|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Ariansyah Putra Ramadhan**  **G1F024080** | **Operator Aritmatika: +, -, \*, /** | **25 September 2024** |

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

Diketahui dari soal :

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

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

Analisis

**Deklarasi Variabel:**

* Dua variabel integer, a dan b, dideklarasikan dan diinisialisasi dengan nilai masing-masing 20 dan 3. Ini menunjukkan bahwa program akan bekerja dengan data numerik integer.

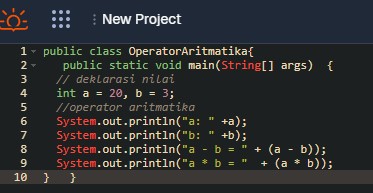
**Operasi Aritmatika:**

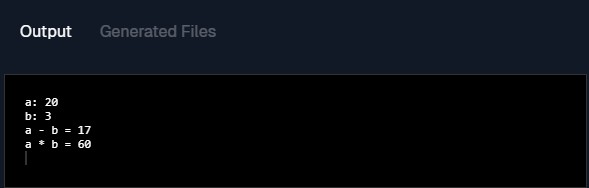
* Program melakukan empat operasi aritmatika dasar: penjumlahan (a + b), pengurangan (a - b), perkalian (a \* b), dan pembagian (a / b).
* Hasil dari setiap operasi dihitung dan disimpan dalam variabel yang sesuai.

**[No.2 ] Penyusunan Algoritma dan Kode Program**

1. Algoritma
2. Mulai program.
3. Deklarasikan dua variabel integer, a dan b, dan inisialisasi dengan nilai 20 dan 3.
4. Cetak nilai variabel a dan b ke layar.
5. Lakukan operasi perkalian: a \* b dan simpan hasilnya dalam variabel sementara.
6. Cetak hasil operasi pengurangan dan perkalian ke layar.
7. Akhiri program.
8. Kode program dan luaran
9. Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot





1. Analisa luaran yang dihasilkan :

Program ini akan menghasilkan keluaran yang sesuai dengan operasi aritmatika yang didefinisikan. Nilai-nilai variabel a dan b akan dicetak, diikuti oleh hasil pengurangan dan perkalian antara kedua variabel.

**[No.1] Kesimpulan**

1. **Analisa**
2. Susunlah kesimpulan berdasarkan permasalahan, algoritma, dan kode program!

**Permasalahan:**

Program ini dirancang untuk melakukan operasi aritmatika sederhana, yaitu pengurangan dan perkalian, terhadap dua bilangan bulat

**Algoritma:**

1. Inisialisasi: Menetapkan nilai awal untuk dua variabel integer, yaitu a dan b.
2. Output: Menampilkan nilai awal dari kedua variabel.
3. Operasi Aritmatika: Melakukan operasi pengurangan dan perkalian antara a dan b.
4. Output Hasil: Menampilkan hasil dari operasi pengurangan dan perkalian.

**Kode Program:**

Kode program di atas mengimplementasikan algoritma tersebut dalam bahasa pemrograman Java. Setiap baris kode memiliki fungsi spesifik:

* Deklarasi variabel: Mendefinisikan variabel a dan b dengan tipe data integer.
* Output: Menggunakan System.out.println() untuk mencetak nilai variabel dan hasil operasi ke konsol.
* Operasi aritmatika: Melakukan operasi pengurangan dan perkalian menggunakan operator - dan \*.

1. Apakah dasar alasan pengambilan keputusan Anda untuk kasus ini?

Dasar alasan membuat program ini adalah untuk melakukan operasi aritmatika sederhana.

Program Operator Aritmatika dirancang untuk:

1. Mendemonstrasikan penggunaan variabel: Program menggunakan dua variabel integer, a dan b, untuk menyimpan nilai-nilai yang akan dioperasikan.
2. Melakukan operasi aritmatika: Program melakukan operasi pengurangan dan perkalian antara variabel a dan b.
3. Menampilkan hasil operasi: Program mencetak hasil operasi pengurangan dan perkalian ke layar.

