|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **ADEL JANUARTI SAPUTRI G1F024010** | **OPERATOR JAVA** | **17 SEPTEMBER 2024** |

**[No. 1] Identifikasi Masalah:**

Pada soal masih ada pesan kesalahan:

System.out.println("a + b = " + (a - b));

**[No.1] Analisis dan Argumentasi**

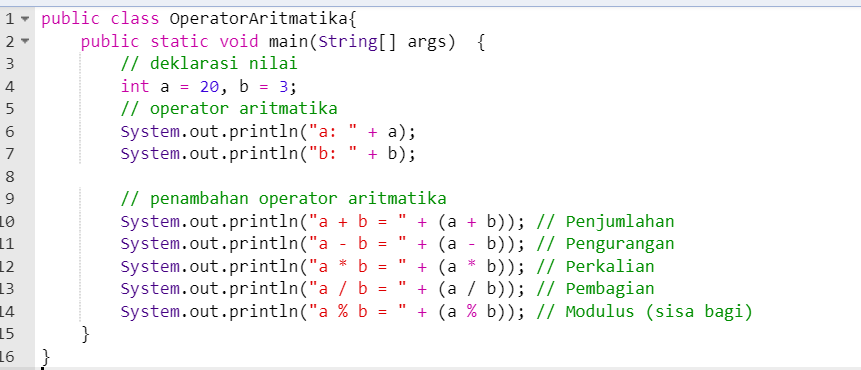
1. Seharusnya, operator yang digunakan adalah **penjumlahan (+)**, tetapi dalam kode tersebut digunakan operator **pengurangan (-)**. Hal ini menyebabkan hasilnya menjadi pengurangan alih-alih penjumlahan dan Solusi nya adalah ubah operator pengurangan menjadi penjumlahan pada baris tersebut

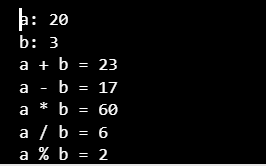
**[No.1 ] Penyusunan Algoritma dan Kode Program**

1. Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.  
Misalkan algoritma membuat batik tulis:

1. Membuat desain batik
2. Melukis di kain
3. Menutupi bagian putih
4. Perwarnaan kain
5. Melukis Kembali dengan canting
6. Menghilangkan lilin
7. Membatik lagi
8. Nglorot
9. Mencuci kain batik
10. Kode program dan luaran





1. Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot

1. Analisa luaran yang dihasilkan adalah:  
   Luaran sudah sesuai dengan program yang disusun dengan  
   Tipe data yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data

**PEMBAHASAN DENGAN BENTUK ANALISA (1)**

* 1. Susunlah Kesimpulan berdasarkan permasalahan,algoritma,dan kode program!
  2. Apakah dasar alasan pengambilan Keputusan anda untuk kasus ini

1. **1.PERMASALAHAN:**

Kode yang diberikan memiliki beberapa kesalahan sintaks, , operator yang digunakan adalah **penjumlahan (+)**, tetapi dalam kode tersebut digunakan operator **pengurangan (-)**

1. **Algoritma**:
   * 1. ubah operator pengurangan menjadi penjumlahan pada baris tersebut
2. Alasan mengambil Keputusan
3. **Inkoherensi antara Teks dan Operasi**: Pada baris ini: System.out.println("a + b = " + (a - b)); yaitu Secara teks, program menyatakan bahwa operasi yang dilakukan adalah "a + b" (penjumlahan), tetapi operator yang digunakan dalam perhitungan sebenarnya adalah pengurangan (a - b). Hal ini menciptakan ketidaksesuaian antara apa yang dinyatakan dengan apa yang dilakukan
4. **Kesalahan Logika dalam Pemrograman**: Jika tujuan program adalah menampilkan hasil penjumlahan (sesuai dengan label "a + b"), maka operator yang digunakan haruslah penjumlahan (+). Menggunakan operator pengurangan (-) tidak sesuai dengan tujuan yang diharapkan
5. **Membuat Kode Lebih Konsisten dan Mudah Dibaca**: Dengan mengubah operator pengurangan menjadi penjumlahan, kode menjadi lebih konsisten antara apa yang ditampilkan kepada pengguna (teks "a + b") dan apa yang dilakukan oleh program (operasi a + b). Ini membuat kode lebih mudah dibaca, dipahami, dan sesuai dengan harapan

Jawaban Analisa:

Masalah utama dalam program ini adalah kesalahan penggunaan operator aritmatika yang tidak sesuai dengan teks yang ditampilkan. Kode menunjukkan "a + b", namun menggunakan operator pengurangan (a - b), yang menghasilkan hasil yang salah. Dengan mengganti operator pengurangan menjadi penjumlahan, program akan menghasilkan output yang benar dan sesuai dengan yang diharapkan

**LATIHAN 2**

2.1Bandingkan hasil Contoh 1 dengan Contoh 2!

