

Template Lembar Kerja Individu dan Kelompok

Nama & NPM	Topik:	Tanggal:
DIFA PUTRA PERDANA G1F024072	OPERATOR	16-09-2024
[Nomor Soal] Identifikasi Masalah:		
<ul style="list-style-type: none">• Uraikan permasalahan dan variabel• Rincikan sumber informasi yang relevan (buku / webpage)• Uraikan rancangan solusi yang diusulkan (jika ada).• Analisis susunan solusi, parameter solusi (jika ada).		
[Nomor Soal] Analisis dan Argumentasi		
<ul style="list-style-type: none">• Uraikan rancangan solusi yang diusulkan.• Analisis solusi, kaitkan dengan permasalahan.		
[Nomor Soal] Penyusunan Algoritma dan Kode Program		
<ul style="list-style-type: none">• Rancang desain solusi atau algoritma• Tuliskan kode program dan luaran<ul style="list-style-type: none">• Beri komentar pada kode• Uraikan luaran yang dihasilkan• Screenshot/ Capture potongan kode dan hasil luaran		
[Nomor Soal] Kesimpulan		
<ul style="list-style-type: none">• Analisa<ul style="list-style-type: none">• Susunlah kesimpulan berdasarkan permasalahan, algoritma, dan kode program!• Apakah dasar alasan pengambilan keputusan Anda untuk kasus ini?• Evaluasi<ul style="list-style-type: none">• Apa konsekuensi dari skenario pemrograman ini?• Evaluasi input, proses, dan luaran yang dihasilkan! (jika ada)• Kreasi<ul style="list-style-type: none">• Apakah ada pengetahuan baru yang dikembangkan dan konsep baru sebagai usulan solusi?• Konstruksikan hubungan antara variabel yang berbeda dengan konsep yang anda ketahui! (jika ada)		

JAWABAN!!!

1.1 Tambahkan baris `System.out.println("a + b = " + (a + b));` Ubahlah operator (+) dengan tanda (-, *, /, %)

Untuk memenuhi latihan ini, Kita perlu menambahkan baris-baris kode dengan berbagai operator aritmatika dan mengganti operator tersebut sesuai instruksi.

```
Main.java
1: public class OperatorAritmatika{
2:     public static void main(String[] args) {
3:         // deklarasi nilai
4:         int a = 20, b = 3;
5:
6:         // output nilai a dan b
7:         System.out.println("a: " + a);
8:         System.out.println("b: " + b);
9:
10:        // operator +
11:        System.out.println("a + b = " + (a + b));
12:
13:        // operator -
14:        System.out.println("a - b = " + (a - b));
15:
16:        // operator *
17:        System.out.println("a * b = " + (a * b));
18:
19:        // operator /
20:        System.out.println("a / b = " + (a / b));
21:
22:        // operator %
23:        System.out.println("a % b = " + (a % b));
24:    }
25: }
26:

java -cp /tmp/mMtizlcc9H/OperatorAritmatika
a: 20
b: 3
a + b = 23
a - b = 17
a * b = 60
a / b = 6
a % b = 2

=== Code Execution Successful ===
```

1.2 Analisa perhitungan matematika yang terjadi!

1. Penjumlahan (a + b):

- Ekspresi: 20 + 3
- Hasil: 23
- Output: a + b = 23

2. Pengurangan (a - b):

- Ekspresi: 20 - 3
- Hasil: 17
- Output: a - b = 17

3. Perkalian (a * b):

- Ekspresi: 20 * 3
- Hasil: 60
- Output: a * b = 60

4. Pembagian (a / b):

- Ekspresi: 20 / 3
- Hasil: 6 (pembagian bilangan bulat membuang sisa, sehingga hanya bagian integer yang ditampilkan)
- Output: a / b = 6

5. Sisa Bagi (a % b):

- Ekspresi: 20 % 3
- Hasil: 2 (sisa dari pembagian 20 dengan 3)
- Output: a % b = 2

Penjelasan Tambahan

- **Penjumlahan (+):** Menambahkan dua angka.
- **Pengurangan (-):** Mengurangi angka kedua dari angka pertama.
- **Perkalian (*):** Mengalikan dua angka.
- **Pembagian (/):** Membagi angka pertama dengan angka kedua. Pada operasi pembagian dengan tipe data int, hasilnya adalah bilangan bulat, dengan bagian desimal dibuang.
- **Sisa Bagi (%):** Menampilkan sisa dari pembagian bilangan bulat.

