

Nama & NPM	Topik:	Tanggal:
Dewangga Nugroho Anwar G1F024045	Operator	10 September 2024

[No. 1] Identifikasi Masalah:

- 1) Uraikan permasalahan dan variabel

program tersebut hanya menampilkan hasil penjumlahan ( $a + b$ ). Dan di soal memerintahkan untuk menampilkan semua operator aritmatika, seperti pengurangan, perkalian, pembagian, dan modulus, maka operasi tersebut belum ada dalam kode ini.

[No.1] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menambahkan operasi aritmatika lainnya (seperti pengurangan, perkalian, pembagian, dan modulus).
- 2) Alasan solusi ini karena program hanya melakukan operasi penjumlahan, meskipun ada lebih banyak operator aritmatika yang dapat dieksplorasi. Dengan menambahkan operasi lain, program ini akan lebih bervariasi dalam menunjukkan berbagai jenis operator aritmatika dasar dalam pemrograman.

[No.1 ] Penyusunan Algoritma dan Kode Program

- 1) Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.

- (a) Deklarasi variabel.
- (b) Tampilkan nilai awal.
- (c) Lakukan operasi aritmatika.
- (d) Cetak hasil operasi.
- (e) Akhir program.

- 2) Kode program dan luaran

```

1 //deklarasi variabel
2 public class Opeator{
3     public static void main(String[] args) {
4         //tampilkan nilai awal
5         int a = 2, b = 3, c = 6, d = 4;
6         //lakukan operasi aritmatika
7         System.out.println("a: " +a);
8         System.out.println("b: " +b);
9         System.out.println("c: " +a);
10        System.out.println("d: " +a);
11        //cetak hasil operasi
12        System.out.println("a*b+c/a-d = " + (a*b+c/a-d));
13        //akhir program
14    }}

```

Gambar Input kode 1.0

- a) Screenshot/ Capture potongan kode dan hasil luaran.

```
a: 2
b: 3
c: 2
d: 2
a*b+c/a-d = 5
```

Gambar Output kode 1.1

b) Analisa luaran yang dihasilkan.

Luaran sudah sesuai dengan program yang disusun.

Tipe data yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data.

#### [No.1] Kesimpulan

1) Analisa.

a) Program pada soal hanya melakukan satu operasi aritmatika (penjumlahan) pada dua variabel a dan b. Karena judul program adalah OperatorAritmatika, idealnya program ini juga harus menunjukkan berbagai operasi aritmatika lainnya seperti pengurangan, perkalian, pembagian, dan modulus. Algoritma yang digunakan sangat sederhana, yaitu hanya melakukan penjumlahan dua angka dan menampilkannya. Kode program sudah benar, namun terbatas pada satu operasi aritmatika.

b) Alasan dalam pengambilan Keputusan tersebut adalah, karena melihat terdapat perintah ' $(2*3 + 6 / 2 - 4)$ ' pada soal, hasil tersebut lah yang menjadi alasan saya untuk menambahkan berbagai operasi aritmatika lainnya.

[No. 2] Identifikasi Masalah:

- 1) Uraikan permasalahan dan variable.

Kode pemrograman tersebut menggunakan operator penugasan untuk menambah nilai variabel b dengan nilai variabel a melalui operator +=. Tidak ada masalah signifikan dalam kode ini, namun karena judul program adalah OperatorPenugasan, dan juga pada soal memerintahkan menambahkan lebih banyak operator penugasan seperti -=, \*=, /=, dan %= untuk memberikan contoh yang lebih lengkap.

[No.2] Analisis dan Argumentasi.

- 1.) Pada program mendeklarasikan dua variabel a dan b dengan tipe data int. Nilai a adalah 20 dan b adalah 3.
- 2.) Penggunaan operator penugasan += dalam kasus ini adalah tepat, karena memungkinkan penulisan yang lebih singkat dan lebih efisien.

[No.2 ] Penyusunan Algoritma dan Kode Program.

- 1) Algoritma.

Algoritma adalah langkah-langkah penyelesaian masalah.