Jawaban :

Contoh 1 adalah operator aritmatika yang memiliki kesalahan logika karena operator yang digunakan dalam perhitungan tidak sesuai dengan label yang ditampilkan (seharusnya penjumlahan tetapi yang dilakukan adalah pengurangan)

Contoh 2 adalah operator penugasan menggunakan operator penugasan untuk menjalankan berbagai operasi aritmatika dengan benar, dan setiap hasil operasi ditampilkan dengan label yang sesuai

**[NO 2] IDENTIFIKASI MASALAH**

Pada soal masih ada kesalahan yaitu:

Nilai b terus diperbarui, yang menyebabkan hasil setiap operasi aritmatika terkait langsung dengan operasi sebelumnya.

Inisialisasi ulang variabel mungkin diperlukan jika setiap operasi harus dilakukan berdasarkan nilai awal variabel.

Pembagian integer menghasilkan pembulatan ke bawah (hasil bulat). Jika Anda membutuhkan hasil pecahan, gunakan tipe data double atau float

**[NO 2] ANALISIS DAN ARGUMENTASI**

Kode asli bekerja dengan baik, tetapi hasil dari setiap operasi bergantung pada hasil operasi sebelumnya. Ini bisa menimbulkan kebingungan jika setiap operasi harus berdiri sendiri.

Inisialisasi ulang nilai b sebelum setiap operasi dapat membantu memisahkan masing-masing operasi dan membuat hasil lebih mudah dipahami.

Pembagian integer akan menghasilkan hasil bulat, dan hasil pecahan memerlukan tipe data double atau float

**[NO 2] PENYUSUNAN ALGORITMA DAN KODE PROGRAM**

1.algoritma

Deklarasi variabel: Buat dua variabel a dengan nilai 20 dan b dengan nilai 3.

Penambahan:

Tambahkan nilai a ke b

Cetak hasil penambahan

Pengurangan:

Kurangi nilai b dengan a

Cetak hasil pengurangan

Perkalian:

Kalikan nilai b dengan a

Cetak hasil perkalian

Pembagian:

Bagi nilai b dengan a

Cetak hasil pembagian

Sisa bagi:

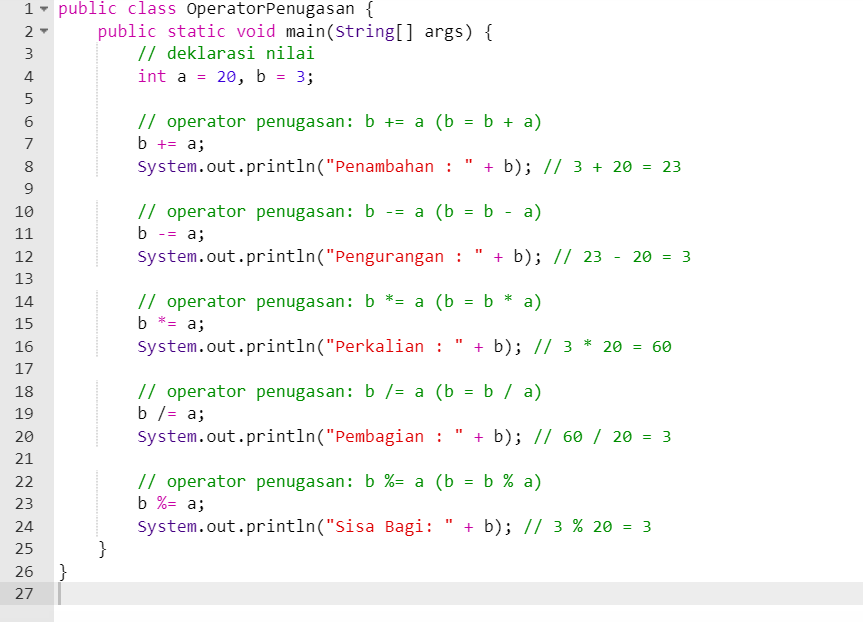
Hitung sisa bagi dari nilai b oleh a

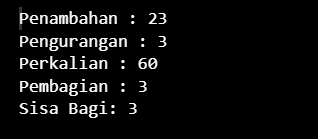
Cetak hasil sisa bagi

2.kode program dan luaran

a) Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot





b.analisa luaran yang dihasilkan

Luaran sudah sesuai dengan program yang disusun Tipe data yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data

