

Modul Pemrograman Berbasis Objek (PBO)

Java Basic - Array list & Static Method Part 2

Modul ini melanjutkan modul sebelumnya, pada modul ini kita akan membahas tentang static method.

Semisal teman teman membuat suatu program untuk melakukan penambahan dan pengurangan

Pertambahan teman teman lakukan pada program tersebut hingga 5x, dan pengurangan hingga 3x. Jika teman teman melakukan secara manual, maka teman teman harus menulis kode pertambahan dan perulangan total sebanyak 8x.

Supaya tidak terjadi hal seperti itu, mari kita berkenalan dengan static method.

Static method adalah sebuah method yang dapat langsung di gunakan pada class tersebut tanpa perlu di inisiasi kedalam objek. Secara definisi seperti itu, dan akan kita bahas secara dalam mengenai method pada modul objek dan class yang akan datang.

Static method terbagi menjadi 2 yaitu prosedur dan fungsi, kita akan membahas apa itu prosedur, apa itu fungsi.

Mari kita mulai dengan prosedur. Buat sebuah file baru bernama Prosedur.java dan tulis kode dibawah ini pada file tersebut dan jalankan

```
1  class Prosedur {
2
3      public static void main(String args []) {
4          pertambahan();
5      }
6
7      static void pertambahan() {
8          int nilaiPertama = 10;
9          int nilaiKedua = 20;
10
11         int nilaiKetiga = nilaiPertama + nilaiKedua;
12
13         System.out.println("Hasil pertambahan dari " +
14                             nilaiPertama + " + " + nilaiKedua +
15                             " adalah = " + nilaiKetiga);
16     }
17 }
```

```
17 }

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 1: bash

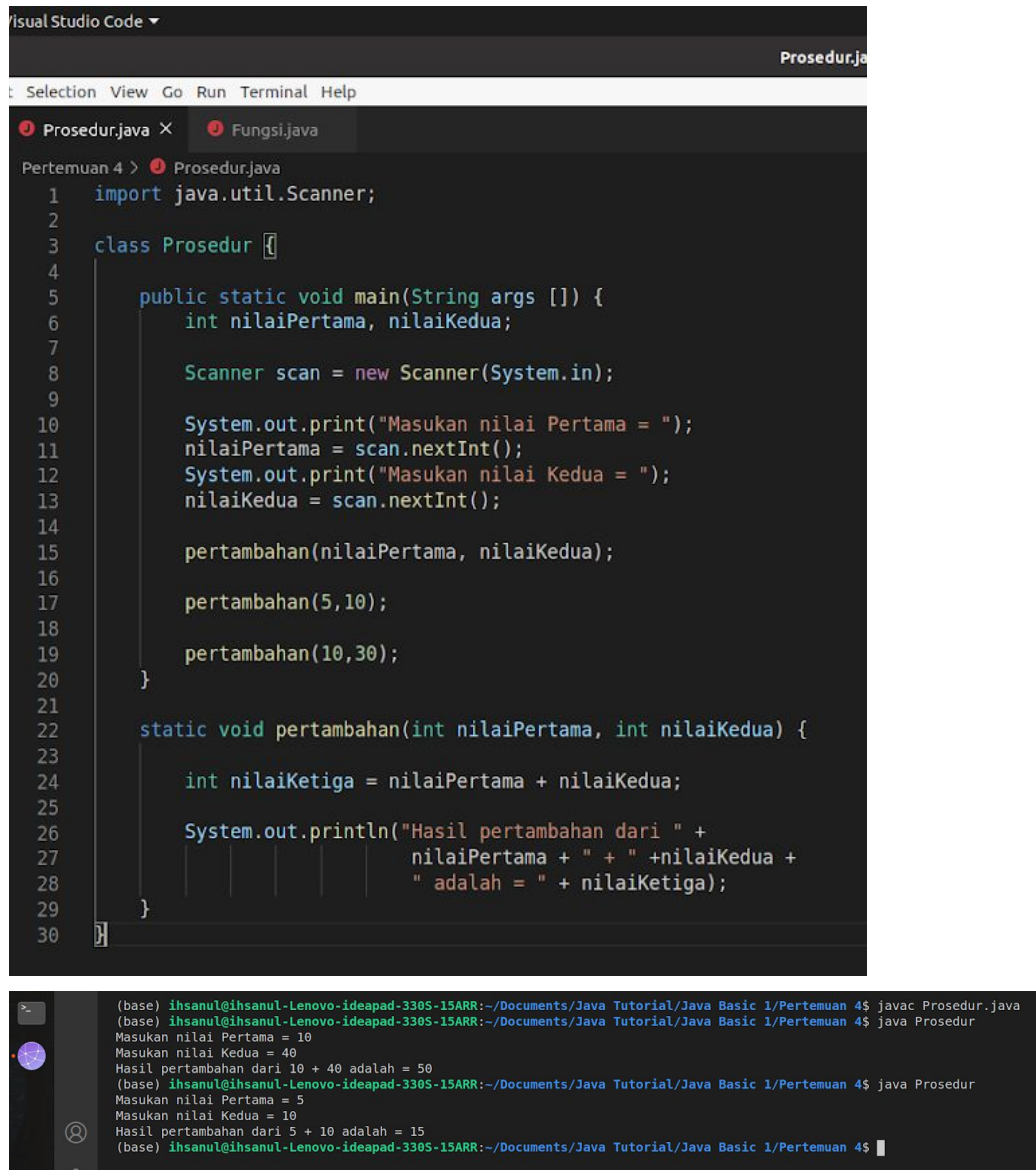
(base) ihsanul@ihsanul-Lenovo-ideapad-330S-15ARR:~/Documents/Java Tutorial/Java Basic 1/Pertemuan 4$ javac Prosedur.java
(base) ihsanul@ihsanul-Lenovo-ideapad-330S-15ARR:~/Documents/Java Tutorial/Java Basic 1/Pertemuan 4$ java Prosedur
Hasil pertambahan dari 10 + 20 adalah = 30
(base) ihsanul@ihsanul-Lenovo-ideapad-330S-15ARR:~/Documents/Java Tutorial/Java Basic 1/Pertemuan 4$
```