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Ariansyah Putra Ramadhan**  **G1F024080** | **Operator Penugasan  ( = )** | **25 September 2024** |

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

1. Diketahui dari soal :

2.1. Bandingkan hasil Contoh 1 dengan Contoh 2! .

**[No.2] Analisis dan Argumentasi**

* Operator Aritmatika lebih cocok untuk perhitungan sederhana dan demonstrasi dasar.
* Operator Penugasan lebih cocok untuk operasi yang melibatkan perubahan nilai variabel secara berulang dan efisien.

**[No.3 ] Penyusunan Algoritma dan Kode Program**

1. Algoritma
2. Mulai:

* Deklarasikan dua variabel bilangan bulat, a dan b, dan berikan nilai awal masing-masing 20 dan 3.

1. Operasi Penambahan:

* Tambahkan nilai variabel a ke variabel b dan simpan hasilnya kembali ke variabel b.
* Tampilkan hasil penambahan.

1. Operasi Pengurangan:

* Kurangkan nilai variabel a dari variabel b dan simpan hasilnya kembali ke variabel b.
* Tampilkan hasil pengurangan.

1. Operasi Perkalian:

* Kalikan nilai variabel a dengan variabel b dan simpan hasilnya kembali ke variabel b.
* Tampilkan hasil perkalian.

1. Operasi Pembagian:

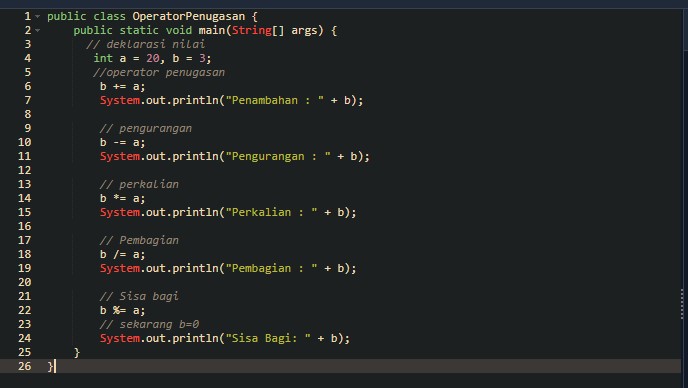
* Bagi nilai variabel b dengan variabel a dan simpan hasilnya kembali ke variabel b.
* Tampilkan hasil pembagian.

1. Operasi Sisa Bagi:

* Bagi nilai variabel b dengan variabel a dan simpan sisa baginya kembali ke variabel b.
* Tampilkan sisa bagi.

1. Selesai
2. Kode program dan luaran
   1. Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot



1. Analisa luaran yang dihasilkan

Program ini secara berurutan melakukan operasi penugasan pada variabel b menggunakan nilai dari variabel a. Setiap operasi mengubah nilai b dan hasilnya langsung ditampilkan ke konsol. Penggunaan operator penugasan membuat kode lebih ringkas dan efisien.

**[No.1] Kesimpulan**

* **Evaluasi**
* Apa konsekuensi/dampak dari kode program yang dibuat?

Program OperatorPenugasan ini, meskipun sederhana, mendemonstrasikan konsep dasar operasi penugasan dalam pemrograman Java. **Tidak ada konsekuensi yang signifikan atau dampak langsung pada sistem secara keseluruhan** akibat menjalankan program ini.

* Evaluasi input program, proses perhitungan, dan luaran yang dihasilkan! (jika ada)

## Evaluasi Program Operator Penugasan

### Input Program:

* **Variabel:** Program ini mendeklarasikan dua variabel bertipe integer, yaitu a dan b.
* **Nilai Awal:** Nilai awal yang diberikan pada variabel adalah:
  + a = 20
  + b = 3
* **Tidak ada input pengguna:** Program ini tidak meminta input dari pengguna selama eksekusi.