**LATIHAN 3**

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

**3.2. Rekomendasikan tipe data yang sesuai untuk data tersebut:**

1. A > B:
2. Sebelumnya: true (karena 12 > 4)
3. Sekarang: false (karena 4 tidak lebih besar dari 4)
4. Penjelasan: Dengan nilai yang sama, 4 > 4 tidak benar.
5. A < B:
6. Sebelumnya: false (karena 12 tidak lebih kecil dari 4)
7. Sekarang: false (karena 4 tidak lebih kecil dari 4)
8. Penjelasan: Dengan nilai yang sama, 4 < 4 tidak benar.
9. A >= B:
10. Sebelumnya: true (karena 12 lebih besar atau sama dengan 4)
11. Sekarang: true (karena 4 sama dengan 4, dan 4 >= 4 benar)
12. Penjelasan: 4 >= 4 benar karena >= mencakup kesamaan.
13. A <= B:
14. Sebelumnya: false (karena 12 tidak lebih kecil atau sama dengan 4)
15. Sekarang: true (karena 4 sama dengan 4, dan 4 <= 4 benar)
16. Penjelasan: 4 <= 4 benar karena <= mencakup kesamaan.
17. A == B:
18. Sebelumnya: false (karena 12 tidak sama dengan 4)
19. Sekarang: true (karena 4 sama dengan 4)
20. Penjelasan: 4 == 4 benar karena kedua nilai sama.
21. A != B:
22. Sebelumnya: true (karena 12 tidak sama dengan 4)
23. Sekarang: false (karena 4 sama dengan 4)
24. Penjelasan: 4 != 4 tidak benar karena kedua nilai sama

**[NO 3] IDENTIFIKASI MASALAH:**

nilaiA > nilaiB (12 > 4) → true karena 12 lebih besar dari 4

nilaiA < nilaiB (12 < 4) → false karena 12 tidak lebih kecil dari 4

nilaiA >= nilaiB (12 >= 4) → true karena 12 lebih besar atau sama dengan 4

nilaiA <= nilaiB (12 <= 4) → false karena 12 tidak lebih kecil atau sama dengan 4

nilaiA == nilaiB (12 == 4) → false karena 12 tidak sama dengan 4

nilaiA != nilaiB (12 != 4) → true karena 12 tidak sama dengan 4

**[NO 3] ANALISIS DAN ARGUMENTASI**

1.) Deklarasi variabel :

nilaiA = 12 dan nilaiB = 4: Dua variabel integer yang akan dibandingkan

boolean hasil: Variabel yang akan menyimpan hasil perbandingan (benar atau salah)

2.)penggunan operator rasional

> (lebih besar dari)

< (lebih kecil dari)

>= (lebih besar atau sama dengan)

<= (lebih kecil atau sama dengan)

== (sama dengan)

!= (tidak sama dengan)