Pada contoh diatas, kita membuat 2 static void yaitu main() dan pertambahan(). static void main akan otomatis langsung di eksekusi pada program ketika pertama kali dijalankan. Sedangkan static void pertambahan() tidak langsung dijalankan melainkan harus dipanggil terlebih dahulu

```
1 class Prosedur {
2
3     public static void main(String args []) {
4         pertambahan();
5     }
6
7     static void pertambahan() {
8         int nilaiPertama = 10;
9         int nilaiKedua = 20;
10
11         int nilaiKetiga = nilaiPertama + nilaiKedua;
12
13         System.out.println("Hasil pertambahan dari " +
14                             nilaiPertama + " + " + nilaiKedua +
15                             " adalah = " + nilaiKetiga);
16     }
17 }
```

Kalian bisa lihat pada static void main, kita memanggil static void pertambahan dengan menuliskan pertambahan(). Setelah dipanggil, maka kode pada static pertambahan akan langsung dijalankan. Bagaimana jika static void pertambahan tidak dipanggil ? Maka kode pada static void method pertambahan tidak akan dieksekusi

Mari kita ubah sedikit kode diatas, sehingga static void pertambahan() dapat diberi inputan langsung oleh user



The image shows a screenshot of Visual Studio Code with a Java file named 'Prosedur.java' open. The code defines a class 'Prosedur' with a 'main' method and a static 'pertambahan' method. The 'main' method uses a 'Scanner' to take user input for two numbers, calls 'pertambahan' with those numbers, and also calls it with hardcoded values (5, 10) and (10, 30). The 'pertambahan' method calculates the sum of the two numbers and prints the result.

```
1 import java.util.Scanner;
2
3 class Prosedur {
4
5     public static void main(String args []) {
6         int nilaiPertama, nilaiKedua;
7
8         Scanner scan = new Scanner(System.in);
9
10        System.out.print("Masukan nilai Pertama = ");
11        nilaiPertama = scan.nextInt();
12        System.out.print("Masukan nilai Kedua = ");
13        nilaiKedua = scan.nextInt();
14
15        pertambahan(nilaiPertama, nilaiKedua);
16
17        pertambahan(5,10);
18
19        pertambahan(10,30);
20    }
21
22    static void pertambahan(int nilaiPertama, int nilaiKedua) {
23
24        int nilaiKetiga = nilaiPertama + nilaiKedua;
25
26        System.out.println("Hasil pertambahan dari " +
27                           nilaiPertama + " + " + nilaiKedua +
28                           " adalah = " + nilaiKetiga);
29    }
30 }
```

Below the code editor, the terminal window shows the execution of the program. It displays the prompts for input, the user's input (10 and 40), and the resulting output: 'Hasil pertambahan dari 10 + 40 adalah = 50'. It also shows the results of the hardcoded calls: 'Hasil pertambahan dari 5 + 10 adalah = 15' and 'Hasil pertambahan dari 10 + 30 adalah = 40'.

