

JobSheet - Week 1 - Fundamental Programming Structures in Java

Nama: Firon Fariz Sahuleka

NIM: 251524101

Kelas: 1D

Repo GitHub: <https://github.com/fironsahuleka2007-art/Teknik-Pemrograman-Week-1>

Instruksi Pengerjaan:

1. Kerjakan 5 soal di bawah ini dengan melengkapi setiap kolom jawaban yang disediakan pada jobsheet ini.
2. Jawaban setiap soal mencakup source code, screenshot hasil dari program yang ditampilkan full screen termasuk taskbar (tambahkan beberapa screenshot jika diperlukan), penjelasan permasalahan dan solusi yang dihadapi, nama teman yang membantu memecahkan masalah (opsional).
3. Dikumpulkan pada Assignment Classroom sesuai dengan deadline yang tertera pada assignment tersebut.
4. Format penamaan file jobsheet: W1_P_<Kelas 1X>_<3 Digit_NIM_Terakhir>.docx/pdf. Contoh: W1_P_1B_001.docx/pdf.
5. Buatlah satu file java yang mengandung jawaban dalam bentuk source untuk satu jawaban yang dapat langsung dieksekusi. Contoh penamaan: 1-DataTypes.java, 2-Variables.java, 3-Arithmetic.java, 4-TypeCasting.java, dan 5-Operator.java.
6. Submit semua jawaban dalam bentuk file java pada repository GitHub masing-masing.

No. 1 Data Types

Soal Praktikum
<p>Java memiliki 8 tipe data primitif; char, boolean, byte, short, int, long, float, dan double.</p> <p>Untuk praktikum ini, kita akan Latihan dengan tipe data primitif yang digunakan untuk menyimpan nilai bilangan bulat, yaitu byte, short, int, dan long.</p> <ul style="list-style-type: none">• A byte is an 8-bit signed integer.• A short is a 16-bit signed integer.• An int is a 32-bit signed integer.• A long is a 64-bit signed integer. <p>Dengan diberikan sebuah bilangan bulat masukan, Anda harus menentukan tipe data primitif mana yang mampu menyimpan masukan tersebut dengan benar.</p>

Input Format

Baris pertama berisi bilangan bulat, T , yang menunjukkan jumlah kasus uji. Setiap kasus uji, T , terdiri dari satu baris dengan bilangan bulat, n , yang nilainya bisa sangat besar atau sangat kecil.

Output Format

Untuk setiap variabel masukan n dan tipe data primitif yang sesuai, Anda harus menentukan apakah tipe data primitif yang diberikan mampu menyimpannya. Jika ya, maka cetak:

```
N can be fitted in:
* datatype
```

Jika terdapat lebih dari satu tipe data yang sesuai, cetak masing-masing pada barisnya sendiri dan urutkan berdasarkan ukurannya (misalnya: **byte < short < int < long**).

Jika angka tersebut tidak dapat disimpan dalam salah satu dari empat tipe data primitif yang disebutkan di atas, cetak baris berikut:

N can't be fitted anywhere

Sample Input:

[illegible]

Sample Output:

[illegible]

Explanation:

Angka 150 dapat disimpan dalam tipe data short, int, atau long. Angka 213333333333333333333333333333 sangat besar dan berada di luar rentang nilai yang diizinkan untuk tipe data primitif yang dibahas dalam masalah ini.

Source Code

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */

```

```

/**
 *
 * @author firon
 */
import java.util.Scanner;
import java.math.BigInteger;

public class MainTest {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        int jumlahData = scan.nextInt();

        for (int i = 0; i < jumlahData; i++) {
            String nStr = scan.next();

            try {
                BigInteger angka = new BigInteger(nStr);

                boolean fit = false;