**[NO 3] PENYUSUNAN ALGORITMA DAN KODE PROGRAM**

1.algoritma

a. Inisialisasi Variabel

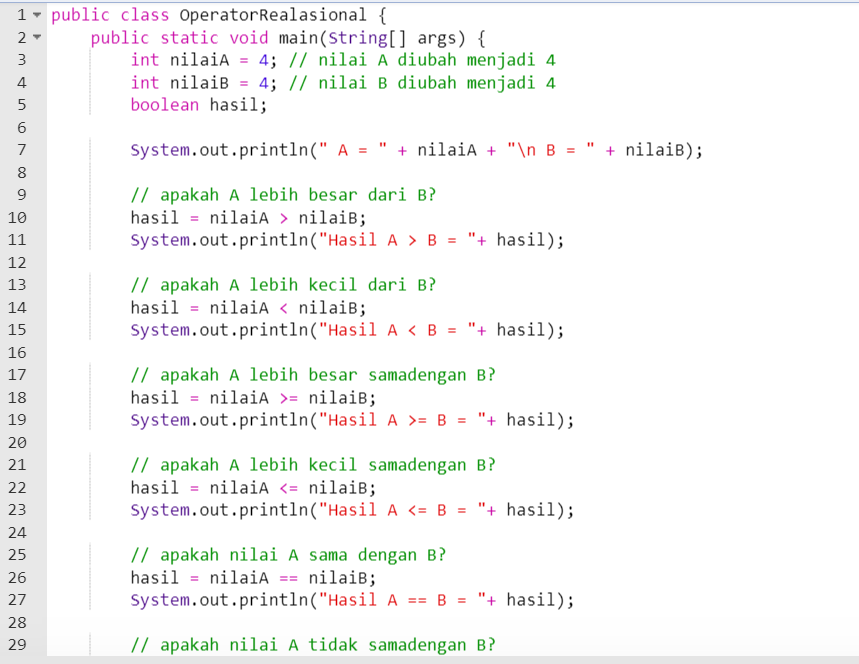
b. Tampilkan nilai A dan B

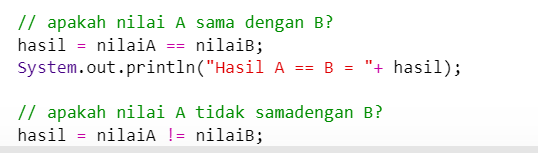
c. Bandingkan nilai A dan B

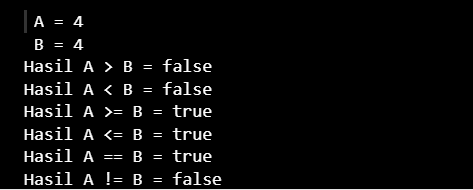
d. Tampilkan Hasil Perbandingan

e. selesai

kode program dan luaran







a.Screenshot/ Capture potongan kode dan hasil luaran beri komentar pada kode yang di Screenshot

b.Analisa luaran yang dihasilkan

Luaran sudah sesuai dengan program yang disusun. Operator yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data

**[NO 3] KESIMPULAN ANALISA**

Pada program itu saya menggunakn bentuk kelas public karena di Java memberikan keuntungan dalam hal visibilitas dan aksesibilitas, mendukung modularitas, interoperabilitas, dan konsistensi struktur kode

**LATIHAN 4**

**[NO 4] IDENTIFIKASI MASALAH**

Pada soal masih terdapat kesalahan yaitu :

Koreksi pada Program: Tidak ada masalah teknis dalam program. Program bekerja sesuai dengan logika increment dan decrement

Pemahaman:

Post-increment/decrement (a++, c--): Gunakan nilai lama, lalu tingkatkan/turunkan nilai.

Pre-increment/decrement (++b, --d): Tingkatkan/turunkan nilai terlebih dahulu, lalu gunakan nilainya.

Luaran Sesuai: Output yang diberikan oleh program sesuai dengan ekspektasi berdasarkan penggunaan operator increment dan decrement

**[NO 4] ANALISIS DAN ARGUMENTASI**

Untuk setiap operator, variabel awal diinisialisasi dengan nilai 10, dan hasilnya dicetak dalam beberapa langkah untuk menunjukkan perbedaan antara post dan pre increment/decrement

**[NO 4**] **PENYUSUNAN ALGORITMA DAN KODE PROGRAM**

1.algoritma

1.)Inisialisasi Variabel:

Inisialisasi 4 variabel (a, b, c, dan d) masing-masing dengan nilai 10.

2.)Post-Increment:

Cetak nilai awal dari variabel a.

Gunakan post-increment (a++), cetak nilai sebelum increment, lalu cetak nilai setelah increment.

3.)Pre-Increment:

Cetak nilai awal dari variabel b.

Gunakan pre-increment (++b), cetak nilai setelah increment.

4.)Post-Decrement:

Cetak nilai awal dari variabel c.

