

Nama & NPM	Topik:	Tanggal:
Melisa Yunita Sari G1F024026	Tuga Individu Operator Java	17 September 2024

[No. 1] Identifikasi Masalah:

```
public class OperatorAritmatika {
    public static void main(String[] args) {
        // deklarasi nilai
        int a = 20, b = 3;
        //operator aritmatika
        System.out.println("a: " +a);
        System.out.println("b: " +b);
        System.out.println("a + b = " + (a - b));
    } }
```

1. Tambahkan baris `System.out.println("a + b = " + (a + b));` Ubahlah operator (+) dengan tanda (-, *, /, %)
2. Analisa perhitungan matematika yang terjadi!

[No.] Analisis dan Argumentasi

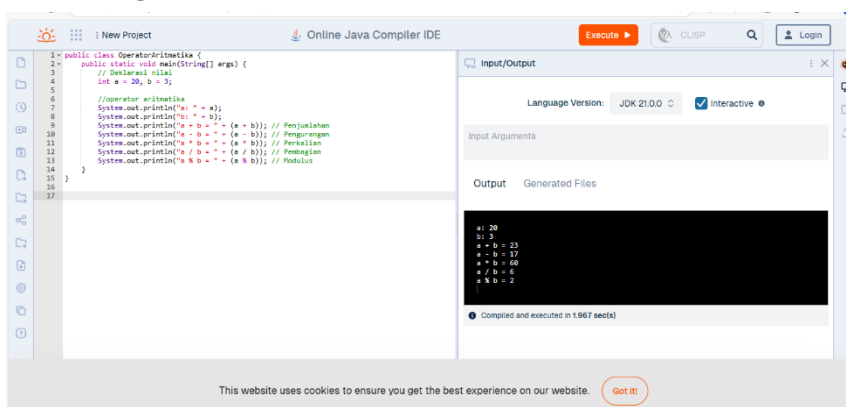
- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara memastikan bahwa semua operator aritmatika dapat digunakan dengan benar sesuai dengan tujuan.
- 2) Alasan solusi ini karena kesalahan dalam penggunaan operator dapat menghasilkan output yang salah, yang mana membingungkan pengguna dan tidak mencerminkan perhitungan yang tepat.
- 3) Perbaiki kode program dengan cara menambahkan menambahkan baris output, memastikan bahwa hasil yang di tampilkan benar dan jelas sesuai yang ada pada kode.

No.[1] Penyusunan Algoritma dan Kode Program

1) Algoritma

- a. Mulai
- b. Deklarasikan Variabel
- c. Inisialisasi Nilai
- d. Tampilkan nilai
- e. Hitung dan tampilkan hasil
- f. Selesai

2) Kode Program Dan Luaran



▪ **Analisa luaran yang dihasilkan**

Luaran sesuai dengan kode operasi aritmatika tersebut, program ini menghasilkan lima keluaran dari operasi aritmatika.

▪ **Analisa perhitungan matematika yang terjadi**

1. Penjumlahan ($a+b$)
 - Operasi: $20+3$
 - Hasil: 23
 - Analisis: Menambahkan nilai a dan b.
2. Pengurangan ($a-b$)
 - Operasi: $20-3$
 - Hasil: 17
 - Analisis: Mengurangkan b dari a, menunjukkan sisa nilai setelah mengurangi.
3. Perkalian ($a*b$)
 - Operasi : $20*3$
 - Hasil: 60
 - Analisa: Mengkalikan nilai a dan b.
4. Pembagian (a/b)
 - Operasi : $20/3$
 - Hasil : 6
 - Analisa : Membagikan nilai dari a dan b untuk mendapatkan hasil
5. Modulus ($a\%b$)
 - Operasi : $20 \bmod 3$
 - Hasil : 2
 - Analisa : Membagikan bilangan 20 dengan 3 yang menghasilkan 6 dan sisa 2. Maka modulusnya adalah 2.

[1] Kesimpulan

Kode ini mengajarkan cara menggunakan operator aritmatika dasar dalam Bahasa pemrograman java. Dan meningkatkan pemahaman dan keterampilan dalam melakukan operasi aritmatika.