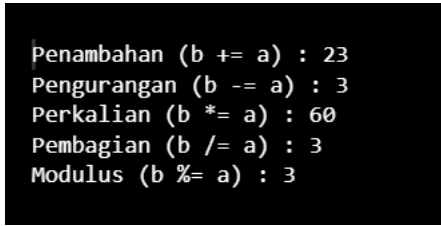
- a) Mulai Program.
- b) Deklarasi Variabel.
- c) Operasi Penugasan.
- d) Akhir Program.

- 2) Kode program dan luaran.

```
1 public class OperatorPenugasan {
2     public static void main(String[] args) {
3         // 1. Deklarasi nilai variabel a dan b
4         int a = 20, b = 3;
5
6         // 2. Melakukan operasi penugasan penjumlahan (b += a)
7         // Sama dengan: b = b + a
8         b += a;
9         // Menampilkan hasil penjumlahan
10        System.out.println("Penambahan (b += a) : " + b);
11
12        // 3. Melakukan operasi penugasan pengurangan (b -= a)
13        // Sama dengan: b = b - a
14        b -= a;
15        // Menampilkan hasil pengurangan
16        System.out.println("Pengurangan (b -= a) : " + b);
17
18        // 4. Melakukan operasi penugasan perkalian (b *= a)
19        // Sama dengan: b = b * a
20        b *= a;
21        // Menampilkan hasil perkalian
22        System.out.println("Perkalian (b *= a) : " + b);
23
24        // 5. Melakukan operasi penugasan pembagian (b /= a)
25        // Sama dengan: b = b / a
26        b /= a;
27        // Menampilkan hasil pembagian
28        System.out.println("Pembagian (b /= a) : " + b);
29
30        // 6. Melakukan operasi penugasan modulus (b %= a)
31        // Sama dengan: b = b % a
32        b %= a;
33        // Menampilkan hasil modulus (sisanya)
34        System.out.println("Modulus (b %= a) : " + b);
35    }
36 }
```

Gambar Input kode 2.0

a) Screenshot/ Capture potongan kode dan hasil luaran



```
Penambahan (b += a) : 23  
Pengurangan (b -= a) : 3  
Perkalian (b *= a) : 60  
Pembagian (b /= a) : 3  
Modulus (b %= a) : 3
```

Gambar Output kode 2.1

b) Analisa luaran yang dihasilkan.

Luaran sudah sesuai dengan program yang disusun.

Tipe data yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data.

## [No.2] Kesimpulan

### 1) Analisa

- a. Kode program tersebut menunjukkan penggunaan berbagai operator penugasan pada dua variabel `a` dan `b`. Variabel `a` diberi nilai 20 dan `b` diberi nilai 3. Program kemudian menggunakan operator penugasan seperti `+=`, `-=`, `\*=`, `/=`, dan `%=` untuk memodifikasi nilai `b` dengan operasi aritmatika penjumlahan, pengurangan, perkalian, pembagian, dan modulus, yang hasilnya ditampilkan setelah setiap operasi. Setiap operator melakukan operasi aritmatika sesuai dengan nilainya dan memperbarui variabel `b` langsung tanpa memerlukan penulisan ekspresi yang lebih panjang. Program ini efektif dalam menunjukkan cara kerja operator penugasan dalam Java dengan cara yang ringkas dan mudah dipahami.

[No. 3] Identifikasi Masalah:

1) Uraikan permasalahan dan variabel

Program ini melakukan beberapa perbandingan menggunakan operator relasional dan mencetak hasilnya. Operator pertama yang digunakan adalah > untuk mengecek apakah nilaiA lebih besar dari nilaiB, yang menghasilkan nilai true atau false tergantung pada perbandingan. Operator < digunakan untuk memeriksa apakah nilaiA lebih kecil dari nilaiB, sedangkan >= dan <= digunakan untuk memeriksa apakah nilaiA lebih besar atau sama dengan nilaiB, dan apakah nilaiA lebih kecil atau sama dengan nilaiB, masing-masing. Selain itu, operator == memeriksa apakah nilaiA sama dengan nilaiB, dan operator != memeriksa apakah nilaiA tidak sama dengan nilaiB. Hasil dari setiap perbandingan ini dicetak ke layar menggunakan System.out.println(), memberikan informasi apakah kondisi yang diperiksa benar atau salah.