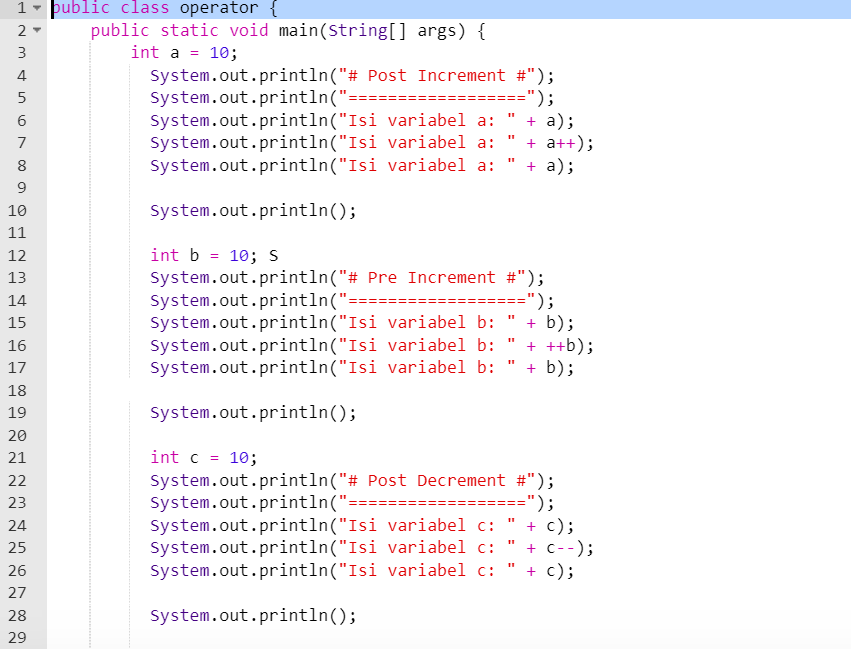
Gunakan post-decrement (c--), cetak nilai sebelum decrement, lalu cetak nilai setelah decrement.

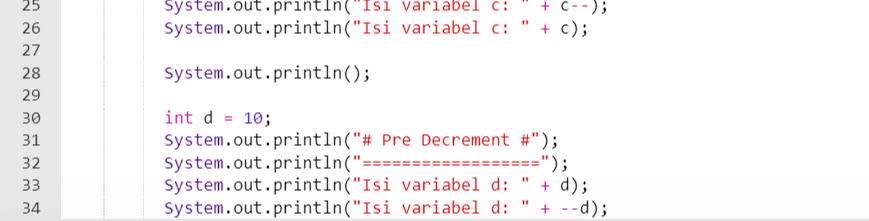
Pre-Decrement:

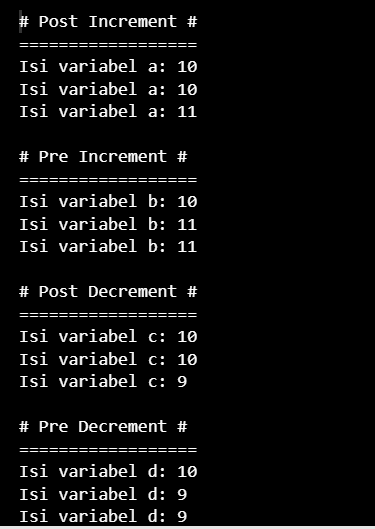
Cetak nilai awal dari variabel d.

Gunakan pre-decrement (--d), cetak nilai setelah decrement.

2.Kode program dan luaran







a.)Screenshot/ Capture potongan kode dan hasil luaranBeri komentar pada kode yang di Screenshot

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 ANALISA**

1.)Post-Increment (a++):  
Pada post-increment, nilai dari variabel a yang dicetak pertama kali adalah **10** karena operasi increment dilakukan setelah nilai dicetak. Setelah increment, nilai a menjadi **11** pada langkah selanjutnya.

2.)Pre-Increment (++b):  
Pada pre-increment, nilai dari variabel b langsung di-increment terlebih dahulu sebelum dicetak, sehingga nilai yang dicetak adalah **11** dari awal setelah operasi increment.

3.) Post-Decrement (c--):  
Pada post-decrement, nilai dari variabel c yang dicetak pertama kali adalah **10** karena decrement dilakukan setelah nilai dicetak. Setelah operasi, nilai c menjadi **9**.