### Proses Perhitungan :

Program ini melakukan serangkaian operasi aritmatika menggunakan operator penugasan pada variabel b. Setiap operasi akan mengubah nilai b dan hasilnya langsung ditampilkan ke konsol.

* **Penambahan:** b += a artinya b = b + a. Nilai b akan ditambah dengan nilai a dan hasilnya disimpan kembali ke b.
* **Pengurangan:** b -= a artinya b = b - a. Nilai b akan dikurangi dengan nilai a dan hasilnya disimpan kembali ke b.
* **Perkalian:** b \*= a artinya b = b \* a. Nilai b akan dikalikan dengan nilai a dan hasilnya disimpan kembali ke b.
* **Pembagian:** b /= a artinya b = b / a. Nilai b akan dibagi dengan nilai a dan hasilnya disimpan kembali ke b. Hasil pembagian ini adalah pembagian bilangan bulat, jadi bagian desimalnya akan diabaikan.
* **Sisa Bagi:** b %= a artinya b = b % a. Nilai b akan dibagi dengan a dan sisanya disimpan kembali ke b.

### Luaran Program :

Luaran dari program ini adalah serangkaian nilai yang dicetak ke konsol, menunjukkan hasil dari setiap operasi. Setiap baris output akan menampilkan label operasi dan hasilnya.

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Ariansyah Putra Ramadhan**  **G1F024080** | **Operator Relasional:   <, >, <=, >=, =, ==, !=** | **25 September 2024** |

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

1. Uraikan permasalahan dan variabel

Diketahui dari soal :

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!

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

1. Saya mengusulkan permasalahan ini dapat diatasi dengan cara :

* Mengubah Nilai A dan B menjadi 4
* Membandingkan Perbedaan Nilai

1. Alasan solusi ini karena :

ketika kedua nilai sama, hasil perbandingan akan menjadi sangat sederhana:

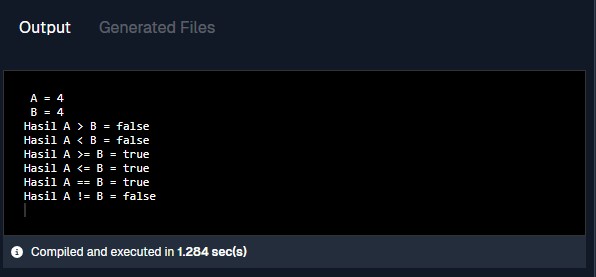
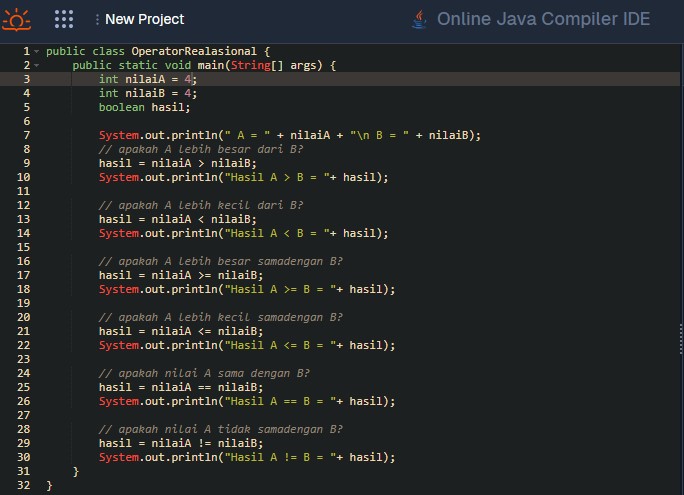
* Perbandingan lebih besar atau lebih kecil: Selalu false.
* Perbandingan sama dengan: Selalu true.
* Perbandingan lebih besar sama dengan atau lebih kecil sama dengan: Selalu true.

Perbaikan kode program dengan cara:

* Komentar: Menjelaskan tujuan setiap bagian kode.
* Nama Variabel: Menggunakan nama yang lebih deskriptif.
* Kasus Uji: Menambah variasi nilai untuk menguji semua kemungkinan.
* Fungsi: Membuat kode lebih modular dan dapat digunakan kembali.
* Struktur Kontrol: Membuat keputusan berdasarkan hasil perbandingan.