[No. 2] Identifikasi Masalah:

```
public class OperatorPenugasan {
    public static void main(String[] args) {
        // deklarasi nilai
        int a = 20, b = 3;
        //operator penugasan
        b += a;
        System.out.println("Penambahan : " + b);

        // pengurangan
        b -= a;
        System.out.println("Pengurangan : " + b);

        // perkalian
        b *= a;
        System.out.println("Perkalian : " + b);

        // Pembagian
        b /= a;
        System.out.println("Pembagian : " + b);
    }
}
```

```

// Sisa bagi
b %= a;
// sekarang b=0
System.out.println("Sisa Bagi: " + b);
}
}

```

[No. 2] Identifikasi Masalah:

1. Bandingkan hasil Contoh 1 dengan Contoh 2!

[No.2] Analisis dan Argumentasi

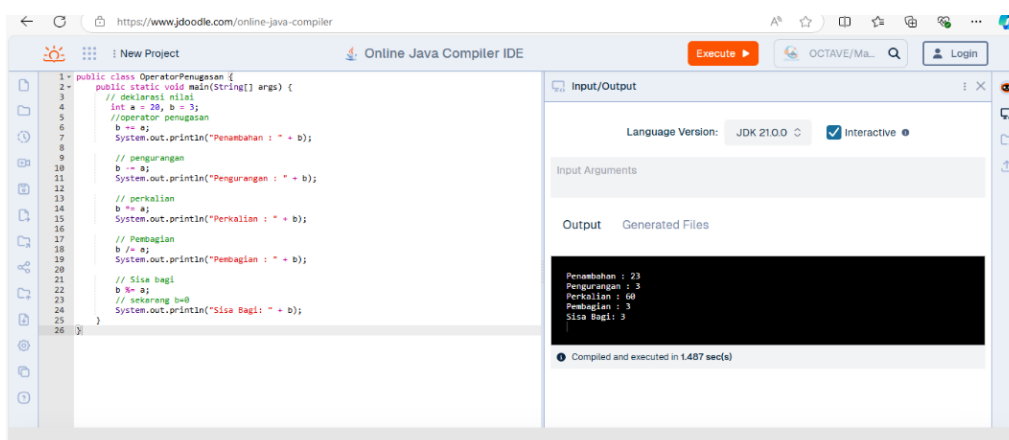
1. Saya mengusulkan permasalahan ini dapat diatasi dengan cara memperbaiki penggunaan kode program.
2. Alasan solusi ini karena dengan memahami perbedaan antara operator aritmatika dan operator penugasan.
3. Perbaiki kode program dengan cara meninjau output dari setiap operasi, sehingga setiap hasil menunjukkan penggunaan yang tepat dari operator yang digunakan.

[No.2] Penyusunan Algoritma dan Kode Program

1. Algoritma

- Mulai
- Deklarasikan dan inialisasi variable
- Tampilkan nilai
- Hitung
- Tampilkan dan selesai

2. Kode program dan luaran



- Analisa Luaran Yang dihasilkan
Kedua program memiliki konteks yang berbeda, contoh 1 kode aritmatika sedangkan contoh 2 kode penugasan yang memodifikasi variable.

- Tujuan
 - Program Aritmatika : menunjukkan bagaimana melakukan operasi aritmatika dasar secara langsung dengan nilai variable.
 - Program Penugasan : Menunjukkan penggunaan operator penugasan untuk melakukan operasi aritmatika dan menyimpan hasilnya ke dalam variable.
- Output
 - Program Aritmatika: Menampilkan hasil dari lima operasi aritmatika dasar, tetapi tidak mengubah nilai b.
 - Program Penugasan : Menampilkan hasil dari lima operasi aritmatika, tetapi setiap hasil bergantung pada nilai b yang telah diperbarui.

[No.2] Kesimpulan

Dalam kedua program, memiliki tujuan utama yaitu untuk memahami dan menerapkan operasi aritmatika. Algoritmanya untuk masing-masing program memberikan Langkah-langkah yang jelas untuk mencapai hasil.