4.)Pre-Decrement (--d):  
Pada pre-decrement, nilai dari variabel d langsung di-decrement sebelum dicetak, sehingga nilai yang ditampilkan langsung **9** dari awal

**LATIHAN 5**

**[NO 5] IDENTIFIKASI MASALAH**

Pada kode program OperatorLogika, tujuannya adalah mengevaluasi operasi logika dengan dua variabel boolean, a dan b, menggunakan operator AND (&&). Operator AND menghasilkan true jika kedua operand bernilai true, dan menghasilkan false jika salah satu atau kedua operand bernilai false

**[NO 5] ANALISIS DAN ARGUMENTASI**

Program OperatorLogika menggunakan operator logika AND (&&) untuk mengevaluasi dua nilai boolean: a yang bernilai true dan b yang bernilai false. Operator logika AND bekerja dengan aturan berikut:

1.)true && true menghasilkan true.

2.)true && false, false && true, dan false && false semuanya menghasilkan false.

Dalam kode ini, variabel c menyimpan hasil dari operasi a && b. Karena salah satu operand (b) bernilai false, hasil akhirnya adalah false. Program kemudian mencetakhasil ini menggunakan System.out.println, yang akan mencetak "true && false = false"

**[NO 5**] **PENYUSUNAN ALGORITMA DAN KODE PROGRAM**

1. algoritma

1.)Inisialisasi Variabel:

* Buat dua variabel boolean, a dan b, dan setel a ke true dan b ke false.

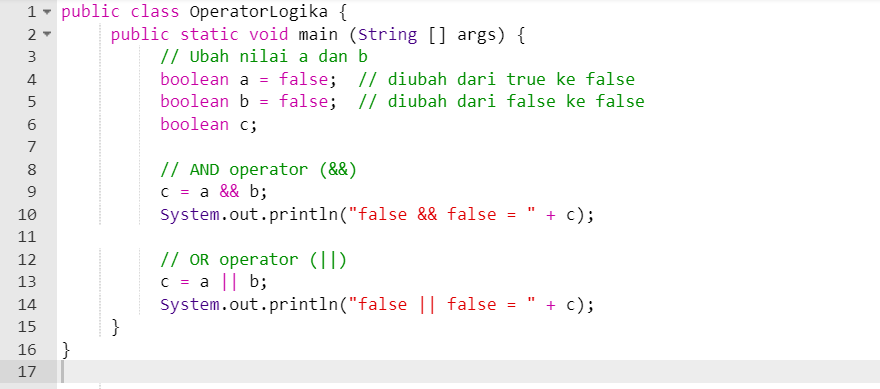
2.) Operasi Logika:

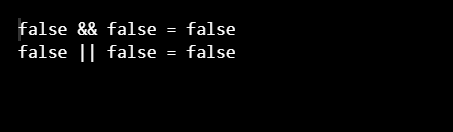
* Gunakan operator logika AND (&&) untuk mengevaluasi hasil dari a && b dan simpan hasilnya dalam variabel c.

3.)Output Hasil:

* Cetak hasil dari operasi logika ke layar dengan format: "true && false = hasil", di mana hasil adalah nilai dari variabel c

2.Kode program dan luaran





a.)Screenshot/ Capture potongan kode dan hasil luaranBeri komentar pada kode yang di Screenshot

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 ANALISA**

1.)Program ini secara efektif mendemonstrasikan bagaimana operator logika AND bekerja dalam Java.

2.) Hasil yang dicetak (false) sesuai dengan aturan bahwa hasil dari true && false adalah false.

3.) Penggunaan operator logika seperti ini umum dalam kondisi pengambilan keputusan di program yang memerlukan evaluasi apakah semua kondisi terpenuhi sebelum melanjutkan eksekusi

**LATIHAN 6**

**[NO 6] IDENTIFIKASI MASALAH**

Program ini menggunakan operator ternary untuk menetapkan nilai variabel status berdasarkan nilai integer nilai. Dalam kasus ini, program berhasil mengevaluasi kondisi dan mencetak "Lulus" karena nilai yang diberikan melebihi batas yang ditentukan. Masalah yang ada adalah inisialisasi variabel status yang tidak perlu, tetapi ini tidak mempengaruhi hasil akhir program