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

1. Algoritma
2. Mulai
3. Inisialisasi Variabel
4. Perbandingan
5. Selesai
6. Kode program dan luaran
7. Screenshot/ Capture potongan kode dan hasil luaran Beri komentar pada kode yang di Screenshot



1. Analisa luaran yang dihasilkan

Ketika dua nilai dibandingkan dan nilainya sama, maka:

* Perbandingan "lebih besar" atau "lebih kecil" akan menghasilkan false.
* Perbandingan "sama dengan" atau "lebih besar sama dengan" atau "lebih kecil sama dengan" akan menghasilkan true.
* Perbandingan "tidak sama dengan" akan menghasilkan false.

**[No.1] Kesimpulan**

1. **Evaluasi**
   1. **Apa konsekuensi/dampak dari kode program yang dibuat?**

Program ini tidak memiliki dampak yang signifikan secara mandiri. Namun, konsep yang diimplementasikan dalam program ini (perbandingan menggunakan operator relasional) sangat penting dalam pemrograman. Konsep ini adalah fondasi untuk membangun program yang lebih kompleks dan menyelesaikan masalah yang lebih besar.

### **Evaluasi input program, proses perhitungan, dan luaran yang dihasilkan! (jika ada)**

### Input Program

* **Nilai Awal:** Program menginisialisasi dua variabel integer, yaitu nilaiA dan nilaiB, dengan nilai awal 4.

### Proses Perhitungan

* **Perbandingan:** Program melakukan serangkaian perbandingan antara nilaiA dan nilaiB menggunakan berbagai operator relasional (>, <, >=, <=, ==, !=).
* **Penentuan Hasil:** Hasil dari setiap perbandingan adalah nilai boolean (true atau false), yang kemudian disimpan dalam variabel hasil.

### Luaran Program

Untuk kasus di mana nilaiA dan nilaiB sama-sama bernilai 4, output program akan seperti berikut:

A = 4

B = 4

Hasil A > B = false

Hasil A < B = false

Hasil A >= B = true

Hasil A <= B = true

Hasil A == B = true

Hasil A != B = false

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Ariansyah Putra Ramadhan**  **G1F024080** | **Operator Increment dan Decrement: ++, -** | **25 September 2024** |

**[No. 1] Identifikasi Masalah:**  
Diketahui dari soal :

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

**[No.2] Analisis dan Argumentasi**

Berdasarkan output pada contoh 4, dapat disimpulkan perbedaan utama antara post-increment/decrement dan pre-increment/decrement sebagai berikut:

* Post-increment/decrement: Nilai variabel digunakan terlebih dahulu dalam ekspresi, baru kemudian nilainya diubah.
* Pre-increment/decrement: Nilai variabel diubah terlebih dahulu sebelum digunakan dalam ekspresi.

**[No.3] Penyusunan Algoritma dan Kode Program**

1. Algoritma
2. Mulai
3. Inisialisasi variabel:

* Deklarasikan variabel a, b, c, dan d dengan nilai awal 10.

1. Output awal:

* Cetak nilai awal variabel a, b, c, dan d.

1. Post-increment:

* Cetak nilai a sebelum dan sesudah increment.
* Increment nilai a menggunakan a++.
* Cetak nilai a setelah increment.

1. Pre-increment:

* Cetak nilai b sebelum dan sesudah increment.
* Increment nilai b menggunakan ++b.
* Cetak nilai b setelah increment.