[No. 3] Identifikasi Masalah:

```
public class OperatorRelasional {
    public static void main(String[] args) {
        int nilaiA = 12;
        int nilaiB = 4;
        boolean hasil;

        System.out.println(" A = " + nilaiA + "\n B = " + nilaiB);
        // apakah A lebih besar dari B?
        hasil = nilaiA > nilaiB;
        System.out.println("Hasil A > B = "+ hasil);

        // apakah A lebih kecil dari B?
        hasil = nilaiA < nilaiB;
        System.out.println("Hasil A < B = "+ hasil);

        // apakah A lebih besar samadengan B?
        hasil = nilaiA >= nilaiB;
        System.out.println("Hasil A >= B = "+ hasil);

        // apakah A lebih kecil samadengan B?
        hasil = nilaiA <= nilaiB;
        System.out.println("Hasil A <= B = "+ hasil);

        // apakah nilai A sama dengan B?
        hasil = nilaiA == nilaiB;
        System.out.println("Hasil A == B = "+ hasil);

        // apakah nilai A tidak samadengan B?
        hasil = nilaiA != nilaiB;
        System.out.println("Hasil A != B = "+ hasil);
    }
}
```

1. Ubahlah nilai A = 4 dan B = 4. Analisa perubahan yang terjadi!
2. Bandingkan bagaimana perbedaan nilai A dan B mempengaruhi nilai luaran!

[No.3] Analisis dan Argumentasi

- 4) Saya mengusulkan permasalahan ini dapat diatasi dengan cara mengenalkan operator relasional yang lebih luas.

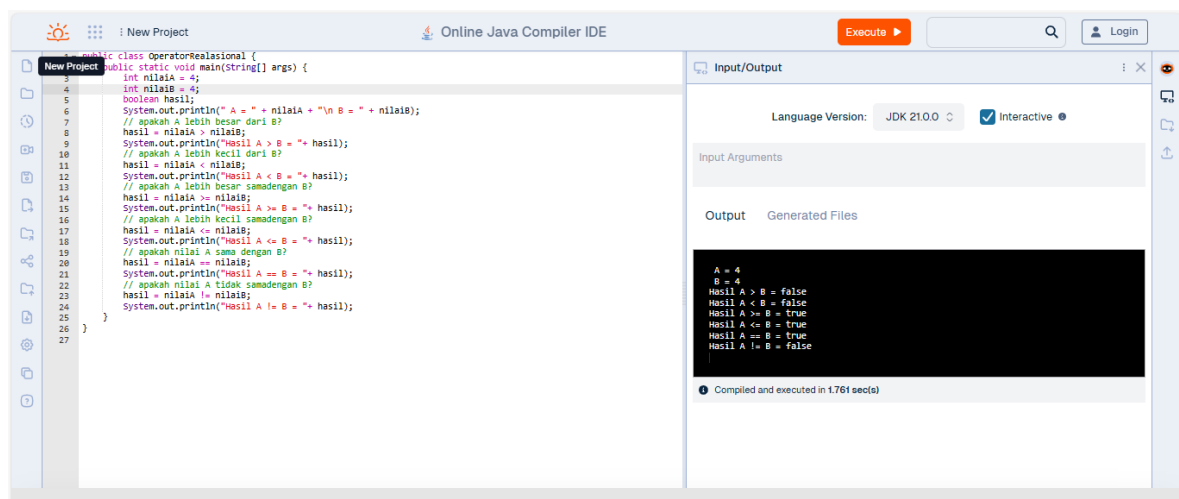
- 5) Alasan solusi ini karena operator relasional adalah bagian penting dalam pengambilan Keputusan kode.
- 6) Perbaiki kode program dengan cara menambahkan penjelasan atau komentar.