**[NO 6] ANALISIS DAN ARGUMENTASI**

1. Penggunaan Operator Ternary:
   * Operator Ternary adalah bentuk singkat dari pernyataan if-else yang digunakan untuk memilih antara dua nilai berdasarkan kondisi.
   * Formatnya adalah kondisi ? nilai\_true : nilai\_false.
   * Dalam kode ini, status = (nilai > 60) ? "Lulus" : "Gagal"; berarti jika nilai lebih besar dari 60, maka status diatur ke "Lulus", jika tidak maka diatur ke "Gagal".
2. Evaluasi Kondisi:
   * Variabel nilai diinisialisasi dengan 80.
   * Kondisi nilai > 60 dievaluasi sebagai true karena 80 lebih besar dari 60.
   * Oleh karena itu, bagian dari operator ternary yang diambil adalah "Lulus".
3. Inisialisasi Variabel:
   * Variabel status diinisialisasi dengan string kosong "", tetapi langsung diatur ulang dengan hasil dari operasi ternary.
   * Inisialisasi awal ini tidak mempengaruhi hasil akhir tetapi bisa dianggap tidak perlu karena status langsung diatur pada baris berikutnya.
4. Output:
   * System.out.println(status); mencetak nilai dari status, yang merupakan hasil dari evaluasi operator ternary.
   * Dalam kasus ini, keluaran yang dicetak adalah "Lulus", sesuai dengan kondisi yang benar.

**[NO 6**] **PENYUSUNAN ALGORITMA DAN KODE PROGRAM**

1.)algoritma

1.)Inisialisasi Variabel:

* Buat variabel nilai dan tetapkan nilainya dengan 80.
* Buat variabel status yang akan menyimpan hasil keputusan.

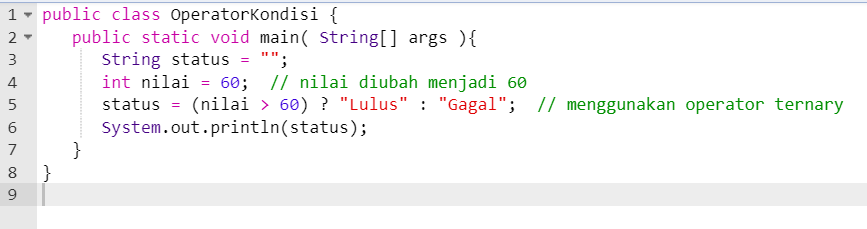
2.)Evaluasi Kondisi:

* Gunakan operator ternary untuk memeriksa apakah nilai lebih besar dari 60.
* Jika benar, setel status ke "Lulus".
* Jika salah, setel status ke "Gagal".

3.)Cetak Hasil:

* Tampilkan nilai dari variabel status ke layar.

2.Kode program dan luaran





a.)Screenshot/ Capture potongan kode dan hasil luaranBeri komentar pada kode yang di Screenshot

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 ANALISA**

Program ini secara efektif menggunakan operator ternary untuk menetapkan status kelulusan berdasarkan nilai integer yang diberikan. Penggunaan operator ternary membuat kode lebih ringkas dan mudah dibaca dibandingkan dengan pernyataan if-else. Hasil output dari program adalah "Lulus", yang sesuai dengan kondisi nilai yang lebih besar dari 60

**LATIHAN 7**

**[NO 7] IDENTIFIKASI MASALAH**

Program ini menggunakan operasi bitwise pada dua integer (a dan b) dan mencetak hasilnya. Berikut adalah hasil dari masing-masing operasi bitwise berdasarkan nilai yang diberikan:

* **a & b**: Menghasilkan hasil bitwise AND.
* **a | b**: Menghasilkan hasil bitwise OR.
* **a ^ b**: Menghasilkan hasil bitwise XOR.
* **~a**: Menghasilkan hasil bitwise NOT.
* **a >> 1**: Menggeser bit a ke kanan dengan 1 posisi.
* **b << 2**: Menggeser bit b ke kiri dengan 2 posisi

**[NO 7] ANALISIS DAN ARGUMENTASI**

1.) Bitwise AND (&):

* Memberikan hasil yang merupakan bit yang sama dari a dan b. Dalam contoh ini, hanya bit ke-2 yang sama di kedua angka (2).

2.)Bitwise OR (|):

* Memberikan hasil yang memiliki bit 1 jika salah satu dari bit dalam a atau b adalah 1. Dalam contoh ini, semua bit menjadi 1 karena operasi OR (15).

3.)Bitwise XOR (^):

* Memberikan hasil yang memiliki bit 1 jika bit dalam a dan b berbeda. Dalam contoh ini, hasilnya adalah 13 karena perbedaan bit.