1. Post-decrement:

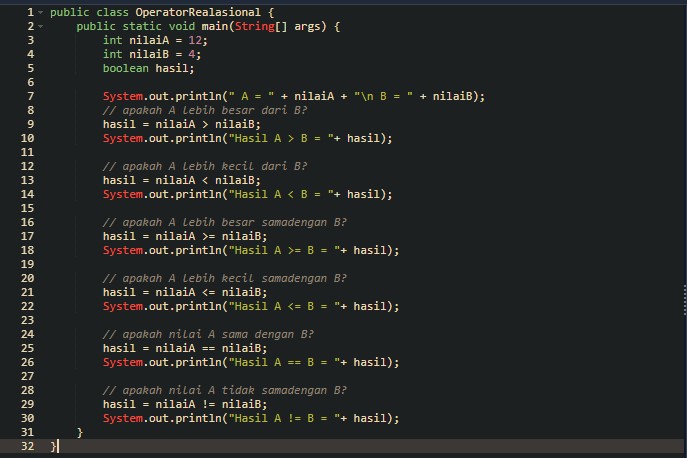
* Cetak nilai c sebelum dan sesudah decrement.
* Decrement nilai c menggunakan c--.
* Cetak nilai c setelah decrement.

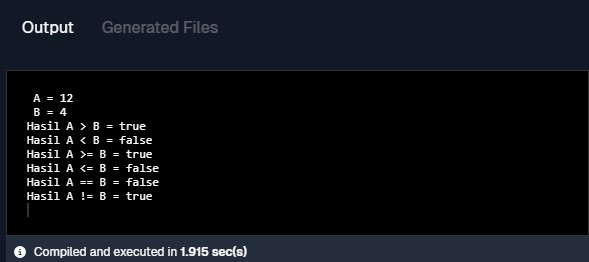
1. Pre-decrement:

* Cetak nilai d sebelum dan sesudah decrement.
* Decrement nilai d menggunakan --d.
* Cetak nilai d setelah decrement.

1. Selesai
2. Kode program dan luaran
3. Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot





Analisa luaran yang dihasilkan :

* **Post-increment/decrement:** Nilai variabel digunakan terlebih dahulu, baru kemudian diubah. Ini berarti nilai yang dicetak pada baris pertama setelah operasi post-increment/decrement akan sama dengan nilai awal variabel.
* **Pre-increment/decrement:** Nilai variabel diubah terlebih dahulu, baru kemudian digunakan. Ini berarti nilai yang dicetak pada baris pertama setelah operasi pre-increment/decrement akan sudah menjadi nilai yang baru setelah diubah.

**[No.1] Kesimpulan**

1. **Evaluasi**
2. Apa konsekuensi/dampak dari kode program yang dibuat?

* Secara umum, program di atas tidak memiliki konsekuensi negatif yang signifikan. Namun, pemahaman yang baik tentang post dan pre increment/decrement dapat membantu dalam menulis kode yang lebih efisien, mudah dibaca, dan menghindari kesalahan logika.

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Ariansyah Putra Ramadhan**  **G1F024080** | **Operator Logika:  &&, ||, !** | **25 September 2024** |

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

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

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

Analisis Perubahan:

* Operator AND (&&):

1. Sebelumnya (a=true, b=false): Hasilnya false karena kedua kondisi harus benar agar menghasilkan true.
2. Sekarang (a=false, b=false): Hasilnya tetap false karena kedua kondisi masih belum terpenuhi.

* Operator OR (||):

1. Sebelumnya (a=true, b=false): Hasilnya true karena salah satu kondisi sudah benar.
2. Sekarang (a=false, b=false): Hasilnya menjadi false karena tidak ada satupun kondisi yang benar.

Perbedaan:

* Operator AND (&&): Hanya akan menghasilkan true jika semua kondisi benar. Jika ada satu saja yang salah, hasilnya akan false.
* Operator OR (||): Akan menghasilkan true jika salah satu atau kedua kondisi benar. Hanya akan menghasilkan false jika kedua kondisi salah.

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

1. Algoritma
2. Mulai
3. Inisialisasi variabel:

* Deklarasikan variabel a, b, dan c sebagai tipe boolean.
* Inisialisasi a dan b dengan nilai false.