                System.out.println(nStr + " can be fitted in:");

                if (angka.compareTo(BigInteger.valueOf(Byte.MIN_VALUE)) >= 0 &&
                    angka.compareTo(BigInteger.valueOf(Byte.MAX_VALUE)) <= 0) {
                    System.out.println("* byte");
                    fit = true;
                }

                if (angka.compareTo(BigInteger.valueOf(Short.MIN_VALUE)) >= 0 &&
                    angka.compareTo(BigInteger.valueOf(Short.MAX_VALUE)) <= 0) {
                    System.out.println("* short");
                    fit = true;
                }

                if (angka.compareTo(BigInteger.valueOf(Integer.MIN_VALUE)) >= 0 &&
                    angka.compareTo(BigInteger.valueOf(Integer.MAX_VALUE)) <= 0) {
                    System.out.println("* int");
                    fit = true;
                }

                if (angka.compareTo(BigInteger.valueOf(Long.MIN_VALUE)) >= 0 &&
                    angka.compareTo(BigInteger.valueOf(Long.MAX_VALUE)) <= 0) {
                    System.out.println("* long");
                    fit = true;
                }

                if (!fit) {
                    System.out.println(nStr + " can't be fitted anywhere.");
                }

            } catch (Exception e) {
                System.out.println(nStr + " can't be fitted anywhere.");
            }
        }
    }
}

```

[illegible]

Penjelasan Permasalahan dan Solusi

Dalam program diatas kita diminta untuk mengelompokan berbagai jenis angka ke dalam beberapa bentuk data primitif yaitu byte, integer, short, dan juga long. Dalam pemrograman Java, setiap tipe data primitif memiliki batas nilai minimum dan maksimum yang dapat disimpan.

- A byte is an 8-bit signed integer.
- A short is a 16-bit signed integer.
- An int is a 32-bit signed integer.
- A long is a 64-bit signed integer.

Tidak semua bilangan dapat disimpan dalam tipe data tertentu. Jika program tetap memaksakan penyimpanan nilai yang melebihi batas tipe data, maka dapat terjadi **overflow yang menyebabkan kesalahan data**. Jika nilai input berada dalam rentang suatu tipe data, maka tipe data tersebut dicetak sebagai tipe yang dapat menyimpan nilai tersebut. Jika nilai input berada di luar semua rentang tipe data yang tersedia, maka program menampilkan bahwa nilai tersebut tidak dapat disimpan pada tipe data primitif yang diberikan.

Diatas saya mencoba untuk menggunakan data yang telah dibuat di soal sebelumnya dimana saya memasukan 5 variabel berbeda untuk diperiksa setiap variabel tersebut apakah sesuai atau tidak:

-150

-150 can be fitted in:

- short
- int
- long

150000

150000 can be fitted in:

- int
- long

150000000

150000000 can be fitted in:

- int
- long

213333333333333333333333333333333333

2133333333333333333333333333333333 can be fitted in:

- 21333333333333333333333333333333 can't be fitted anywhere.

-10000000000000000

-10000000000000000 can be fitted in:

- long

Nama Teman Hal yang Dibantu (Opsional)

No. 2 Variables

Soal Praktikum

Perhatikan dua bagian program di bawah ini.

Bagian 1:

```
public class Constants {
    public static void main(String[] args) {
        final double CM_PER_INCH = 2.54; double paperWidth = 8.5;
        double paperHeight = 11;
        System.out.println("Paper size in centimeters: " + paperWidth
* CM_PER_INCH + " by " + paperHeight * CM_PER_INCH);
    }
}
```

Bagian 2:

```
public class Constants2 {
    public static final double CM_PER_INCH = 2.54; public static void
main(String[] args) {
        double paperWidth = 8.5; double paperHeight = 11;
        System.out.println("Paper size in centimeters: " + paperWidth
* CM_PER_INCH + " by " + paperHeight * CM_PER_INCH);
    }
}
```

Dari 2 contoh baris program diatas, jawablah pertanyaan dibawah ini:

1. Bagaimana output dari masing masing class Constants dan Constants2?
2. Apa perbedaan penggunaan final double dengan public static final double?

Source Code

Constants.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */

/**
 *
 * @author firon
 */
public class Constants {
    public static void main(String[] args) {
        final double CM_PER_INCH = 2.54; double paperWidth = 8.5;
        double paperHeight = 11;
        System.out.println("Paper size in centimeters: " + paperWidth
* CM_PER_INCH + " by " + paperHeight * CM_PER_INCH);
    }
}
```