[No.3] Analisis dan Argumentasi

- 1) Program ini secara benar menggunakan operator relasional untuk membandingkan dua nilai integer. Hasil dari setiap perbandingan disimpan dalam variabel hasil dan dicetak ke layar. Kode ini merupakan cara yang baik untuk mempelajari dan memahami bagaimana operator relasional bekerja dalam bahasa pemrograman Java.

[No.2] Penyusunan Algoritma dan Kode Program

1) Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.

- (a) Mulai
- (b) Inisialisasi
- (c) Tampilkan nilai awal
- (d) Perbandingan
- (e) Tutup

2) Kode program dan luaran

```

1 public class Operator {
2
3     public static void main(String[] args) {
4         int nilaiA = 10;
5         int nilaiB = 2;
6         boolean hasil;
7         System.out.println(" A = " + nilaiA + "\n B = " + nilaiB);
8         // apakah A lebih besar dari B?
9         hasil = nilaiA > nilaiB;
10        System.out.println("Hasil A > B = " + hasil);
11        // apakah A lebih kecil dari B?
12        hasil = nilaiB < nilaiA;
13        System.out.println("Hasil B < A = " + hasil);
14        // apakah A lebih besar samadengan B?
15        hasil = nilaiA >= nilaiB;
16        System.out.println("Hasil A >= B = " + hasil);
17        // apakah A lebih kecil samadengan B?
18        hasil = nilaiB <= nilaiA;
19        System.out.println("Hasil B <= A = " + hasil);
20        // apakah nilai A sama dengan B?
21        hasil = (nilaiA - nilaiB) == 8;
22        System.out.println("Hasil A == B = " + hasil);
23        // apakah nilai A tidak samadengan B?
24        hasil = nilaiA != nilaiB;
25        System.out.println("Hasil A != B = " + hasil);
26
27    }
28
29 }

```

Gambar Input kode 3.0

- a) Screenshot/ Capture potongan kode dan hasil luaran.

```

A = 10
B = 2
Hasil A > B = true
Hasil B < A = true
Hasil A >= B = true
Hasil B <= A = true
Hasil A == B = true
Hasil A != B = true

```

Gambar Output kode 3.1

- b) Analisa luaran yang dihasilkan.

Luaran sudah sesuai dengan program yang disusun.

Tipe data yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data.

### [No.3] Kesimpulan

- 1) Analisa.

- a) Kode Java ini mendemonstrasikan penggunaan operator relasional untuk membandingkan dua nilai integer, nilaiA dan nilaiB, yang diinisialisasi dengan nilai 12 dan 4, masing-masing. Pertama-tama, kode ini mencetak nilai dari variabel-variabel tersebut. Kemudian, kode ini mengevaluasi dan mencetak hasil dari berbagai operasi relasional. Kode pertama memeriksa apakah nilaiA lebih besar dari nilaiB ( $12 > 4$ ), yang hasilnya adalah true, sehingga outputnya adalah true. Selanjutnya, kode memeriksa apakah nilaiA lebih kecil dari nilaiB ( $12 < 4$ ), yang hasilnya adalah false, sehingga outputnya adalah false. Kode berikutnya mengevaluasi apakah nilaiA lebih

besar atau sama dengan nilaiB ( $12 \geq 4$ ), yang hasilnya adalah true, sehingga outputnya adalah true. Kode kemudian memeriksa apakah nilaiA lebih kecil atau sama dengan nilaiB ( $12 \leq 4$ ), yang hasilnya adalah false, sehingga outputnya adalah false. Terakhir, kode memeriksa apakah nilaiA sama dengan nilaiB ( $12 == 4$ ), yang hasilnya adalah false, sehingga outputnya adalah false, dan memeriksa apakah nilaiA tidak sama dengan nilaiB ( $12 != 4$ ), yang hasilnya adalah true, sehingga outputnya adalah true. Setiap hasil dicetak dengan label yang sesuai, memberikan demonstrasi yang jelas tentang cara kerja operator relasional di Java.

[No. 4] Identifikasi Masalah:

1) Uraikan permasalahan dan variabel

Permasalahan nya jika tujuan nya adalah melihat nilai a setelah increment, maka baris kedua `System.out.println` tidak mencerminkan perubahan ini. Untuk melihat nilai setelah increment, Anda harus menggunakan `++a` (pre-increment) atau mencetak nilai a setelah operasi increment.