Pada contoh diatas, kita menulis

```
static void pertambahan(int nilaiPertama, int nilaiKedua) {
~kode~
}
```

nilaiPertama dan nilaiKedua bisa kita sebut sebagai parameter, dimana jika memberi parameter seperti diatas, kita dapat menambahkan nilai yang akan dipakai pada pemanggilan static method pertambahan pada static method main.

Contohnya, pada main() kita menerima inputan pada nilaiPertama dan nilaiKedua, lalu hasil inputan tersebut dimasukkan pada static method pertambahan(). Setelah itu

pertambahan() dieksekusi dimana inputan tersebut ditambahkan dan disimpan pada variabel nilaiKetiga dan setelah itu diberikan perintah untuk menampilkan teks.

Lalu, pada pemanggilan pertambahan yang kedua, kita memberi nilai 5 dan 10. Angka 5 menempati parameter nilaiPertama dan 10 menempati nilaiKedua, setelah angka 5 ditambah angka 10 dan disimpan pada variabel nilaiKetiga lalu ditampilkan.

Begitu pula dengan pemanggilan ketiga pertambahan pada main(). Pada pemanggilan ketiga yang dimasukan adalah 10 dan 30. 10 menempati parameter nilaiPertama, dan 30 menempati parameter nilaiKedua. Kedua parameter tersebut ditambahkan dan disimpan pada variabel nilaiKetiga lalu ditampilkan.

Static void pertambahan dapat kita sebut sebagai prosedur. Dimana pada contoh diatas, kita dapat menjalankan prosedur pertambahan beberapa kali tanpa perlu menuliskan kode yang sama berulang kali. Sehingga hal ini membuat kita lebih hemat dalam menulis kode

Selain prosedur terdapat juga fungsi, mari kita lihat langsung apa itu fungsi. Buat file baru bernama Fungsi.java dan masukan kode dibawah ini pada file tersebut

```
3  class Fungsi {
4
5      public static void main(String args []) {
6
7          int hasilTambah = pertambahan(10, 3);
8          int hasilPengurangan = pengurangan(5, 2);
9
10         System.out.println("Hasil pertambahan 10+3 = " + hasilTambah);
11         System.out.println("Hasil pengurangan 5-2 = " + hasilPengurangan);
12     }
13
14     public static int pertambahan(int nilaiPertama, int nilaiKedua) {
15         int hasilPertambahan = nilaiPertama+nilaiKedua;
16
17         return hasilPertambahan;
18     }
19
20     public static int pengurangan(int nilaiPertama, int nilaiKedua) {
21         int hasilPengurangan = nilaiPertama-nilaiKedua;
22
23         return hasilPengurangan;
24     }
25 }
```

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE  1: bash
(base) ihsanul@ihsanul-Lenovo-ideapad-330S-15ARR:~/Documents/Java Tutorial/Java Basic 1/Pertemuan 4$ javac Fungsi.java
(base) ihsanul@ihsanul-Lenovo-ideapad-330S-15ARR:~/Documents/Java Tutorial/Java Basic 1/Pertemuan 4$ java Fungsi
Hasil pertambahan 10+3 = 13
Hasil pengurangan 5-2 = 3
(base) ihsanul@ihsanul-Lenovo-ideapad-330S-15ARR:~/Documents/Java Tutorial/Java Basic 1/Pertemuan 4$
```

Contoh diatas, kita menyimpan fungsi penambahan pada variabel hasilTambah, lalu menyimpan fungsi pengurangan pada hasilPengurangan.