[No.3] Penyusunan Algoritma dan Kode Program

1. Algoritma

- a. Mulai
- b. Deklarasi Variabel
- c. Input nilai
- d. Tampilkan Nilai
- e. Lakukan Perbandingan
- f. Selesai

2. Kode program dan luaran



The screenshot shows an online Java IDE with a code editor on the left and an output window on the right. The code defines a class `OperatorRelasional` with a `main` method. It declares two integer variables, `A` and `B`, both set to 4. It then performs several relational comparisons: `A > B`, `A < B`, `A >= B`, `A <= B`, `A == B`, and `A != B`, printing the results. The output window shows the results of these comparisons: `A = 4`, `B = 4`, `Hasil A > B = false`, `Hasil A < B = false`, `Hasil A >= B = true`, `Hasil A <= B = true`, `Hasil A == B = true`, and `Hasil A != B = false`. The IDE also shows the language version as JDK 21.0.0 and the execution time as 1.761 seconds.

- Analisa luaran yang dihasilkan
Dengan mengubah nilai A dan B menjadi 4 hasil dari beberapa perbandingan berubah perbandingan yang sebelumnya menghasilkan true untuk nilai yang sebelumnya, sekarang menjadi false , dan perbandingan yang memiliki kesetaraan (`==`) dan relasi yang lebih besar atau sama dengan (`>=`) dan lebih kecil atau sama dengan (`<=`) menghasilkan true, menunjukkan bahwa kedua nilai tersebut sama. Perubahan nilai A dan B memberikan dampak yang signifikan dimana yang semulanya hasilnya false bisa berubah menjadi true atau sebaliknya yang awalnya true menjadi false.

[No.3] Kesimpulan

Pada kode program ini saya mengubah angka A menjadi 4 yang dimana angka tersebut sebanding dengan B. Yang mana mengubah output yang awalnya true menjadi false, dikarenakan kesamaan angka dalam kode.

[No. 4] Identifikasi Masalah

```
public class operator {  
    public static void main(String[] args) {  
        int a = 10;  
    }  
}
```

```

        System.out.println("# Post Increment #");
        System.out.println("=====");
        System.out.println("Isi variabel a: " + a);
        System.out.println("Isi variabel a: " + a++);
        System.out.println("Isi variabel a: " + a);

        System.out.println();

        int b = 10;
        System.out.println("# Pre Increment #");
        System.out.println("=====");
        System.out.println("Isi variabel b: " + b);
        System.out.println("Isi variabel b: " + ++b);
        System.out.println("Isi variabel b: " + b);

        System.out.println();

        int c = 10;
        System.out.println("# Post Decrement #");
        System.out.println("=====");
        System.out.println("Isi variabel c: " + c);
        System.out.println("Isi variabel c: " + c--);
        System.out.println("Isi variabel c: " + c);

        System.out.println();

        int d = 10;
        System.out.println("# Pre Decrement #");
        System.out.println("=====");
        System.out.println("Isi variabel d: " + d);
        System.out.println("Isi variabel d: " + --d);
        System.out.println("Isi variabel d: " + d);
    }
}

```

1. Berdasarkan luaran program Contoh 4, bandingkan hasil Post dan Pre untuk Increment dan Decrement!

[No.4] Analisis dan Argumentasi

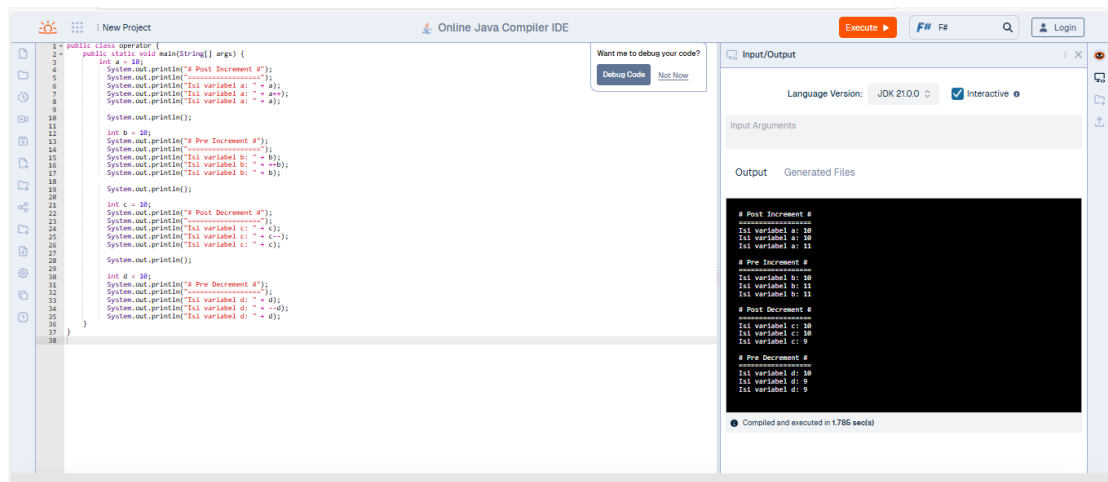
- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menggunakan pemahaman yang lebih baik tentang perbedaan antara post dan pre increment atau decrement.
- 2) Alasan solusi ini karena kesalahan dalam penggunaan kode operator ini dapat menyebabkan hasil yang berbeda, yang berdampak pada fungsi program.
- 3) Perbaiki kode program dengan cara menyertakan komentar yang jelas.