4.)Bitwise NOT (~):

* Membalik semua bit dari a dan memberikan hasil negasi bitwise. Untuk 10, hasilnya adalah -11 dalam representasi dua's complement.

5.) Right Shift (>>):

* Menggeser bit ke kanan dengan mempertahankan bit tanda. Dalam contoh ini, hasilnya adalah 5 setelah satu posisi geser.

6.)Left Shift (<<):

* Menggeser bit ke kiri dan mengisi bit kosong dengan 0. Dalam contoh ini, hasilnya adalah 28 setelah dua posisi geser.

Program ini dengan jelas menunjukkan hasil dari berbagai operasi bitwise pada variabel integer a dan b. Hasil yang diperoleh sesuai dengan aturan operasi bitwise, memberikan wawasan tentang bagaimana bitwise operasi memanipulasi bit individual dalam representasi biner dari angka. Program ini efektif untuk memahami dasar-dasar operasi bitwise dalam pemrograman Java

**[NO 7**] **PENYUSUNAN ALGORITMA DAN KODE PROGRAM**

1.)algoritma

1.)Inisialisasi Variabel:

* Setel variabel a dengan nilai 10.
* Setel variabel b dengan nilai 7.

2.)Operasi Bitwise dan Output:

* Bitwise AND (a & b):
  + Lakukan operasi bitwise AND antara a dan b.
  + Cetak hasilnya.
* Bitwise OR (a | b):
  + Lakukan operasi bitwise OR antara a dan b.
  + Cetak hasilnya.
* Bitwise XOR (a ^ b):
  + Lakukan operasi bitwise XOR antara a dan b.
  + Cetak hasilnya.
* Bitwise NOT (~a):
  + Lakukan operasi bitwise NOT pada a.
  + Cetak hasilnya.

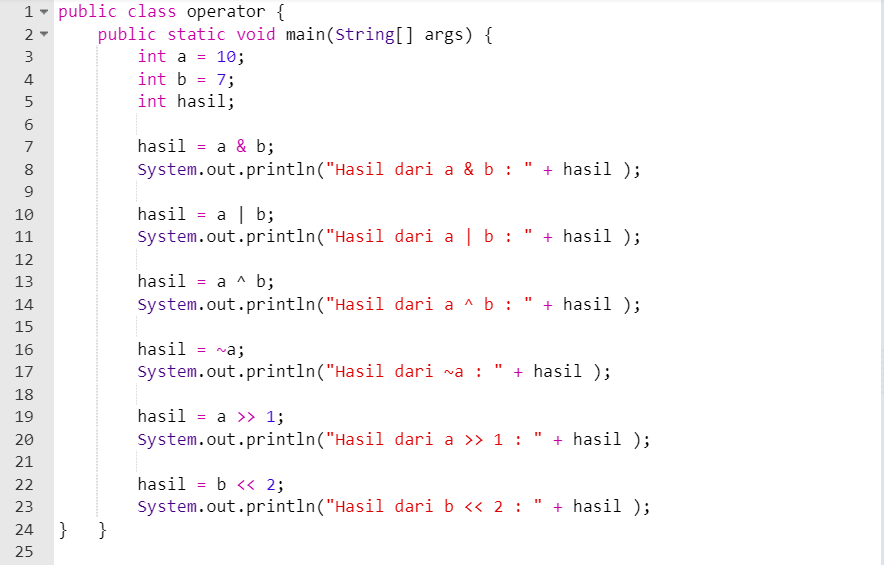
3.)Right Shift (a >> 1):

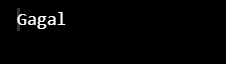
* + Lakukan operasi right shift pada a dengan satu posisi.
  + Cetak hasilnya.

4.)Left Shift (b << 2):

* + Lakukan operasi left shift pada b dengan dua posisi.
  + Cetak hasilnya

2.Kode program dan luaran





a.)Screenshot/ Capture potongan kode dan hasil luaranBeri komentar pada kode yang di Screenshot

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 ANALISA**

Program ini menunjukkan penggunaan operasi bitwise (&, |, ^, ~) dan shift (>>, <<). Hasil yang ditunjukkan mencerminkan bagaimana setiap operasi memanipulasi representasi biner dari angka. Ini memberikan pemahaman tentang bagaimana bitwise operations mempengaruhi nilai dan bagaimana shifting bit memengaruhi data