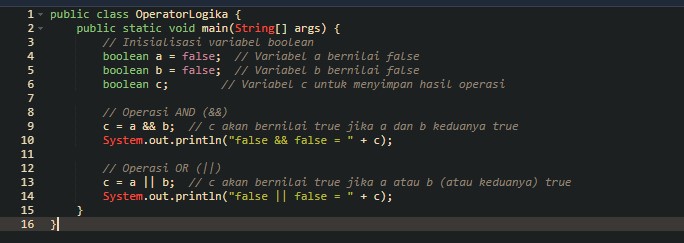
1. Operasi AND:

* Hitung hasil dari a && b dan simpan hasilnya dalam variabel c.
* Cetak hasil operasi ke konsol.

1. Operasi OR:

* Hitung hasil dari a || b dan simpan hasilnya dalam variabel c.
* Cetak hasil operasi ke konsol.

1. Selesai
2. Kode program dan luaran
3. Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot

1. Analisa luaran yang dihasilkan   
   Luaran tersebut merupakan hasil eksekusi dari dua operasi logika dasar dalam pemrograman, yaitu **AND** (&&) dan **OR** (||) yang diterapkan pada dua nilai boolean, false dan false.

**[No.1] Kesimpulan**

1. **Evaluasi**

* Apa konsekuensi/dampak dari kode program yang dibuat?

Kode program di atas pada dasarnya adalah sebuah contoh sederhana yang tidak memiliki dampak langsung yang signifikan. Namun, pemahaman tentang operator logika sangat penting dalam pemrograman, karena operator ini digunakan untuk membuat keputusan, mengontrol aliran program, dan membangun algoritma yang lebih kompleks.

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Ariansyah Putra Ramadhan**  **G1F024080** | **Operator Kondisional (Ternary):  ?:** | **25 September 2024** |

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

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

**[No.2] Analisis dan Argumentasi**

Analisis:

* Perubahan Nilai: Nilai variabel nilai sekarang menjadi 60.
* Kondisi: Kondisi dalam operator ternary adalah nilai > 60.
* Evaluasi Kondisi:
  + Karena nilai sekarang sama dengan 60, maka kondisi nilai > 60 bernilai false.
* Hasil:
  + Ketika kondisi bernilai false, maka nilai yang diberikan setelah tanda ":" kedua akan digunakan.
  + Dalam hal ini, nilai yang diberikan adalah "Gagal".
* Output:
  + Program akan mencetak "Gagal" ke konsol.

Proses yang Terjadi:

1. Inisialisasi: Variabel status diinisialisasi sebagai string kosong, nilai diinisialisasi dengan nilai 60.
2. Evaluasi Kondisi: Kondisi nilai > 60 dievaluasi. Karena nilai adalah 60, kondisi ini tidak terpenuhi (false).
3. Penentuan Nilai: Karena kondisi false, maka nilai "Gagal" diberikan kepada variabel status.
4. Output: Nilai variabel status ("Gagal") dicetak ke konsol.

**[No.3 ] Penyusunan Algoritma dan Kode Program**

1. Algoritma
2. Mulai program.
3. Deklarasikan variabel:

* status sebagai string kosong untuk menyimpan hasil akhir.
* nilai sebagai integer dengan nilai awal 60.

1. Periksa kondisi:

* Jika nilai lebih besar dari 60, maka lanjutkan ke langkah 4a.
* Jika tidak (nilai kurang dari atau sama dengan 60), lanjutkan ke langkah 4b.

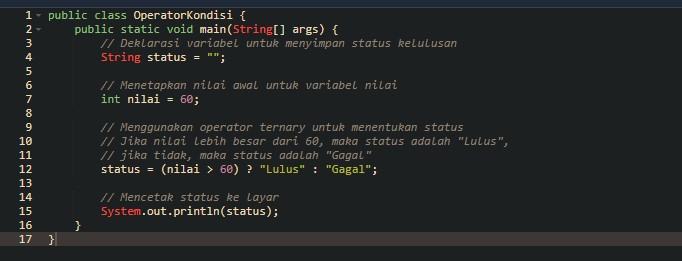
1. Tentukan status:

* 4a: Jika kondisi benar (nilai > 60), maka simpan kata "Lulus" ke dalam variabel status.
* 4b: Jika kondisi salah (nilai <= 60), maka simpan kata "Gagal" ke dalam variabel status.