Dengan memahami cara kerja operator aritmatika ini, Kita bisa lebih efektif dalam menulis kode dan memanipulasi nilai dalam pemrograman. Selamat mencoba dan bereksperimen dengan berbagai operasi!

2.1 Bandingkan hasil Contoh 1 dengan Contoh 2!

- **Deklarasi Variabel:**
 - **Kedua contoh** mendeklarasikan variabel a dengan nilai 20 dan b dengan nilai 3.
- **Hasil dari Operator Aritmatika:**
 - **Contoh 1:** Menggunakan operator aritmatika dasar (+, -, *, /, %) di luar perhitungan:
 - Penjumlahan, pengurangan, perkalian, pembagian, dan sisa bagi tidak digunakan secara eksplisit di Contoh 1.
 - Hanya hasil pengurangan yang diperlihatkan dengan hasil $a - b = 17$.
 - **Contoh 2:** Menggunakan operator penugasan (+=, -=, *=, /=, %=) yang mengubah nilai variabel b secara bertahap:
 - b dimodifikasi dengan setiap operator penugasan, yang secara langsung mengubah nilai b dan kemudian dicetak setelah setiap operasi.
 - Hasil akhir dari setiap operasi adalah:
 - Penambahan: 23
 - Pengurangan: 3
 - Perkalian: 60
 - Pembagian: 3
 - Sisa Bagi: 3
- **Hasil Akhir:**
 - **Contoh 1:**
 - Hanya menampilkan hasil dari satu operasi aritmatika, yaitu pengurangan.
 - Hasilnya: $a - b = 17$
 - **Contoh 2:**
 - Menampilkan hasil dari beberapa operasi aritmatika dengan menggunakan operator penugasan yang mengubah nilai b secara bertahap.
 - Hasilnya:
 - Penambahan: 23
 - Pengurangan: 3
 - Perkalian: 60
 - Pembagian: 3
 - Sisa Bagi: 3

Kesimpulan

- **Contoh 1** menampilkan hasil dari satu jenis operasi aritmatika (pengurangan) dengan nilai tetap dari variabel a dan b tanpa memodifikasi nilai variabel tersebut.
- **Contoh 2** menunjukkan penggunaan operator penugasan yang mengubah nilai variabel b melalui berbagai operasi aritmatika. Hasil akhirnya mencerminkan setiap operasi yang dilakukan secara berurutan.

Dengan kata lain, **Contoh 1** fokus pada satu hasil operasi aritmatika sedangkan **Contoh 2** menunjukkan perubahan nilai variabel melalui berbagai operasi aritmatika berturut-turut.

3.1 Ubahlah nilai A = 4 dan B = 4. Analisa perubahan yang terjadi!

```

public class OperatorRelasional {
    public static void main(String[] args) {
        int nilaiA = 4; // Mengubah nilai A menjadi 4
        int nilaiB = 4; // Mengubah nilai B menjadi 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);
    }
}

```

Output yang Diharapkan dengan nilaiA = 4 dan nilaiB = 4

Dengan nilai variabel A dan B keduanya diatur menjadi 4, hasil dari setiap perbandingan adalah:

- **nilaiA > nilaiB:**
 - Memeriksa apakah nilaiA lebih besar dari nilaiB.
 - **Hasil:** false (karena 4 tidak lebih besar dari 4).
- **nilaiA < nilaiB:**
 - Memeriksa apakah nilaiA lebih kecil dari nilaiB.
 - **Hasil:** false (karena 4 tidak lebih kecil dari 4).
- **nilaiA >= nilaiB:**
 - Memeriksa apakah nilaiA lebih besar atau sama dengan nilaiB.
 - **Hasil:** true (karena 4 sama dengan 4, jadi juga memenuhi syarat lebih besar atau sama dengan 4).
- **nilaiA <= nilaiB:**
 - Memeriksa apakah nilaiA lebih kecil atau sama dengan nilaiB.
 - **Hasil:** true (karena 4 sama dengan 4, jadi juga memenuhi syarat lebih kecil atau sama dengan 4).
- **nilaiA == nilaiB:**
 - Memeriksa apakah nilaiA sama dengan nilaiB.
 - **Hasil:** true (karena 4 sama dengan 4).
- **nilaiA != nilaiB:**
 - Memeriksa apakah nilaiA tidak sama dengan nilaiB.
 - **Hasil:** false (karena 4 sama dengan 4, jadi tidak memenuhi syarat berbeda).

```
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

=== Code Execution Successful ===
```

Analisis Perubahan

- **Ketika nilaiA dan nilaiB memiliki nilai yang sama (4):**
 - **Operator >:** false karena 4 tidak lebih besar dari 4.
 - **Operator <:** false karena 4 tidak lebih kecil dari 4.
 - **Operator >=:** true karena 4 sama dengan 4, yang memenuhi syarat lebih besar atau sama dengan 4.
 - **Operator <=:** true karena 4 sama dengan 4, yang memenuhi syarat lebih kecil atau sama dengan 4.
 - **Operator ==:** true karena 4 sama dengan 4.
 - **Operator !=:** false karena 4 sama dengan 4, jadi tidak ada ketidaksamaan.