[No.4] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menambahkan operator `a++` adalah operator post-increment. Operator ini meningkatkan nilai a setelah nilai a yang lama digunakan dalam ekspresi. Jadi, `a++` menggunakan nilai 5 dalam pernyataan `System.out.println`, dan kemudian meningkatkan nilai a menjadi 6.
- 2) Alasan solusi ini karena program Operator `a++` adalah contoh dari operator post-increment yang meningkatkan nilai variabel setelah ekspresi dievaluasi. Jika Anda ingin melihat hasil setelah peningkatan dalam output, Anda bisa menggunakan operator pre-increment (`++a`) yang meningkatkan nilai terlebih dahulu sebelum digunakan dalam ekspresi.

[No.4] Penyusunan Algoritma dan Kode Program

1) Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.

- (a) Inisialisasi.
- (b) Tampilan nilai awal.
- (c) Operasi post-increment.
- (d) Tutup.



## 2) Kode program dan luaran

```
1- public class operator {
2-     public static void main(String[] args) {
3         // Deklarasi variabel a dengan nilai awal 10
4         int a = 10;
5         System.out.println("# Post Increment #");
6         System.out.println("_____");
7
8         // Menampilkan nilai awal variabel a
9         System.out.println("Isi variabel a: " + a);
10
11        // Menampilkan nilai a sebelum dilakukan post-increment (nilai ditampilkan dulu, lalu increment)
12        System.out.println("Isi variabel a: " + a++);
13
14        // Menampilkan nilai a setelah dilakukan post-increment (sekarang a bernilai 11)
15        System.out.println("Isi variabel a: " + a);
16
17        System.out.println(); // Baris kosong sebagai pemisah
18
19        // Deklarasi variabel b dengan nilai awal 10
20        int b = 10;
21        System.out.println("# Pre Increment #");
22        System.out.println("_____");
23
24        // Menampilkan nilai awal variabel b
25        System.out.println("Isi variabel b: " + b);
26
27        // Melakukan pre-increment (nilai b di-increment dulu, kemudian ditampilkan)
28        System.out.println("Isi variabel b: " + ++b);
29
30        // Menampilkan nilai b setelah increment (nilai b sekarang adalah 11)
31        System.out.println("Isi variabel b: " + b);
32
33        System.out.println(); // Baris kosong sebagai pemisah
34
35        // Deklarasi variabel c dengan nilai awal 10
36        int c = 10;
37        System.out.println("# Post Decrement #");
38        System.out.println("_____");
39
40        // Menampilkan nilai awal variabel c
41        System.out.println("Isi variabel c: " + c);
42
43        // Menampilkan nilai c sebelum dilakukan post-decrement (nilai ditampilkan dulu, lalu decrement)
44        System.out.println("Isi variabel c: " + c--);
45
46        // Menampilkan nilai c setelah dilakukan post-decrement (sekarang c bernilai 9)
47        System.out.println("Isi variabel c: " + c);
48
49        System.out.println(); // Baris kosong sebagai pemisah
50
51        // Deklarasi variabel d dengan nilai awal 10
52        int d = 10;
53        System.out.println("# Pre Decrement #");
54        System.out.println("_____");
55
56        // Menampilkan nilai awal variabel d
57        System.out.println("Isi variabel d: " + d);
58
59        // Melakukan pre-decrement (nilai d di-decrement dulu, kemudian ditampilkan)
60        System.out.println("Isi variabel d: " + --d);
61
62        // Menampilkan nilai d setelah decrement (nilai d sekarang adalah 9)
63        System.out.println("Isi variabel d: " + d);
64    }
65 }
```

Gambar Input kode 4.0

### a) Screenshot/ Capture potongan kode dan hasil luaran.

```
# Post Increment #
Isi variabel a: 10
Isi variabel a: 10
Isi variabel a: 11

# Pre Increment #
Isi variabel b: 10
Isi variabel b: 11
Isi variabel b: 11

# Post Decrement #
Isi variabel c: 10
Isi variabel c: 10
Isi variabel c: 9

# Pre Decrement #
Isi variabel d: 10
Isi variabel d: 9
Isi variabel d: 9
```

Gambar Output kode 4.1

### b) Analisa luaran yang dihasilkan.

Luaran sudah sesuai dengan program yang disusun.