[No.4] Penyusunan Algoritma dan Kode Program

1. Algoritma

- a. Mulai
- b. Inisialisasi Variabel
- c. Post increment
- d. Pre Decrement
- e. Post Decrement
- f. Pre Increment
- g. Selesai

2. Kode program dan luaran



The screenshot shows an online Java compiler interface. On the left, the code editor contains a Java program that demonstrates the use of pre-increment, post-increment, pre-decrement, and post-decrement operators. The code includes comments and output statements for each operation. On the right, the 'Input/Output' panel shows the results of the program execution, which are grouped into sections for each operator type. The output shows the state of variables before and after each operation.

```
1 public class operator {
2     public static void main(String[] args) {
3         int a = 10;
4         System.out.println("Pre Increment #");
5         System.out.println("=====");
6         System.out.println("Isi variabel a: " + a);
7         System.out.println("Isi variabel a: " + ++a);
8         System.out.println("Isi variabel a: " + a);
9         System.out.println();
10
11         int b = 10;
12         System.out.println("Pre Increment #");
13         System.out.println("=====");
14         System.out.println("Isi variabel b: " + b);
15         System.out.println("Isi variabel b: " + ++b);
16         System.out.println("Isi variabel b: " + b);
17         System.out.println();
18
19         int c = 10;
20         System.out.println("Post Increment #");
21         System.out.println("=====");
22         System.out.println("Isi variabel c: " + c);
23         System.out.println("Isi variabel c: " + c++);
24         System.out.println("Isi variabel c: " + c);
25         System.out.println();
26
27         int d = 10;
28         System.out.println("Post Decrement #");
29         System.out.println("=====");
30         System.out.println("Isi variabel d: " + d);
31         System.out.println("Isi variabel d: " + d--);
32         System.out.println("Isi variabel d: " + d);
33     }
34 }
```

Output:

```
# Pre Increment #
=====
Isi variabel a: 10
Isi variabel a: 11
Isi variabel a: 11

# Post Increment #
=====
Isi variabel b: 10
Isi variabel b: 11
Isi variabel b: 11

# Pre Decrement #
=====
Isi variabel c: 10
Isi variabel c: 9
Isi variabel c: 9

# Post Decrement #
=====
Isi variabel d: 10
Isi variabel d: 9
Isi variabel d: 9
```

Compiled and executed in 1.786 secs

- Analisa Luaran Yang dihasilkan program di atas menunjukkan penggunaan operator increment (++) dan decrement (--) dalam Java, baik dalam bentuk post (setelah) maupun pre (sebelum). Memiliki Perbandingan antara keduanya yaitu, post nilai variable lebih dulu baru diincrement atau decrement dan hasil pertama dan kedua sama, sedangkan hasil ketiga menunjukkan nilai setelah perubahan. Dan sedangkan Pre nilai variabelnya lebih dulu sebelum digunakan dan hasil kedua langsung mencerminkan perubahan.

[No.1] Kesimpulan

Kode ini membantu kita untuk memahami cara kerja operator increment dan decrement dalam java, untuk menghindari kesalahan dalam pemrograman. Hasil dari penggunaan kode ini dapat mempengaruhi alur eksekusi program. Kesalahan dalam operator berdampak pada Keputusan yang diambil.