Kesimpulan

Ketika nilai A dan B sama, hasil dari operator relasional == dan != jelas menunjukkan perbedaan antara kesamaan dan ketidaksamaan, sedangkan operator >, <, >=, dan <= menunjukkan apakah nilai tersebut memenuhi kondisi relatif. Perubahan ini memperjelas bagaimana operator relasional bekerja untuk menentukan hubungan antara dua nilai yang sama.

3.2 Bandingkan bagaimana perbedaan nilai A dan B mempengaruhi nilai luaran!

Perbandingan Hasil

Perubahan Nilai dari Kasus Awal ke Kasus Baru:

- **nilaiA > nilaiB**
 - Kasus Awal: true (karena 12 > 4)
 - Kasus Baru: false (karena 4 tidak lebih besar dari 4)
- **nilaiA < nilaiB**
 - Kasus Awal: false (karena 12 tidak lebih kecil dari 4)
 - Kasus Baru: false (karena 4 tidak lebih kecil dari 4)
- **nilaiA >= nilaiB**
 - Kasus Awal: true (karena 12 lebih besar atau sama dengan 4)
 - Kasus Baru: true (karena 4 sama dengan 4)
- **nilaiA <= nilaiB**
 - Kasus Awal: false (karena 12 tidak lebih kecil atau sama dengan 4)
 - Kasus Baru: true (karena 4 sama dengan 4)
- **nilaiA == nilaiB**
 - Kasus Awal: false (karena 12 tidak sama dengan 4)

- Kasus Baru: true (karena 4 sama dengan 4)
- **nilaiA != nilaiB**
 - Kasus Awal: true (karena 12 tidak sama dengan 4)
 - Kasus Baru: false (karena 4 sama dengan 4)

Kesimpulan

Perubahan nilai A dan B dari 12 dan 4 menjadi 4 dan 4 mempengaruhi hasil dari operator relasional sebagai berikut:

- **Perbandingan lebih besar dan lebih kecil (>, <):** Jika kedua nilai sama, hasil perbandingan ini akan selalu false.
- **Perbandingan lebih besar atau sama dengan dan lebih kecil atau sama dengan (>=, <=):** Jika kedua nilai sama, hasil perbandingan ini akan selalu true.
- **Perbandingan sama dengan (==):** Jika kedua nilai sama, hasil perbandingan ini akan selalu true.
- **Perbandingan tidak sama dengan (!=):** Jika kedua nilai sama, hasil perbandingan ini akan selalu false.

Operator relasional membantu dalam menentukan hubungan antara dua nilai, dan perubahan nilai dapat dengan jelas mempengaruhi hasil dari perbandingan ini.

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

```

1 public class operator {
2     public static void main(String[] args) {
3         int a = 10;
4         System.out.println("# Post 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
10        System.out.println();
11
12        int b = 10;
13        System.out.println("# Pre Increment #");
14        System.out.println("=====");
15        System.out.println("Isi variabel b: " + b);
16        System.out.println("Isi variabel b: " ++b);
17        System.out.println("Isi variabel b: " + b);
18
19        System.out.println();
20
21        int c = 10;
22        System.out.println("# Post Decrement #");
23        System.out.println("=====");
24        System.out.println("Isi variabel c: " + c);
25        System.out.println("Isi variabel c: " + c--);
26        System.out.println("Isi variabel c: " + c);
27
28        System.out.println();
29
30        int d = 10;
31        System.out.println("# Pre Decrement #");
32        System.out.println("=====");
33        System.out.println("Isi variabel d: " + d);
34        System.out.println("Isi variabel d: " + --d);
35        System.out.println("Isi variabel d: " + d);
36    }
37 }

```

Penjelasan Operator

- **Post Increment (a++):**
 - **Proses:** Nilai variabel a digunakan terlebih dahulu, lalu variabel a dinaikkan satu.
 - **Hasil:**
 - **Sebelum a++:** a = 10
 - **Cetak a++:** 10 (nilai a sebelum increment)
 - **Nilai a setelah a++:** 11 (nilai a setelah increment)