```
int hasilTambah = pertambahan(10, 3);  
int hasilPengurangan = pengurangan(5, 2);
```

Sedangkan pada contoh pertama tentang prosedur kita tidak menyimpan prosedur tersebut dalam suatu variabel, tapi langsung dipanggil seperti pada contoh dibawah ini

```
pertambahan(5,10);  
  
pertambahan(10,30);
```

Pada prosedur kita tidak memungkinkan untuk menyimpan prosedur pada variabel, sedangkan pada fungsi dimungkinkan. Kenapa hal tersebut memungkinkan ? Mari kita bandingkan kembali antara kodingan fungsi dan prosedur

Pada kodingan fungsi :

```
public static int pertambahan(int nilaiPertama, int nilaiKedua) {  
    int hasilPertambahan = nilaiPertama+nilaiKedua;  
  
    return hasilPertambahan;  
}
```

Pada kodingan prosedur :

```
21  
22     static void pertambahan(int nilaiPertama, int nilaiKedua) {  
23  
24         int nilaiKetiga = nilaiPertama + nilaiKedua;  
25  
26         System.out.println("Hasil pertambahan dari " +  
27             nilaiPertama + " + " + nilaiKedua +  
28             " adalah = " + nilaiKetiga);  
29     }  
30
```

Pertama pada kodingan fungsi, terdapat kata return. Dimana ketika fungsi dipanggil maka fungsi tersebut akan menghasilkan nilai sesuai yang tertera pada return

Maka pada contoh dibawah ini

```
int hasilTambah = pertambahan(10, 3);  
int hasilPengurangan = pengurangan(5, 2);
```

Variabel hasilTambah akan menyimpan nilai pada fungsi pertambahan. Dimana hasilTambah akan menyimpan angka 13.

Darimana angka 13 ?

```
public static int pertambahan(int nilaiPertama, int nilaiKedua) {  
    int hasilPertambahan = nilaiPertama+nilaiKedua;  
  
    return hasilPertambahan;  
}
```

dari variabel hasilPertambahan yang menyimpan pertambahan nilaiPertama ditambah nilaiKedua.

Hal ini juga berlaku pada variabel hasilPengurangan, dimana akan menghasilkan angka 3

```
public static int pengurangan(int nilaiPertama, int nilaiKedua) {  
    int hasilPengurangan = nilaiPertama-nilaiKedua;  
  
    return hasilPengurangan;  
}
```

angka 3 didapatkan dari variabel hasilPengurangan yang menyimpan nilai pengurangan dari nilaiPertama dikurang nilaiKedua.

Dan hal tersebut lah yang membuat fungsi dapat disimpan pada variabel.

Sedangkan pada prosedur tidak ada nilai yang dikembalikan karena tidak terdapat return

```
21  
22     static void pertambahan(int nilaiPertama, int nilaiKedua) {  
23  
24         int nilaiKetiga = nilaiPertama + nilaiKedua;  
25  
26         System.out.println("Hasil pertambahan dari " +  
27             nilaiPertama + " + " + nilaiKedua +  
28             " adalah = " + nilaiKetiga);  
29     }  
30
```


Dan perbedaan berikut nya antara fungsi dan prosedur adalah :

Prosedur :

```
21  
22     static void pertambahan(int nilaiPertama, int nilaiKedua) {  
23  
24         int nilaiKetiga = nilaiPertama + nilaiKedua;  
25  
26         System.out.println("Hasil pertambahan dari " +  
27             nilaiPertama + " + " + nilaiKedua +  
28             " adalah = " + nilaiKetiga);  
29     }  
30
```

Terdapat void pada pertambahan

Fungsi :

```
public static int pertambahan(int nilaiPertama, int nilaiKedua) {  
    int hasilPertambahan = nilaiPertama+nilaiKedua;  
  
    return hasilPertambahan;  
}
```

Terdapat int pada pertambahan

Jika pada fungsi kita harus menambahkan tipe data yang akan kita gunakan pada fungsi tersebut, sedangkan pada prosedur kita tidak perlu menambahkan tipe data tetapi kita harus menuliskan kata void pada prosedur.

Mari kita sedikit bermain dengan kode pada file Fungsi.java

Ubah kode yang terdapat pada main menjadi seperti dibawah ini (Masih didalam file fungsi.java)

```
public static void main(String args []) {
    int nilaiPertama, nilaiKedua, nilaiKetiga;
    String pilihan;
    Scanner scan = new Scanner(System.in);

    System.out.print("Masukan nilai pertama = ");
    nilaiPertama = scan.nextInt();

    System.out.print("Masukan nilai kedua = ");
    nilaiKedua = scan.nextInt();
    scan.nextLine();