Constants2.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */

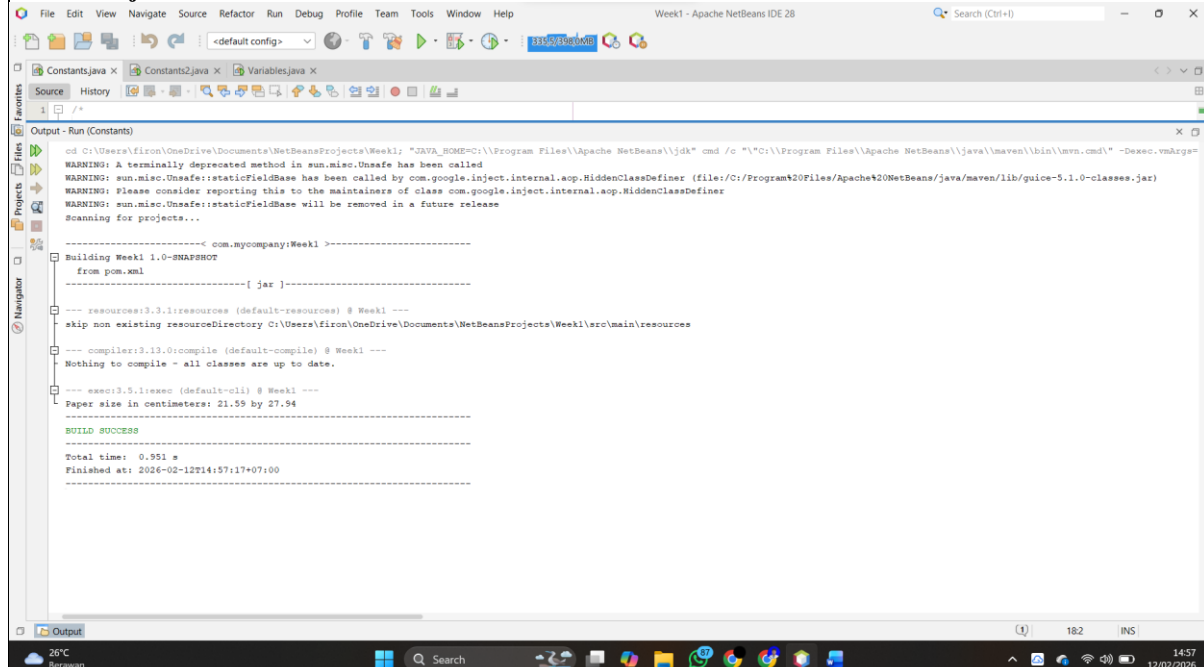
/**
 *
 * @author firon
 */
public class Constants2 {
    public static final double CM_PER_INCH = 2.54; public static void
    main(String[] args) {

        double paperWidth = 8.5; double paperHeight = 11;
        System.out.println("Paper size in centimeters: " + paperWidth