- **Pre Increment (++b):**
 - **Proses:** Variabel b dinaikkan satu terlebih dahulu, lalu nilai baru dari b digunakan.
 - **Hasil:**
 - **Sebelum ++b:** b = 10
 - **Nilai ++b:** 11 (nilai b setelah increment)
 - **Nilai b setelah ++b:** 11 (nilai b setelah increment)
- **Post Decrement (c--):**
 - **Proses:** Nilai variabel c digunakan terlebih dahulu, lalu variabel c dikurangi satu.
 - **Hasil:**
 - **Sebelum c--:** c = 10
 - **Cetak c--:** 10 (nilai c sebelum decrement)
 - **Nilai c setelah c--:** 9 (nilai c setelah decrement)
- **Pre Decrement (--d):**
 - **Proses:** Variabel d dikurangi satu terlebih dahulu, lalu nilai baru dari d digunakan.
 - **Hasil:**
 - **Sebelum --d:** d = 10
 - **Nilai --d:** 9 (nilai d setelah decrement)
 - **Nilai d setelah --d:** 9 (nilai d setelah decrement)

Output kode

```
# 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

=== Code Execution Successful ===
```

Analisis

- **Post Increment (a++) dan Post Decrement (c--):**
 - **Perbedaan Utama:** Operator ini mengembalikan nilai variabel sebelum perubahan dilakukan. Setelah itu, nilai variabel diubah (dinaikkan atau diturunkan).
- **Pre Increment (++b) dan Pre Decrement (--d):**
 - **Perbedaan Utama:** Operator ini mengubah nilai variabel terlebih dahulu, kemudian mengembalikan nilai variabel yang telah diubah.

Dengan memanfaatkan kedua bentuk operator ini, Anda dapat mengontrol kapan perubahan nilai terjadi dalam ekspresi dan bagaimana hasilnya dipengaruhi dalam kode .

5.1 Tambahkan baris kode untuk memeriksa a || b.

```
1 public class OperatorLogika {
2     public static void main (String [] args) {
3         boolean a = true;
4         boolean b = false;
5         boolean c;
6
7         c = a && b;
8         System.out.println("true && false = " + c);
9
10        // Tambahkan baris berikut
11        c = a || b;
12        System.out.println("true || false = " + c);
13    }
14 }
15
```

```
java -cp /tmp/mX8BshzVgZ/OperatorLogika
true && false = false
true || false = true

=== Code Execution Successful ===
```

1. **a && b** menghasilkan false karena salah satu operand (b) adalah false.
2. **a || b** menghasilkan true karena salah satu operand (a) adalah true.

5.2 Ubahlah nilai a = false dan b = false. Analisa perubahan dan perbedaan boolean yang terjadi!

```
1 public class OperatorLogika {
2     public static void main (String [] args) {
3         boolean a = false;
4         boolean b = false;
5         boolean c;
6
7         c = a && b;
8         System.out.println("false && false = " + c);
9
10        c = a || b;
11        System.out.println("false || false = " + c);
12    }
13 }
14
```

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

=== Code Execution Successful ===
```

1. **a && b** menghasilkan false karena kedua operand adalah false.
2. **a || b** juga menghasilkan false karena kedua operand adalah false.

5.3 Apabila diketahui pernyataan a || b && a || !b. Uraikan urutan logika yang akan dikerjakan! Analisa luaran true atau false dari pernyataan tersebut!

- Operator && (AND) memiliki prioritas lebih tinggi daripada || (OR). Jadi, b && a dievaluasi terlebih dahulu.
- Evaluasi !b (NOT) terjadi setelah evaluasi b && a.
- Kemudian hasil dari b && a dan !b dioperasikan dengan || (OR).

Mari kita contohkan dengan nilai:

- Misalkan a = true, b = false.

Langkah Evaluasi:

- **Evaluasi b && a:**
 - false && true = false.
- **Evaluasi !b:**
 - !false = true.
- **Gabungkan hasil dengan operator ||:**
 - a || (hasil dari b && a) = true || false = true.
 - (hasil dari b && a) || !b = false || true = true.

Hasil Akhir:

- Dengan a = true dan b = false, pernyataan a || b && a || !b menghasilkan true.

Latihan 6

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