Tipe data yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data.

#### [No.4] Kesimpulan

##### 1) Analisa.

- a) Program ini mendemonstrasikan penggunaan operator increment (++) dan decrement (--) dalam bentuk post-increment, pre-increment, post-decrement, dan pre-decrement. Dengan menggunakan empat variabel (a, b, c, dan d), program ini menunjukkan perbedaan cara kerja operator-operator tersebut, yaitu bagaimana nilai variabel berubah ketika operator ditempatkan sebelum atau setelah variabel.

[No. 5] Identifikasi Masalah:

1) Uraikan permasalahan dan variabel

Program ini menggunakan operator logika AND (&&) untuk mengevaluasi dua variabel boolean, yaitu a yang bernilai true dan b yang bernilai false. Operator AND akan menghasilkan nilai true hanya jika kedua operand bernilai true. Dalam kasus ini, karena b bernilai false, hasil dari operasi a && b adalah false. Program ini kemudian mencetak hasil dari operasi logika tersebut dengan menampilkan pesan "Hasil logika (a && b) : false". Permasalahan yang diperiksa oleh program adalah bagaimana operator AND bekerja ketika salah satu nilai boolean adalah false, yang membuat keseluruhan ekspresi juga bernilai false.

[No.5] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara penggunaan operator logika AND (&&) dengan dua variabel boolean, a yang bernilai true dan b yang bernilai false. Operator logika AND akan menghasilkan true hanya jika kedua operand bernilai true; jika salah satu atau keduanya bernilai false, hasilnya adalah false. Dalam program ini, karena nilai a adalah true dan nilai b adalah false, hasil dari operasi a && b adalah false. Program kemudian mencetak hasil operasi logika tersebut dengan output: "Hasil logika (a && b) : false", yang sesuai dengan aturan logika AND.

[No.5] Penyusunan Algoritma dan Kode Program

1) Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.

- a) Inisialisasi.
- b) Post-Increment variabel a
- c) Post-Increment variabel b
- d) Post-Decrement variabel c
- e) Post-Decrement variable d
- f) Tutup.

2) Kode program dan luaran

```

1 public class OperatorLogika {
2     public static void main (String [] args) {
3         // Deklarasi variabel boolean a dengan nilai true
4         boolean a = true;
5
6         // Deklarasi variabel boolean b dengan nilai true
7         boolean b = true;
8
9         // Deklarasi variabel boolean c
10        boolean c;
11
12        // Operasi AND (&&) antara variabel a dan b
13        c = a && b;
14
15        // Menampilkan hasil operasi AND (a && b)
16        System.out.println("true && true = " + c);
17
18        // Menampilkan hasil operasi OR (a || b) namun masih menggunakan nilai c
19        System.out.println("true || true = " + c);
20    }
21 }

```

Gambar Input kode 5.0

- a) Screenshot/ Capture potongan kode dan hasil luaran.

```

true && false = true
true || false = true

```

Gambar Output kode 5.1

- b) Analisa luaran yang dihasilkan.

Luaran sudah sesuai dengan program yang disusun.

Tipe data yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data.

## [No.5] Kesimpulan

- 1) Analisa.

Program ini mendemonstrasikan penggunaan operator logika pada variabel bertipe boolean. Kedua variabel a dan b diinisialisasi dengan nilai true. Program melakukan operasi logika AND (&&) antara a dan b, yang menghasilkan true karena kedua operand bernilai true, dan hasil ini disimpan dalam variabel c. Namun, saat menampilkan hasil operasi OR (||), program secara tidak sengaja mencetak nilai dari variabel c, yang berisi hasil operasi AND. Seharusnya, untuk menampilkan hasil operasi OR dengan benar, program harus menghitung a || b dan mencetak hasilnya. Sebagai hasil, output saat ini salah karena tidak mencerminkan operasi OR yang sebenarnya.

