3. Ya hasil dari kedua fungsi sama. Baik faktorialRekursif() maupun faktorialIteratif() akan menghasilkan nilai faktorial yang sama, karena keduanya menghitung faktorial dari suatu bilangan dengan cara yang berbeda.

Rekrusif


- a. Fungsi memanggil dirinya sendiri hingga mencapai kondisi dasar
- b. Nilai faktorial dihitung dari belakang (dari 0 atau 1) ke depan.
- c. Menggunakan call stack untuk menyimpan setiap pemanggilan fungsi.
 1. Fungsi dipanggil pertama kali dengan $n = 5$.
 2. Fungsi terus memanggil dirinya sendiri dengan $n-1$ hingga mencapai $n = 0$.
 3. Setelah mencapai basis kasus, fungsi mulai kembali menyusun hasil (backtracking) hingga ke panggilan pertama.

Iteratif

- a. Fungsi ini menggunakan loop (for-loop) yang dimulai dari n dan mengalikan nilai faktor secara berulang hingga mencapai 1.
- b. Pada setiap iterasi, nilai faktor dikalikan dengan nilai i dalam loop, yang berkurang dari n hingga 1.
- c. Misalnya, ketika memanggil faktorialIteratif(5):
- d. Mulai dengan faktor = 1 dan $i = 5$.
 Pada iterasi pertama, faktor = $1 \times 5 = 5$,
 Pada iterasi kedua, faktor = $5 \times 4 = 20$,
 Pada iterasi ketiga, faktor = $20 \times 3 = 60$,
 Pada iterasi keempat, faktor = $60 \times 2 = 120$,
 Pada iterasi kelima, faktor = $120 \times 1 = 120$.

Percobaan 2

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

 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.

```
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 bertipe int dengan nama bilangan dan pangkat
5. Tambahkan kode berikut ini untuk menerima input dari keyboard
6. Lakukan pemanggilan fungsi hitungPangkat yang telah dibuat sebelumnya dengan mengirimkan dua nilai parameter.

```

1  import java.util.Scanner;
2  public class Percobaan2 {
3      static int hitungPangkat (int x, int y){
4          if (y == 0){
5              return (1);
6          } else {
7              return (x * hitungPangkat(x, y - 1));
8          }
9      }
10
11      Run | Debug
12      public static void main(String[] args) {
13          Scanner sc = new Scanner(System.in);
14          int bilangan, pangkat;
15
16          System.out.print(s:"Bilangan yang dihitung: ");
17          bilangan = sc.nextInt();
18          System.out.print(s:"Pangkat: ");
19          pangkat = sc.nextInt();
20          System.out.println(hitungPangkat(bilangan, pangkat));
21      }
22  }

```

7. Jalankan program tersebut.

```

2_c2b4cbb\bin' 'Percobaan2'
Bilangan yang dihitung: 4
Pangkat: 2
16
PS C:\Users\Muhammad Hikmal AG

```

PERTANYAAN

1. Pemanggilan fungsi hitungPangkat() secara berulang kali akan terus berlangsung hingga angka yang diinput mencapai nilai dasar yang ditentukan yaitu $y == 0$ dan mengembalikan nilai 1. Setelah mencapai $y == 0$, rekursi berhenti, dan hasil perhitungan dikembalikan melalui pemanggilan fungsi sebelumnya.
2. Tambahkan kode program untuk mencetak deret perhitungannya.

```

11      static void cetakDeret(int x, int y) {
12          for (int i = 0; i < y; i++) {
13              System.out.print(x);
14              if (i < y - 1) {
15                  System.out.print(s:" x ");
16              }
17          }
18          System.out.print(s:" x 1 = ");
19      }

```

```

31      cetakDeret(bilangan, pangkat);
32      System.out.println(hitungPangkat(bilangan, pangkat));

```

Percobaan 3

1. Pada project Rekursif, dan buat file Java dengan nama Percobaan3

```
J 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.

```

3      static double hitungLaba (double saldo, int tahun){
4          if (tahun == 0) {
5              return (saldo);
6          } else {
7              return (1.11 * hitungLaba(saldo, tahun - 1));
8          }
9      }

```

3. Buatlah fungsi main dan deklarasikan Scanner dengan nama sc
4. Buatlah sebuah variabel bertipe double dengan nama saldoAwal dan sebuah variabel bertipe int bernama tahun
5. Tambahkan kode untuk menerima input dari keyboard