```
1 public class OperatorKondisi {
2     public static void main(String[] args) {
3         String status = "";
4         int nilai = 60; // Mengubah nilai menjadi 60
5         status = (nilai > 60) ? "Lulus" : "Gagal";
6         System.out.println(status);
7     }
8 }
9
```

```
java -cp /tmp/0tVHnK0u0g/OperatorKondisi
Gagal

=== Code Execution Successful ===
```

- **Ekspresi Ternary:**

status = (nilai > 60) ? "Lulus" : "Gagal";

- **Kondisi:** nilai > 60
- **Nilai jika benar:** "Lulus"
- **Nilai jika salah:** "Gagal"
- **Evaluasi Kondisi:**
 - Dengan nilai yang diubah menjadi 60, ekspresi nilai > 60 dievaluasi sebagai false karena 60 tidak lebih besar dari 60. Artinya, kondisi nilai > 60 tidak terpenuhi.
- **Hasil dari Ekspresi Ternary:**
 - Karena kondisi nilai > 60 adalah false, maka hasil dari ekspresi ternary adalah nilai setelah tanda titik dua : yaitu "Gagal".

Kesimpulan:

- Ketika nilai diubah menjadi 60, kondisi nilai > 60 tidak terpenuhi, sehingga ekspresi ternary mengembalikan "Gagal". Ini menunjukkan bahwa operator ternary mengevaluasi kondisi dan memilih nilai yang sesuai berdasarkan hasil evaluasi tersebut.

Operator ternary sangat berguna untuk menyederhanakan kode saat hanya ada dua kemungkinan hasil (true/false) untuk kondisi yang diberikan, terutama ketika setiap cabang hanya melibatkan satu pernyataan.

Latihan 7

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

1. Operasi &

Biner:

- a = 10 = 1010 (biner)
- b = 7 = 0111 (biner)

Hasil Biner: 0010

Hasil Desimal: 2

2. Operasi |

Biner:

- a = 10 = 1010 (biner)
- b = 7 = 0111 (biner)

Hasil Biner: 1111

Hasil Desimal: 15

3. Operasi <<

Biner:

- $b = 7 = 0111$ (biner)

Hasil Biner: 11100

Hasil Desimal: 28

Simpulan

- **Operasi &:**
 - Perhitungan: $1010 \& 0111$ menghasilkan 0010 yang setara dengan desimal 2.
 - Artinya, hanya bit-bit yang setara (1 di kedua operand) yang dihasilkan sebagai 1 di hasil.
- **Operasi |:**
 - Perhitungan: $1010 | 0111$ menghasilkan 1111 yang setara dengan desimal 15.
 - Artinya, bit-bit yang salah satu atau kedua operandnya 1 akan dihasilkan sebagai 1 di hasil.
- **Operasi <<:**
 - Perhitungan: $0111 \ll 2$ menghasilkan 11100 yang setara dengan desimal 28.
 - Artinya, angka biner digeser ke kiri sebanyak 2 posisi, menambah dua 0 di sebelah kanan.

Operasi bitwise sering digunakan dalam pemrograman sistem dan aplikasi yang memerlukan manipulasi bit yang efisien.

Contoh Jawaban:

Nama & NPM	Topik:	Tanggal:
Putri G1A000001	Tipe Data	26 Agustus 2022

[No. 1] Identifikasi Masalah:

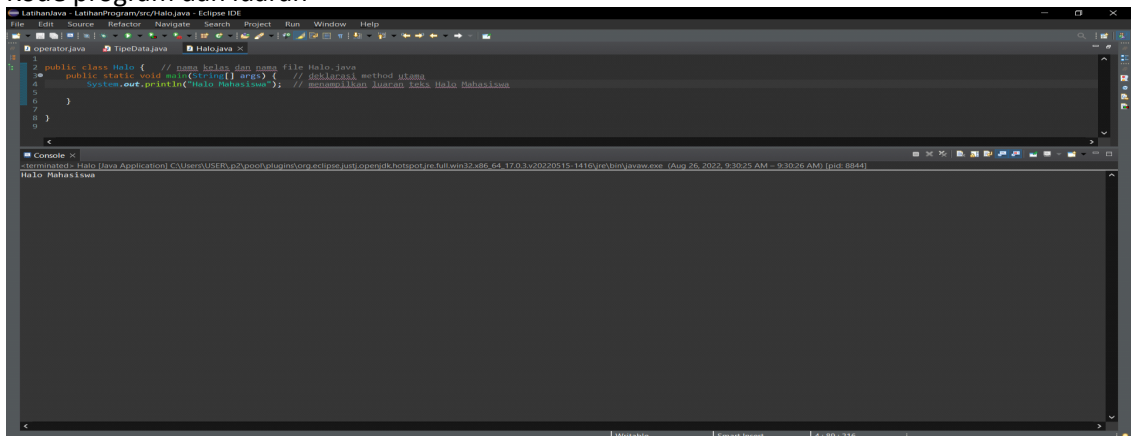
- Uraikan permasalahan dan variabel
Contoh:
Tuliskan kembali soal:
Pada soal masih ada pesan kesalahan _____
Atau
Diketahui dari soal : variabel _____

[No.1] Analisis dan Argumentasi

- Saya mengusulkan permasalahan ini dapat diatasi dengan cara _____
- Alasan solusi ini karena _____
- Perbaiki kode program dengan cara _____

[No.1] Penyusunan Algoritma dan Kode Program

- Algoritma
Algoritma adalah langkah-langkah penyelesaian masalah.
Misalkan algoritma memasak mi instan:
 - Masak air
 - Buka bungkus
 - Masukkan mie
 - Masukkan bumbu
 - Hasilnya mie matang, taruh di piring
 - Mie siap disantap.
- Kode program dan luaran



The screenshot shows the Eclipse IDE interface. The main editor window displays the following Java code:

```
1 // nama kelas dan nama file Halo.java
2 public class Halo {
3     public static void main(String[] args) { // deklarasi method utama
4         System.out.println("Halo Mahasiswa"); // menampilkan output teks Halo Mahasiswa
5     }
6 }
```

The console window at the bottom shows the output of the program:

```
Halo Mahasiswa
```

- Screenshot/ Capture potongan kode dan hasil luaran
Beri komentar pada kode yang di Screenshot

- Analisa luaran yang dihasilkan

Contoh:

Luaran sudah sesuai dengan program yang disusun.

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

(Tuliskan penjelasan dari program yang dibuat, apakah kode dan luaran sudah benar?)

[No.1] Kesimpulan

(PILIH SALAH SATU ANDA INGIN MEMBAHAS DENGAN CARA ANALISA/ EVALUASI / KREASI)

- **Analisa**

- Susunlah kesimpulan berdasarkan permasalahan, algoritma, dan kode program!
- Apakah dasar alasan pengambilan keputusan Anda untuk kasus ini?

Contoh jawaban Analisa:

Pada program itu saya menggunakan bentuk kelas public karena _____

Perbaiki program dengan menambahkan _____ karena struktur java mengharuskan _____

(penjelasan analisa mengulangi kembali materi yang sudah diberikan)

(penjelasan mengkaitkan dengan materi yang ada)

- **Evaluasi**

- Apa konsekuensi/dampak dari kode program yang dibuat?
- Evaluasi input program, proses perhitungan, dan luaran yang dihasilkan! (jika ada)

Contoh jawaban Evaluasi:

Pada program itu saya mengkonversi bentuk kelas public karena _____

Setelah dikonversi, saya mengevaluasi bahwa tipe data _____ lebih baik digunakan untuk bentuk data seperti _____

(penjelasan evaluasi mengulangi kembali materi yang sudah diberikan dan mengetahui kekurangan dari materi hasil eksperimen)

(misal tipe data ____ ternyata tidak dapat dipakai untuk _____ karena _____)

- **Kreasi**

- Apakah ada pengetahuan baru yang dikembangkan dan konsep baru sebagai usulan solusi?
- Susunlah hubungan antara variabel yang berbeda dengan konsep yang anda ketahui! (jika ada)

Contoh jawaban Kreasi:

Pada program itu saya mengkonversi bentuk kelas public karena _____

Setelah dikonversi, saya mengevaluasi bahwa tipe data _____ lebih baik digunakan untuk bentuk data seperti _____

Saya telah mencoba mengubah menjadi kelas private dan protected, ternyata menghasilkan _____

Berarti kelas private dan protected mempengaruhi _____

(sampaikan temuan Anda yang baru diketahui, eksperimen baru diluar materi yang diberikan)

(penjelasan kreasi mengulangi kembali materi yang sudah diberikan dan

menambahkan pengetahuan baru dari pengalaman dari hasil eksperimen)

Lanjutkan ke soal nomor 2 – 3 – ... – dan seterusnya

Refleksi

(Tuliskan singkat tentang pengalaman belajar, pemaknaan pengetahuan yang baru, tantangan yang dihadapi pada minggu tersebut. Ringkasan singkat dari semua soal, bukan per soal)