[No. 6] Identifikasi Masalah:

1) Uraikan permasalahan dan variabel

Program Java yang diberikan menggunakan operator kondisional (ternary) untuk menentukan status berdasarkan nilai. Kode ini mendeklarasikan variabel status sebagai string kosong dan variabel nilai diinisialisasi dengan nilai 80. Operator kondisional  $(\text{nilai} > 60) ? \text{"Lulus"} : \text{"Gagal"}$  memeriksa apakah nilai lebih besar dari 60. Karena 80 lebih besar dari 60, hasil dari operator kondisional adalah "Lulus", yang kemudian disimpan dalam variabel status. Program kemudian mencetak nilai dari status, yang hasilnya adalah "Lulus". Secara keseluruhan, penggunaan operator kondisional dalam kode ini sudah benar, dan tidak ada masalah dalam logika atau sintaksis kode. Kode ini berfungsi dengan baik untuk menentukan dan menampilkan status berdasarkan nilai yang diberikan.

[No.6] Analisis dan Argumentasi

- 1) Program ini menggunakan operator ternary untuk menentukan status kelulusan berdasarkan nilai yang diberikan. Variabel `nilai` diinisialisasi dengan `80`, dan menggunakan operator ternary  $(\text{nilai} > 60) ? \text{"Lulus"} : \text{"Gagal"}$ , status akan ditetapkan sebagai `"Lulus"` jika nilai lebih besar dari `60`, dan `"Gagal"` jika sebaliknya. Karena nilai yang diberikan adalah `80`, yang lebih besar dari `60`, status yang dicetak adalah `"Lulus"`. Operator ternary secara efektif menggantikan pernyataan if-else sederhana dan membuat kode lebih ringkas dan mudah dibaca. Program ini dengan jelas menyelesaikan tugasnya sesuai dengan logika yang diinginkan, dan tidak memerlukan penambahan operasi aritmatika tambahan atau perubahan untuk mencapai tujuannya.

[No.6 ] Penyusunan Algoritma dan Kode Program

1) Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.

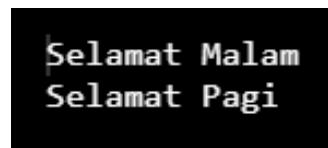
- (a) Deklarasi dan Inisialisasi
- (b) Operator Ternary
- (c) Output
- (d) Tutup

2) Kode program dan luaran.

```
1 public class OperatorKondisi {
2     public static void main(String[] args) {
3         // Deklarasi variabel status dengan tipe data String dan nilai awal kosong
4         String status = "";
5         // Deklarasi variabel jam dengan nilai 12
6         int jam = 12;
7         // Menggunakan operator ternary untuk mengecek kondisi apakah jam kurang dari 12
8         status = (jam < 12) ? "Selamat Pagi" : "Selamat Malam";
9         // Mencetak nilai dari variabel status
10        System.out.println(status);
11
12        // Deklarasi variabel kedua dengan tipe data String dan nilai awal kosong
13        String kedua = "";
14        // Deklarasi variabel Jam dengan nilai 12
15        int Jam = 12;
16        // Menggunakan operator ternary untuk mengecek kondisi apakah Jam lebih dari 12
17        kedua = (Jam > 12) ? "Selamat Malam" : "Selamat Pagi";
18        // Mencetak nilai dari variabel kedua
19        System.out.println(kedua);
20    }
21 }
```

Gambar Input kode 6.0

a) Screenshot/ Capture potongan kode dan hasil luaran.



```
Selamat Malam
Selamat Pagi
```

Gambar Output kode 6.1

b) Analisa luaran yang dihasilkan.

Luaran sudah sesuai dengan program yang disusun.

Tipe data yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data.

[No.6] Kesimpulan

1) Analisa.

- a) Kesimpulannya, program ini menggunakan operator ternary untuk menentukan pesan berdasarkan nilai waktu. Dengan variabel `jam` bernilai 12, program memeriksa apakah waktu kurang dari atau lebih besar dari 12 untuk mengatur pesan "Selamat Pagi" atau "Selamat Malam". Karena kedua variabel waktu (`jam` dan `Jam`) memiliki nilai 12, output dari program ini akan selalu menghasilkan "Selamat Malam" untuk variabel `status` dan "Selamat Pagi" untuk variabel `kedua`, sesuai dengan logika yang diterapkan pada masing-masing operator ternary. Program ini menggambarkan penggunaan dasar operator ternary dalam pengambilan keputusan sederhana di Java.