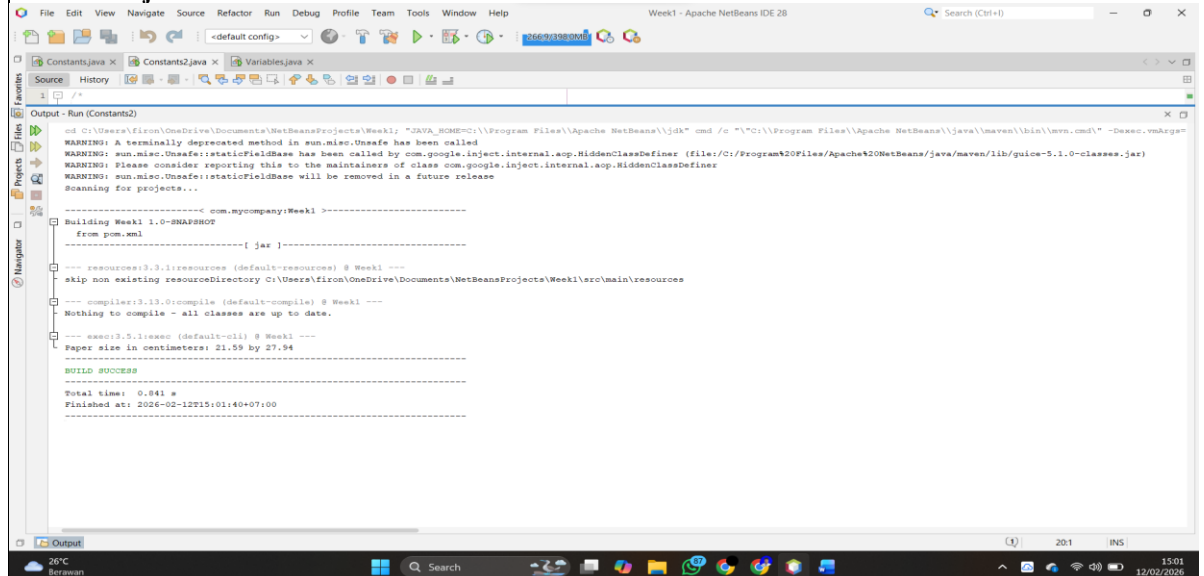
        * CM_PER_INCH + " by " + paperHeight * CM_PER_INCH);
    }
}
```

Screenshot Hasil

Constants.java



Constants2.java

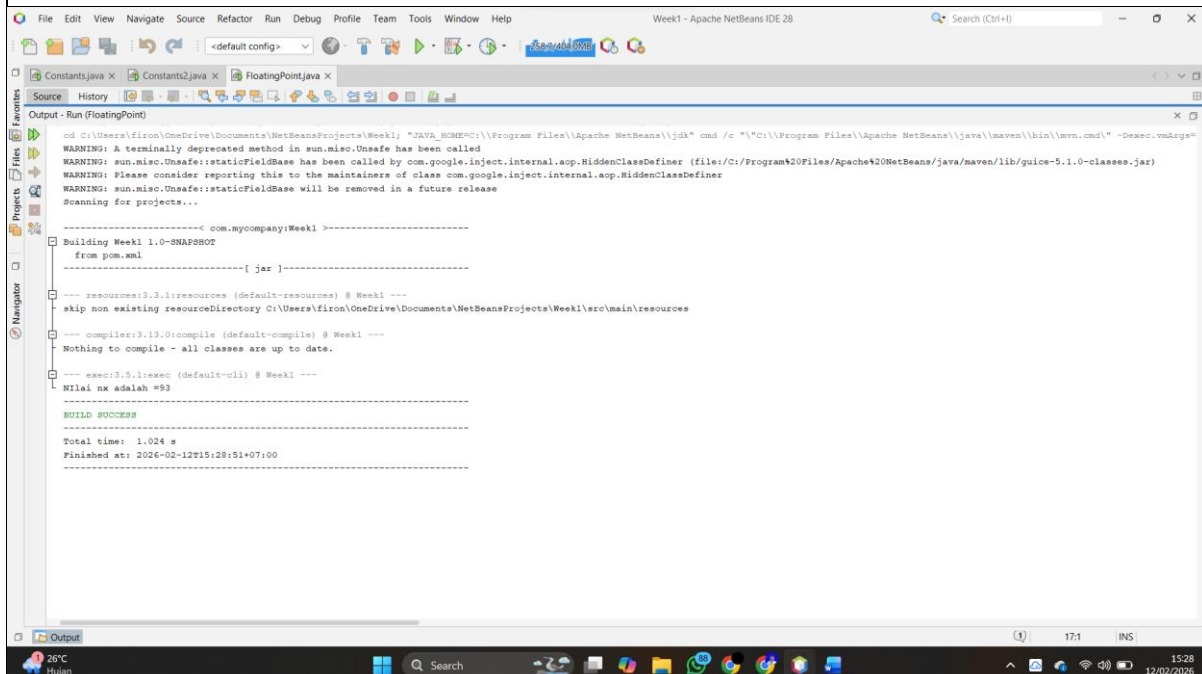


Penjelasan Permasalahan dan Solusi
<p>Dari hasil pengamatan yang sudah saya lakukan dan dari hasil source code yang sudah saya coba di netbeans menghasilkan output yang sama yaitu "Paper size in centimeters: 21.59 by 27.94" kedua program tersebut sama-sama ingin mengubah suatu ukuran kertas yang berasal dari ukuran inch menjadi centimeter. Hal tersebut menghasilkan nilai dan output yang sama karena diprogram dengan rumus serta nilai ukuran kertas yang sama mulai dari tinggi dan lebar kertasnya, dengan perhitungan sebagai berikut :</p> <p>Lebar = $8.5 \times 2.54 = 21.59$ cm Tinggi = $11 \times 2.54 = 27.94$ cm</p> <p>Yang membedakan Constants.java dengan Constants2.java bagaimana fungsi dari final double dan juga public static final double disana kita bahwa nilai pada Constants.java bersifat mutlak nilainya tidak bisa diubah dan hanya bisa digunakan pada method itu saja jika ingin dilakukan pada method lain perlu melakukan deklarasi ulang, lalu untuk Constants2.java itu merupakan variabel yang bisa digunakan pada method lain dan bisa dipanggil dimana saja jika command tersebut diperlukan. Dari kedua hal tersebut saya bisa simpulkan mirip dengan modular programming memiliki variabel lokal dan juga global. Final double digunakan ketika konstanta hanya diperlukan di dalam satu method saja, sedangkan public static final double digunakan ketika method atau class lain memerlukannya. Jadi keduanya tergantung situasi digunakannya.</p>
Nama Teman dan Hal yang Dibantu (Opsional)

No. 3 Arithmetic - Math Class

Soal Praktikum
<p>Perhatikan bagian program di bawah ini.</p> <pre> Class FloatingPoint { public static void main(String[] args) { double x = 92.98; int nx = (int) Math.round(x); } } </pre>
<p>Math Class berisi bermacam-macam fungsi matematika seperti pada contoh diatas pada penggunaan round(x), terdapat beberapa pertanyaan yang perlu untuk dijelaskan:</p> <ol style="list-style-type: none"> 1. Pada kasus berikut jelaskan nilai nx setelah digunakan Math.round(x)! 2. Kenapa dibutuhkan cast (int) dalam penggunaan Math.round(x)?
Source Code
<p>FloatingPoint.java</p> <pre> /* * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template */ /** * * @author firon */ public class FloatingPoint { public static void main(String[] args) { double x = 92.98; int nx = (int) Math.round(x); System.out.println("Nilai nx adalah =" + nx); } } </pre>

Screenshot Hasil



Penjelasan Permasalahan dan Solusi

Fungsi dari command round adalah untuk membulatkan sebuah nilai desimal dimana nilai awal x adalah 92,98 sedangkan pada program meminta nilai bertipe integer atau bilangan bulat oleh karena perlu adanya command round untuk membulatkan nilai tersebut. Nilai Desimal ≥ 0.5 akan dibulatkan naik Desimal < 0.5 akan dibulatkan turun.