[No. 5] Identifikasi Masalah

```
public class OperatorLogika {
    public static void main (String [] args) {
        boolean a = true;
        boolean b = false;
        boolean c;
        c = a && b;
        System.out.println("true && false = " +c);
    }
}
```

- 1.Tambahkan baris kode untuk memeriksa a || b.
- 2.Ubahlah nilai a = false dan b = false. Analisa perubahan dan perbedaan Boolean yang terjadi!
- 3.Apabila diketahui pernyataan a || b && a || !b. Uraikan urutan logika yang akan dikerjakan! Analisa luaran true atau false dari pernyataan tersebut!

[No.5] Analisis dan Argumentasi

- 1.Saya mengusulkan permasalahan ini dapat diatasi dengan cara memberikan penjelasan yang jelas tentang penggunaan operator logika.
- 2.Alasan solusi ini karena pemahaman yang mendalam mengenai operator logika akan membantu kita menghindari kesalahan logika dalam kode.

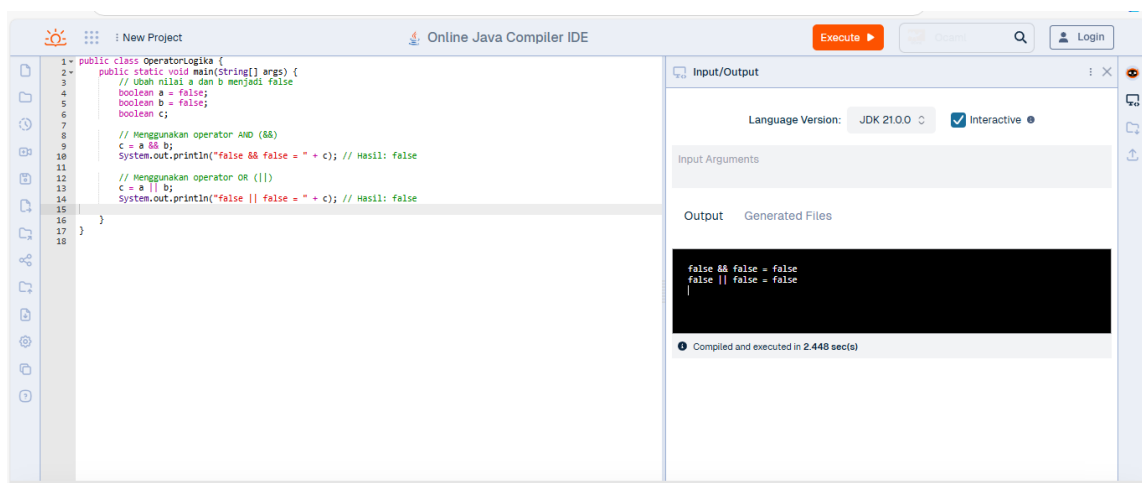
3. Perbaiki kode program dengan cara menambahkan komentar yang menjelaskan setiap operasi logika dan hasilnya.

[No.5] Penyusunan Algoritma dan Kode Program

1. Algoritma

- Mulai
- Inisialisasi kode
- Evaluasi
- Selesai

2. Kode program dan luaran



The screenshot shows an online Java compiler interface. On the left, the code editor contains a Java class named `OperatorLogika` with a `main` method. The code initializes two boolean variables, `a` and `b`, to `false`. It then uses the `&&` (AND) and `||` (OR) operators to evaluate expressions involving `a` and `b`, printing the results. Comments in Indonesian explain the operations. On the right, the 'Input/Output' panel shows the output of the program, which displays the results of the logical operations: `false && false = false` and `false || false = false`. Below the output, it states 'Compiled and executed in 2.448 secs'.

```
1 public class OperatorLogika {
2     public static void main(String[] args) {
3         // Ubah nilai a dan b menjadi false
4         boolean a = false;
5         boolean b = false;
6         boolean c;
7
8         // Menggunakan operator AND (&&)
9         c = a && b;
10        System.out.println("false && false = " + c); // Hasil: false
11
12        // Menggunakan operator OR (||)
13        c = a || b;
14        System.out.println("false || false = " + c); // Hasil: false
15    }
16 }
17
18 }
```