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

```

11 public static void main(String[] args) {
12     Scanner sc = new Scanner(System.in);
13     int saldoAwal, tahun;
14
15     System.out.print(s:"Jumlah saldo awal: ");
16     saldoAwal = sc.nextInt();
17     System.out.print(s:"Lamanya investasi: ");
18     tahun = sc.nextInt();
19
20     System.out.print("Jumlah saldo setelah " + tahun + " tahun : ");
21     System.out.println(hitungLaba(saldoAwal, tahun));
22 }

```

- Jalankan program tersebut

```

-jobsheet12_c2b4cbcb\bin' 'Percobaan3'
Jumlah saldo awal: 200000
Lamanya investasi: 3
Jumlah saldo setelah 3 tahun : 273526.20000000007
PS C:\Users\Muhammad Hikmal AG\OneDrive\Documents

```

PERTANYAAN

- Base case** adalah kondisi yang menghentikan rekursi, yaitu saat kita mencapai kondisi dasar yang tidak lagi memerlukan pemanggilan rekursif. Base case dari kode program tersebut

```

4         if (tahun == 0) {
5             return (saldo);

```

Recursion call adalah kode yang memanggil fungsi itu sendiri selama kondisi base case belum terpenuhi. Recursion call dari kode program tersebut adalah

```

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

```

- Trace fase ekspansi dan fase substitusi algoritma perhitungan laba di atas jika diberikan nilai hitungLaba(100000,3) :

Fase ekspansi :

- 1.11 * hitungLaba(100000, 2)
- 1.11 * hitungLaba(100000, 1)
- 1.11 * hitungLaba(100000, 0)
- Basecase

Fase Substitusi :

- hitungLaba(100000, 0) = 100000
- hitungLaba(100000, 1) = 1.11 * 100000 = 111000
- hitungLaba(100000, 2) = 1.11 * 111000 = 123210
- hitungLaba(100000, 3) = 1.11 * 123210 = 136771,1

TUGAS

- DescendingRekursif

```

1  import java.util.Scanner;
2  public class DeretDescendingRekursif15 {
3      Run | Debug
4      public static void main(String[] args) {
5          Scanner sc = new Scanner(System.in);
6
7          System.out.print(s:"Masukkan sebuah angka: ");
8          int angka = sc.nextInt();
9
10         System.out.print(s:"Hasil (rekursif):  ");
11         descendingRekursif(angka);
12         System.out.println();
13
14         System.out.print(s:"Hasil (iteratif):  ");
15         descendingIteratif(angka);
16     }
17
18     static void descendingRekursif(int angka) {
19         if (angka < 0) {
20             return;
21         }
22         System.out.print(angka + " ");
23         descendingRekursif(angka - 1);
24     }
25
26     static void descendingIteratif(int angka) {
27         for (int i = angka; i >= 0; i--) {
28             System.out.print(i + " ");
29         }
30     }
31 }

```

2. PenjumlahanRekursif

```

2  public class PenjumlahanRekursif15 {
3      public static void main(String[] args) {
4
5          System.out.print(s:"===== KALKULATOR PENJUMLAHAN REKURSIF =====");
6          System.out.println();
7          System.out.print(s:"Masukkan angka bebas untuk dijumlahkan: ");
8          int f = sc15.nextInt();
9
10         System.out.print(s:"Hasil: ");
11         tampilkanDeretPenjumlahan(f, angka:1);
12
13         int hasil = hitungPenjumlahan(f);
14         System.out.println(" = " + hasil);
15     }
16
17     static int hitungPenjumlahan(int n) {
18         if (n == 0) {
19             return 0;
20         } else {
21             return n + hitungPenjumlahan(n - 1);
22         }
23     }
24
25     static void tampilkanDeretPenjumlahan(int n, int angka) {
26         if (angka > n) {
27             return;
28         }
29         System.out.print(angka);
30         if (angka < n) {
31             System.out.print(s:" + ");
32         }
33         tampilkanDeretPenjumlahan(n, angka + 1);
34     }
35 }

```

3. Fibonacci

```

1  public class Fibonacci15 {
2      static int fibonacci(int bulan){
3          if (bulan == 1 || bulan == 2) {
4              return 1;
5          }else{
6              return fibonacci(bulan - 1) + fibonacci(bulan -2);
7          }
8      }
9      Run | Debug
10     public static void main(String[] args) {
11         int bulan = 12;
12         int jmlMarmut = fibonacci(bulan);
13         System.out.println("Jumlah pasangan marmut pada akhir bulan ke-" + bulan + " adalah " + fibonacci(bulan));
14     }
15 }

```