```
public class FloatingPoint {
    public static void main(String[] args) { double x = 92.49;
        int nx = (int) Math.round(x);
        System.out.println("Nilai nx adalah =" + nx);
    }
}
```

*Ketika saya mencoba untuk mengubah nilai menjadi 92.49 maka ia akan membulatkan nilai kebawah menjadi 92

```
Nilai nx adalah =92
-----
BUILD SUCCESS
-----
Total time: 0.837 s
Finished at: 2026-02-12T15:43:25+07:00
-----
```

Casting sangat diperlukan karena nilai x awalnya bertipe double dimana itu merupakan tipe data primitif untuk menggunakan angka desimal lalu method tersebut ingin mengubah nilainya menjadi sebuah int atau bilangan bulat oleh karena itu casting digunakan untuk merubah tipe double menjadi integer.

Pada source code yang diberikan memiliki kekurangan dimana ia tidak bisa menampilkan output oleh karena itu saya mencoba untuk menambahkan "System.out.println("Nilai nx adalah =" + nx);" sehingga program tersebut bisa menghasilkan sebuah output di layar.

Nama Teman dan Hal yang Dibantu (Opsional)

No. 4 Type Casting/ Data Type Conversion

Soal Praktikum

Perhatikan baris program dibawah ini:

```
class ConvertDataType {  
    static short methodOne(long l) {  
        int i = (int) l; return (short)i;  
    }  
  