Output:

```
false && false = false
false || false = false
```

Compiled and executed in 2.448 secs

- Analisa luaran yang dihasilkan
Kode ini memberikan gambaran jelas tentang bagaimana operator logika AND dan OR berfungsi dalam situasi ketika kedua operand bernilai `false`. Negasi (!) memiliki prioritas lebih tinggi dan dievaluasi sebelum operator AND dan OR. Dengan nilai `a = false` dan `b = false`, pernyataan `a || b` dan `a || !b` dievaluasi menjadi `true` setelah mengikuti urutan logika dan prioritas operator.

[No.5] Kesimpulan

Pada program diatas, saya mendapatkan pengetahuan baru yaitu bahwa ! (negasi) lebih tinggi dibandingkan && (AND) dan || (OR).

[No. 6] Identifikasi Masalah:

```
public class OperatorKondisi{
    public static void main( String[] args ){
        String status = "";
        int nilai = 80;
        status = (nilai > 60)?"Lulus":"Gagal";
        System.out.println( status );
    } }
```

1. Berdasarkan Contoh 6, ubahlah nilai = 60. Analisis hasil dan proses yang terjadi!

[No.6] Analisis dan Argumentasi

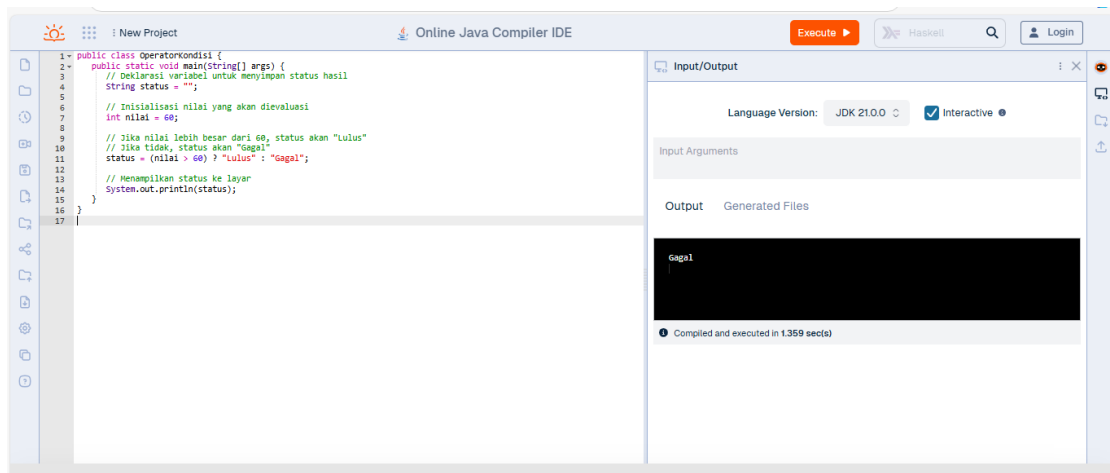
1. Saya mengusulkan permasalahan ini dapat diatasi dengan cara mengubah kondisi dalam operator untuk menghitung nilai ambang batas.
2. Alasan solusi ini karena memperhitungkan nilai ambang batas untuk mengubah status nilai > menjadi lulus.
3. Perbaiki kode program dengan cara mengubah ekspresi pada operator.

[No.6] Penyusunan Algoritma dan Kode Program

1. Algoritma

- a. Mulai
- b. Deklarasikan variable
- c. Evaluasi kondisi
- d. Tampilkan hasil
- e. Selesai

2. kode program dan luaran



- Analisa luaran yang dihasilkan
Dengan mengubah nilai menjadi 60 maka hasil dari program menunjukkan nilai ambang batas, maka hasil evaluasi status adalah "gagal".

[No.6] Kesimpulan

Pada program ini saya menggunakan operator ternary yang dimana membuat kode lebih ringkas dan mudah dibaca untuk kasus sederhana.