1. Tampilkan hasil:

* Cetak nilai variabel status ke layar.

1. Selesai program.
2. Kode program dan luaran
3. Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot

1. Analisa luaran yang dihasilkan :  
   Program di atas dirancang untuk menentukan status kelulusan berdasarkan nilai yang diberikan.

**[No.1] Kesimpulan**

**Analisa**

1. Susunlah kesimpulan berdasarkan permasalahan, algoritma, dan kode program!
2. Program yang diberikan bertujuan untuk **menentukan status kelulusan seseorang berdasarkan nilai yang diperoleh**. Algoritma yang digunakan sangat sederhana, yaitu membandingkan nilai yang diberikan dengan nilai ambang batas (dalam kasus ini, 60). Jika nilai lebih besar dari ambang batas, maka status dinyatakan "Lulus", sebaliknya jika sama dengan atau kurang dari ambang batas, maka status dinyatakan "Gagal".
3. Kode program merepresentasikan algoritma ini dengan sangat baik. Operator ternary digunakan untuk mengecek kondisi secara efisien dan menentukan status dalam satu baris. Variabel nilai menyimpan nilai yang akan dievaluasi, sedangkan variabel status menyimpan hasil akhir.

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Ariansyah Putra Ramadhan**  **G1F024080** | **Operator Bitwise: &, |, ^, ~, <<, >>, >>>** | **25 September 2024** |

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

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

**[No.2] Analisis dan Argumentasi**

### . Operasi AND (&)

* **Perhitungan:**
  + a = 10 dalam biner adalah 1010
  + b = 7 dalam biner adalah 0111
  + Operasi AND dilakukan bit per bit:
    - 1 AND 0 = 0
    - 0 AND 1 = 0
    - 1 AND 1 = 1
    - 0 AND 1 = 0
* **Hasil:**
  + hasil = a & b = 0100 (dalam desimal adalah 4)
* **Kesimpulan:**
  + Operasi AND hanya menghasilkan 1 jika kedua bit yang dibandingkan bernilai 1. Dalam kasus ini, hanya bit ketiga dari kedua bilangan yang sama-sama bernilai 1, sehingga hasil akhirnya adalah 4.

### Operasi OR (|)

* **Perhitungan:**
  + a = 10 dalam biner adalah 1010
  + b = 7 dalam biner adalah 0111
  + Operasi OR dilakukan bit per bit:
    - 1 OR 0 = 1
    - 0 OR 1 = 1
    - 1 OR 1 = 1
    - 0 OR 1 = 1
* **Hasil:**
  + hasil = a | b = 1111 (dalam desimal adalah 15)
* **Kesimpulan:**
  + Operasi OR menghasilkan 1 jika salah satu atau kedua bit yang dibandingkan bernilai 1. Dalam kasus ini, semua bit menghasilkan 1, sehingga hasil akhirnya adalah 15.

### . Operasi XOR (^)

* **Perhitungan:**
  + a = 10 dalam biner adalah 1010
  + b = 7 dalam biner adalah 0111
  + Operasi XOR dilakukan bit per bit:
    - 1 XOR 0 = 1
    - 0 XOR 1 = 1
    - 1 XOR 1 = 0
    - 0 XOR 1 = 1
* **Hasil:**

hasil = a ^ b = 1001 (dalam desimal adalah 9)

* **Kesimpulan:**

Operasi XOR menghasilkan 1 jika kedua bit yang dibandingkan berbeda. Dalam kasus ini, bit pertama, kedua, dan keempat berbeda, sehingga menghasilkan 1. Hanya bit ketiga yang sama, sehingga menghasilkan 0.