    public static void main(String[] args) {  
        double d = 10.25; float f = (float) d;  
        byte b = (byte) methodOne((long) f); System.out.println(b);  
    }  
}
```

Program berikut melakukan convert tipe data yang berukuran besar ke kecil (long -> int -> short) dan (double -> float -> byte).

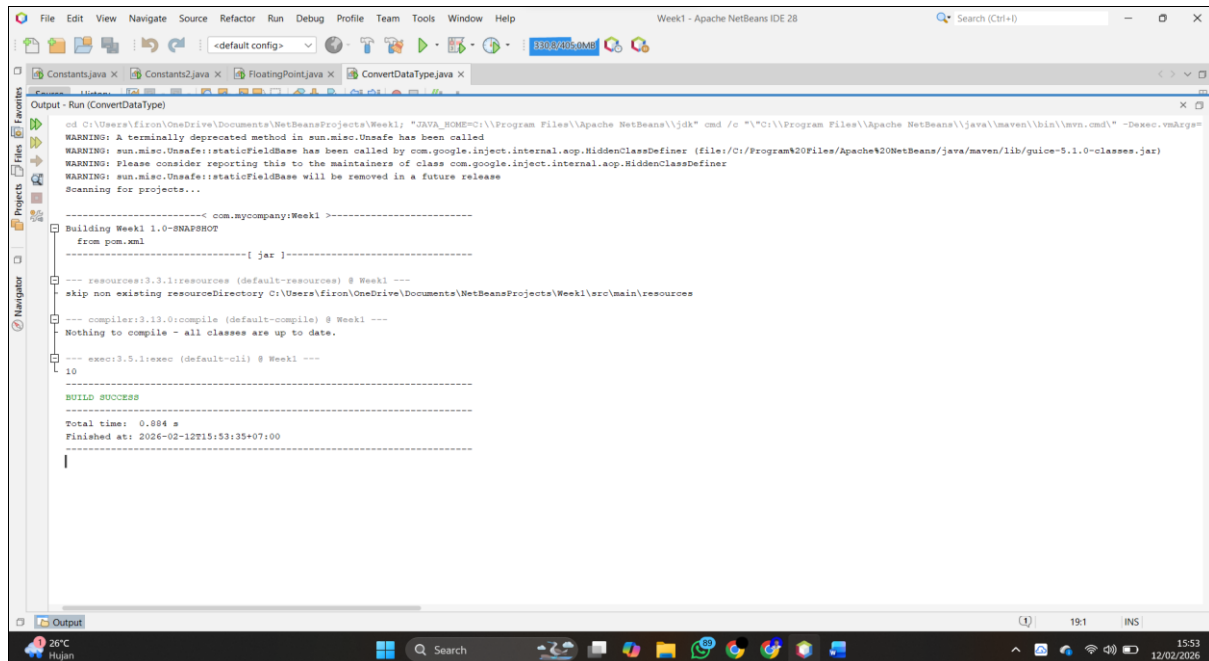
1. Jelaskan output nilai dari variable b.
2. Jelaskan apa yang berubah dari variable d menjadi variable b setelah dilakukan cast?

Source Code

ConvertDataType.java

```
/*  
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  
 */  
  
/**  
 *  
 * @author firon  
 */  
class ConvertDataType {  
    static short methodOne(long l) {  
        int i = (int) l; return (short)i;  
    }  
    public static void main(String[] args) { double d = 10.25; float f = (float) d;  
        byte b = (byte) methodOne((long) f); System.out.println(b);  
    }  
}
```

Screenshot Hasil



Penjelasan Permasalahan dan Solusi

Nilai output variable b adalah 10 karena pada awalnya ia merupakan nilai bertipe double dengan nilai 10.25 lalu diubah atau menggunakan metode **cast menjadi tipe data float** dengan nilai yang sama 10.25 karena pada tipe float masih bisa menyimpan nilai dengan bentuk desimal. Setelah diubah menjadi tipe data float nilai kembali di konversi menjadi **tipe data long** dimana tipe data ini tidak bisa menyimpan nilai yang bersifat desimal sehingga ia berubah menjadi nilai 10, lalu ia mencoba untuk mengubah nilai tersebut menjadi tipe data short dan sama masih dengan nilai 10 lalu ia mengubah lagi ke integer dengan nilai yang sama 10 lalu yang terakhir ia melakukan **cast menjadi sebuah byte dengan nilai yang sama yaitu 10**. Pada program tersebut ia mencoba untuk menghasilkan ke layar dalam bentuk byte oleh karena itu output yang keluar adalah 10. Jika diperhatikan kompleksitas nilai yang muncul menjadi lebih sederhana namun jika kita memerlukan nilai hingga angka desimalnya maka method dari program ini kurang baik untuk digunakan.

Nama Teman dan Hal yang Dibantu (Opsional)

No. 5 Operator

Soal Praktikum

Perhatikan bagian program di bawah ini.