[No. 7] Identifikasi Masalah:

```
public class operator {
    public static void main(String[] args) {
        int a = 10;
        int b = 7;
        int hasil;

        hasil = a & b;
        System.out.println("Hasil dari a & b : " + hasil );

        hasil = a | b;
```

```

        System.out.println("Hasil dari a | b : " + hasil );
    }

    hasil = a ^ b;
    System.out.println("Hasil dari a ^ b : " + hasil );

    hasil = ~a;
    System.out.println("Hasil dari ~a : " + hasil );

    hasil = a >> 1;
    System.out.println("Hasil dari a >> 1 : " + hasil );

    hasil = b << 2;
    System.out.println("Hasil dari b << 2 : " + hasil );
}
}

```

1. Pilihlah 3 perhitungan Contoh 7, kemudian uraikan perhitungan biner! Simpulkan hasilnya!

[No.7] Analisis dan Argumentasi

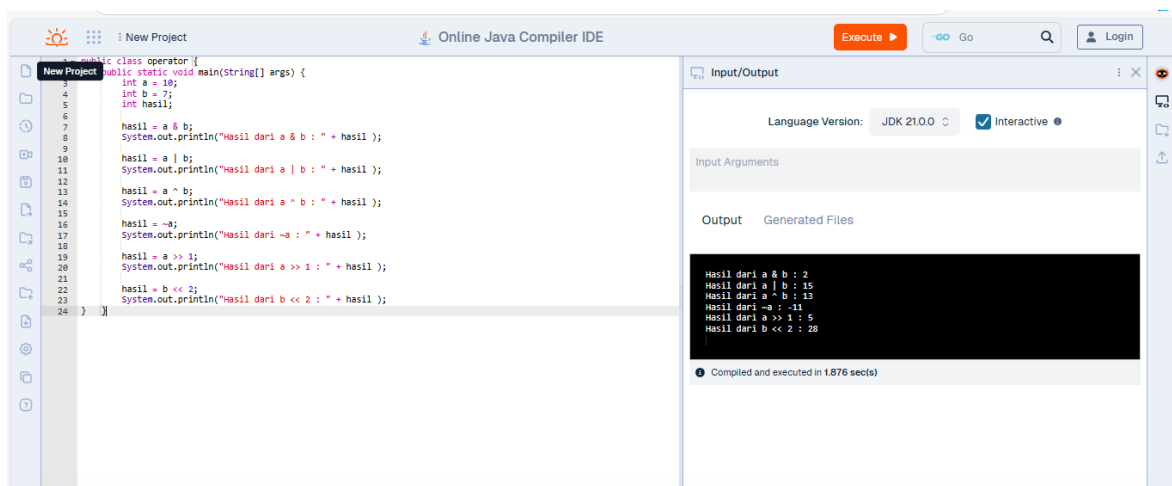
1. Saya mengusulkan permasalahan ini dapat diatasi dengan cara menggunakan operator bitwise yang lebih efisien.
2. Alasan solusi ini karena operator bitwise memberikan kecepatan dan efisiensi yang lebih tinggi dalam pengolahan data
3. Perbaiki kode program dengan cara menambahkan lebih banyak komentar untuk menjelaskan setiap operasi.

[No.7] Penyusunan Algoritma dan Kode Program

1. Algoritma

- a. Mulai
- b. Inisialisasi variable
- c. Operasi
- d. Akhiri program

2.kode program dan luaran



- Inisialisasi

Berikut merupakan 3 kode program

1.Hasil: 2 (dalam biner: 0010)

Penjelasan: Hanya bit yang sama-sama 1 yang menghasilkan 1.

10: 1010

7: 0111

2. Hasil: 15 (dalam biner: 1111)

Penjelasan: Bit yang salah satu atau kedua bernilai 1 menghasilkan 1.

10: 1010

7: 0111

15: 1111

3. Hasil: 13 (dalam biner: 1101)

Penjelasan: Bit yang berbeda menghasilkan 1.

10: 1010

7: 0111

[No.7] Kesimpulan

Dengan menggunakan kode ini membuat kita memahami bagaimana operator bitwise bekerja.