**[No.3 ] Penyusunan Algoritma dan Kode Program**

1. Algoritma
2. Mulai
3. Inisialisasi:

* Deklarasikan tiga variabel bertipe integer: a, b, dan hasil.
* Berikan nilai awal untuk a dan b.

1. Operasi AND:

* Lakukan operasi bitwise AND antara a dan b.
* Simpan hasilnya dalam variabel hasil.
* Tampilkan hasil ke layar.

1. Operasi OR:

* Lakukan operasi bitwise OR antara a dan b.
* Simpan hasilnya dalam variabel hasil.
* Tampilkan hasil ke layar.

1. Operasi XOR:

* Lakukan operasi bitwise XOR antara a dan b.
* Simpan hasilnya dalam variabel hasil.
* Tampilkan hasil ke layar.

1. Operasi NOT:

* Lakukan operasi bitwise NOT pada a.
* Simpan hasilnya dalam variabel hasil.
* Tampilkan hasil ke layar.

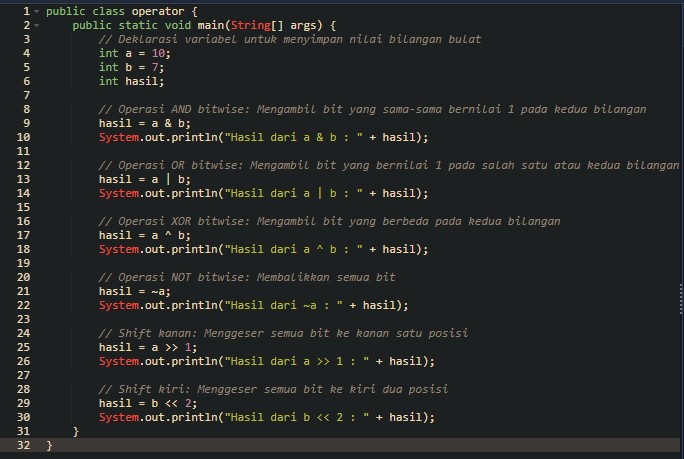
1. Shift Kanan:

* Lakukan shift kanan satu bit pada a.
* Simpan hasilnya dalam variabel hasil.
* Tampilkan hasil ke layar.

1. Shift Kiri:

* Lakukan shift kiri dua bit pada b.
* Simpan hasilnya dalam variabel hasil.
* Tampilkan hasil ke layar.

1. Selesai
2. Kode program dan luaran
3. Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot



1. Analisa luaran yang dihasilkan   
   **Program** di atas melakukan berbagai operasi bitwise pada bilangan bulat a (nilai 10) dan b (nilai 7).
2. **Evaluasi**

* Apa konsekuensi/dampak dari kode program yang dibuat?

**Kode program** di atas secara khusus dirancang untuk **melakukan operasi-operasi bitwise** pada bilangan bulat. Operasi-operasi ini bekerja langsung pada representasi biner dari bilangan, memungkinkan kita untuk melakukan manipulasi tingkat rendah pada data.

Refleksi:

Belajar operator Java adalah langkah awal yang krusial dalam memahami logika pemrograman. Operator-operator ini seperti alat-alat yang kita gunakan untuk memanipulasi data. Pada awalnya, mungkin terasa membingungkan dengan banyaknya jenis operator dan cara kerjanya. Namun, dengan latihan dan pemahaman yang mendalam, saya mulai melihat pola dan kegunaan masing-masing operator.

Pemaknaan Pengetahuan Baru

• Operator sebagai alat manipulasi data: Operator memungkinkan saya untuk melakukan perhitungan, perbandingan, dan manipulasi data lainnya. Dengan memahami operator, saya dapat membangun logika yang lebih kompleks dalam program kita.

Tantangan yang Dihadapi

• Perbedaan antara post-increment dan pre-increment: Ini adalah salah satu tantangan umum bagi pemula. Memahami kapan harus menggunakan post-increment (nilai digunakan dulu, baru diubah) dan pre-increment (nilai diubah dulu, baru digunakan) sangat penting.