[No. 7] Identifikasi Masalah:

1) Uraikan permasalahan dan variabel

Permasalahan yang diangkat adalah bagaimana melakukan operasi bitwise antara dua variabel integer a dan b. Variabel a diinisialisasi dengan nilai 10, sedangkan variabel b diinisialisasi dengan nilai 7. Program ini menggunakan tiga operator bitwise: AND (&), OR (|), dan XOR (^) untuk menghitung hasil operasi bitwise antara a dan b, lalu mencetak hasilnya ke layar.

Operasi pertama menggunakan operator AND (&), yang akan membandingkan setiap bit dari a dan b, dan menghasilkan 1 hanya jika kedua bit pada posisi yang sama adalah 1. Operasi kedua menggunakan operator OR (|), yang akan menghasilkan 1 jika salah satu dari bit pada posisi yang sama adalah 1. Operasi terakhir menggunakan operator XOR (^), yang akan menghasilkan 1 hanya jika salah satu dari bit adalah 1 tetapi bukan keduanya. Hasil dari setiap operasi ini disimpan dalam variabel hasil, yang kemudian dicetak ke layar menggunakan `System.out.println`. Program ini mengilustrasikan cara penggunaan operator bitwise untuk melakukan manipulasi level bit di Java.

[No.7] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menambahkan operasi penggunaan operator bitwise dalam bahasa pemrograman Java. Program ini mendeklarasikan dua variabel integer, a dengan nilai 10 dan b dengan nilai 7, kemudian melakukan operasi bitwise antara kedua variabel tersebut. Pertama, program menggunakan operator bitwise AND (&) yang melakukan operasi AND pada tiap bit yang sesuai dari kedua operand. Dalam hal ini, 10 dalam biner adalah 1010 dan 7 adalah 0111, sehingga hasil `a & b` adalah 0010 atau 2 dalam desimal. Hasil ini kemudian dicetak dengan `System.out.println`.

[No.7] Penyusunan Algoritma dan Kode Program

1) Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.

- (a) Mulai Program
- (b) Deklarasi Variabel
- (c) Operasi Bitwise AND (&)
- (d) Operasi Bitwise XOR (^)
- (e) Akhir Program

2) Kode program dan luaran.

```
1 public class operator {
2     public static void main(String[] args) {
3         // Mendeklarasikan dan menginisialisasi variabel a dan b
4         int a = 10; // dalam biner: 1010
5         int b = 7;  // dalam biner: 0111
6         int hasil;
7
8         // Operator bitwise AND (&)
9         hasil = a & b;
10        System.out.println("Hasil dari a & b : " + hasil );
11
12        // Operator bitwise OR (|)
13        hasil = a | b;
14        System.out.println("Hasil dari a | b : " + hasil );
15
16        // Operator bitwise XOR (^)
17        hasil = a ^ b;
18        System.out.println("Hasil dari a ^ b : " + hasil );
19
20        // Operator bitwise NOT (~)
21        hasil = ~a;
22        System.out.println("Hasil dari ~a : " + hasil );
23
24        // Operator bitwise shift kanan (>>)
25        hasil = a >> 1;
26        System.out.println("Hasil dari a >> 1 : " + hasil );
27
28        // Operator bitwise shift kiri (<<)
29        hasil = b << 2;
30        System.out.println("Hasil dari b << 2 : " + hasil );
31    }
32 }
```

Gambar Input kode 7.0

a) Screenshot/ Capture potongan kode dan hasil luaran.

```
Hasil dari a & b : 2
Hasil dari a | b : 15
Hasil dari a ^ b : 13
Hasil dari ~a : -11
Hasil dari a >> 1 : 5
Hasil dari b << 2 : 28
```

Gambar Output kode 7.1

b) Analisa luaran yang dihasilkan.

Luaran sudah sesuai dengan program yang disusun.

Tipe data yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data.

[No.7] Kesimpulan

1) Analisa.

Program Java ini menunjukkan penggunaan operator bitwise. Operator & menghasilkan hasil bitwise AND, | untuk bitwise OR, dan ^ untuk bitwise XOR antara dua angka. Operator ~ adalah bitwise NOT yang membalikkan semua bit dari angka, sedangkan >> adalah bitwise right shift yang menggeser bit ke kanan, dan << adalah bitwise left shift yang menggeser bit ke kiri. Hasil dari operasi ini tergantung pada representasi biner dari angka-angka yang dioperasikan.