    System.out.println("Mau melakukan pertambahan atau pengurangan ? ");
    System.out.println("A. tulis 'tambah' untuk melakukan pertambahan");
    System.out.println("B. tulis 'kurang' untuk melakukan pengurangan");
    System.out.print("Pilihan mu = ");
    pilihan = scan.nextLine();

    switch (pilihan.toLowerCase()) {
        case "tambah":
            nilaiKetiga = pertambahan(nilaiPertama, nilaiKedua);
            System.out.println("Hasil pertambahan " + nilaiPertama + "+" + nilaiKedua + " = " + nilaiKetiga);
            break;
        case "kurang":
            nilaiKetiga = pengurangan(nilaiPertama, nilaiKedua);
            System.out.println("Hasil pengurangan " + nilaiPertama + "-" + nilaiKedua + " = " + nilaiKetiga);
            break;
        default:
            System.out.println("Pilihan tidak tersedia");
            break;
    }
}
```

Pada contoh kodingan diatas, kita membuat nilaiPertama dan nilaiKedua di input langsung oleh User di terminal.

Lalu, kita buat sebuah pilihan kepada user, dimana user bisa memilih untuk melakukan pertambahan dengan menulis tambah atau melakukan pengurangan dengan menulis kurang.

Jika user menulis tambah, maka fungsi pertambahan akan dijalankan dan disimpan pada variabel nilaiKetiga dan ditampilkan.

Jika user menulis kurang, maka fungsi pengurangan akan dijalankan dan disimpan pada variabel nilaiKetiga dan ditampilkan.

Jika user menulis selain kedua hal tersebut, maka akan muncul tulis pilihan tidak tersedia

--- Notes ---

Pada contoh diatas, kita melihat `scan.nextInt()` untuk menerima inputan `nilaiPertama` dan `nilaiKedua`.

Di penjelasan modul sebelum nya, jika kita menginput angka dengan `scan.nextInt()`, maka jika terdapat inputan berikutnya berupa teks, kita harus menutup baris terlebih dahulu, baru menjalankan perintah inputan teks kembali. Tetapi hal ini tidak berlaku jika setelah menginput angka, maka yang di input adalah angka kembali dengan menggunakan `scan.nextInt()`

```
System.out.print("Masukan nilai pertama = ");
nilaiPertama = scan.nextInt();

System.out.print("Masukan nilai kedua = ");
nilaiKedua = scan.nextInt();
scan.nextLine();

System.out.println("Mau melakukan penambahan atau pengurangan ? ");
System.out.println("A. tulis 'tambah' untuk melakukan penambahan");
System.out.println("B. tulis 'kurang' untuk melakukan pengurangan");
System.out.print("Pilihan mu = ");
pilihan = scan.nextLine();
```

Setelah inputan `nilaiKedua`, kita ingin menginput huruf kembali, maka baru hal ini berlaku kembali, dan karena itu pula kita menutup baris dengan perintah `scan.nextLine()`. Dan setelah itu kita baru bisa menginput huruf kembali karena baris sudah ditutup.

--- End Notes ---

Tugas

1. Buatlah laporan terkait praktikum modul Java Basic - Array list & Static Method Part 1 dan part 2
2. Buatlah program dari pernyataan dibawah ini dengan bahasa pemrograman java.
 - Buatlah sebuah `arrayList` yang menerima inputan berupa nama nasabah dan jumlah tabungan yang dia punya
 - Buat fungsi untuk menambahkan tabungan dan mengambil tabungan
 - Jika fungsi menambahkan tabungan dijalankan maka tabungan yang dimiliki nasabah tersebut akan bertambah sesuai jumlah yang ditabung
 - Jika fungsi mengambil tabungan dijalankan, maka tabungan yang dimiliki nasabah tersebut akan berkurang sesuai jumlah yang diambil

3. Screenshot kode dan hasil kode yang dijalankan. Masukkan hasil screenshot tersebut dan jelaskan.

Tugas dikirim pada elen, dengan format sebagai berikut

1. Pada penamaan file gunakan format berikut

Praktikum keberapa_Judul Praktikum_Nama_Nim

Contoh :

Praktikum01_Perkenalan-Java_IhsanulFikriAbiyyu_0220318021

2. Format laporan bertipe pdf
3. Pada cover praktikum, masukan beberapa hal dibawah ini
 - Praktikum ke berapa
 - Judul Praktikum
 - Nama
 - Nim
4. Tenggat waktu mengerjakan adalah 1 pekan semenjak tugas diberikan