```
class OperatorChallenge {
    public static void main(String[] args) {
        int a = 5;
        int b = 10;

        boolean result = (++a * 2 > b) && (b++ % 3 == 1);

        System.out.println("Hasil Boolean: " + result);
        System.out.println("Nilai a: " + a);
        System.out.println("Nilai b: " + b);
    }
}
```

Pertanyaan Analisis:

1. **Analisis Langkah Demi Langkah:** Jelaskan urutan eksekusi pada baris `boolean result`. Mana yang dijalankan lebih dulu antara `++a` dan perkalian `*`?
2. **Short-Circuit Logic:** Jika bagian pertama `(++a * 2 > b)` bernilai `false`, apakah bagian kedua `(b++ % 3 == 1)` akan tetap dieksekusi oleh Java? Jelaskan dampaknya pada nilai akhir variabel `b`.
3. **Output:** Berapakah nilai akhir dari `result`, `a`, dan `b`?

Source Code

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */

/**
 *
 * @author firon
 */
class OperatorChallenge {
    public static void main(String[] args) { int a = 5;
    int b = 10;
    boolean result = (++a * 2 > b) && (b++ % 3 == 1);
    System.out.println("Hasil Boolean: " + result); System.out.println("Nilai a: " + a); System.out.println("Nilai b:
    " + b);
    }
}
```

Screenshot Hasil

```

C:\Users\Firon\OneDrive\Documents\NetBeansProjects\Week1> "C:\Program Files\Apache NetBeans\jdk" cmd /c "C:\Program Files\Apache NetBeans\java\maven\bin\mvn.cmd" -Dexec.vmArgs=
WARNING: A terminally deprecated method in sun.misc.Unsafe has been called
WARNING: sun.misc.Unsafe:staticFieldBase has been called by com.google.inject.internal.asp.HiddenClassDefiner (file: C:\Program Files\Apache NetBeans\java\maven\lib\guice-5.1.0-classes.jar)
WARNING: Please consider reporting this to the maintainers of class com.google.inject.internal.asp.HiddenClassDefiner
WARNING: sun.misc.Unsafe:staticFieldBase will be removed in a future release
Scanning for projects...

-----< com.mycompany:Week1 >-----
Building Week1 1.0-SNAPSHOT
from pom.xml

-----[ jar ]-----
--- resources:3.3.1:resources (default-resources) @ Week1 ---
skip non existing resourceDirectory C:\Users\Firon\OneDrive\Documents\NetBeansProjects\Week1\src\main\resources
--- compiler:3.13.0:compile (default-compile) @ Week1 ---
Nothing to compile - all classes are up to date.
--- exec:3.5.1:exec (default-cli) @ Week1 ---
Hasil Boolean: true
Nilai a: 6
Nilai b: 11

BUILD SUCCESS

Total time: 0.870 s
Finished at: 2026-02-12T16:11:42+07:00
  
```

Penjelasan Permasalahan dan Solusi

Dari masalah diatas hal yang pertama kali dilakukan ialah prefix increment yaitu menambahkan nilai variabel sebelum digunakan **++a** mengibaratkan untuk menambah 1 nilai jadi jika **++a** maka nilai yang ditambahkan ialah **1+5** dan nilai **a** akan menjadi **6**. Setelah hal tersebut dilakukan maka ia akan dikalikan dengan 2 dan menghasilkan nilai 12. Lalu setelah itu ia kan melihat dalam bentuk boolean apakah nilai **a** lebih besar dan jawabannya ya **12>10** maka bernilai **true** lalu lanjut ke logika selanjutnya lalu lakukan kembali program aritmatika dimana **nilai b++** berbeda dengan **++b** ia tidak menambahkan nilai terlebih dahulu jadi masih memakai nilai **b** yaitu **10**. Setelah itu **10%3** artinya modulus atau sisa dari pembagian tersebut adalah 1 dan terjadi proses boolean apakah **1==1** dan jawabannya **true** jadi **true && true** akan menghasilkan juga **true** sehingga output yang keluar adalah **Hasil boolean : true, nilai a : 6 dan nilai b : 11** karena ia sudah ditambahkan satu setelah operasi aritmatika. Untuk menjawab soal nomor 2 jika proses aritmatika pertama sudah menghasilkan **false** maka proses aritmatika selanjutnya tidak akan dilakukan dan variabel **b** akan tetap bernilai **10** dan tidak berubah menjadi **11**

Nama Teman dan Hal yang Dibantu (Opsional)