

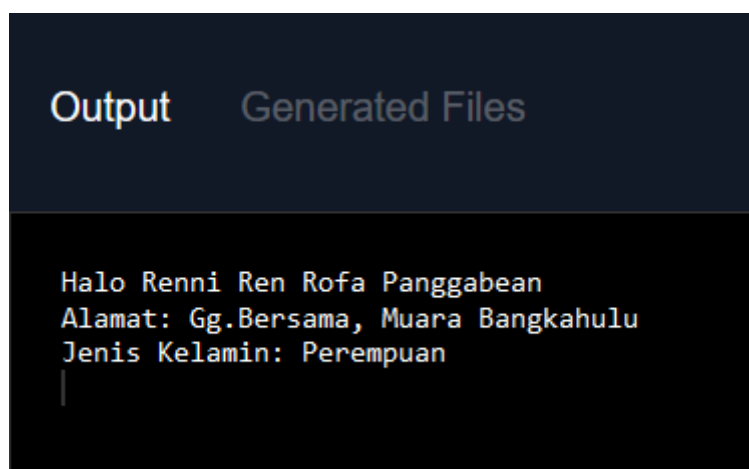
Template Lembar Kerja Individu

Nama & NPM	Topik:	Tanggal:
RENNI REN ROFA.PGB G1F021002	Pengenalan Java dan Tipe Data	30 Agustus 2024
[1.] Identifikasi Masalah:		
<p>1) Permasalahan: Kode yang diberikan tidak dapat dikompilasi dan dijalankan karena terdapat kesalahan sintaksis.</p> <p>Variabel : Tidak adanya variabel yang dideklarasikan dalam kode tersebut.</p> <p>Soal :</p> <ol style="list-style-type: none">1. Evaluasi penyebab kesalahan terjadi dan perbaiki agar program dapat berjalan.2. Ubah teks yang ditampilkan program menjadi nama lengkap Anda.3. Tambahkan baris <code>System.out.println("");</code> untuk diisi dengan data alamat, dan jenis kelamin. <p>2) Rincikan sumber informasi yang relevan : <i>WebPage: <u>Oracle Java Documentation</u> - Sumber daya resmi untuk memahami sintaks dan struktur program Java</i></p>		
[1.] Analisis dan Argumentasi		
<p>1) Saya mengusulkan solusi dengan memperbaiki sintaks kode dengan menutup tanda kutip pada string, kemudian mengubah metode 'main' dari 'private' menjadi 'public'.</p> <p>2) Alasan solusi ini karena ada beberapa Kesalahan pada kode nya yaitu :</p> <ul style="list-style-type: none">• Tanda Kutip yang Hilang• Deklarasi Metode main yang Tidak Tepat <p>Dengan memperbaiki kesalahan sintaks, kode program akan dapat berjalan dengan baik dan menghasilkan output yang sesuai dengan yang diharapkan.</p>		
[1.] Penyusunan Algoritma dan Kode Program		
<p>1) Algoritma :</p> <ul style="list-style-type: none">• Deklarasikan kelas KelasKu.• Buat metode main dengan akses publik.• Tampilkan pesan ke konsol menggunakan <code>System.out.println</code>.• Tambahkan baris untuk alamat dan jenis kelamin. <p>2) Tuliskan kode program dan luaran</p> <p>a) Luaran yang dihasilkan:</p> <p>Halo Renni Ren Rofa Panggabean Alamat: Gg.Bersama, Muara Bangkahulu Jenis Kelamin: Perempuan</p> <p>b) Capture Kode Program dan luaran:</p>		



```
1
2
3
4
5 public class KelasKu {
6     public static void main(String[] args) {
7         // Menampilkan pesan selamat datang
8         System.out.println("Halo Renni Ren Rofa Panggabean"); // Ganti "Halo mahasiswa UNIB" dengan nama Lengkap Anda
9         System.out.println("Alamat: Gg.Bersama, Muara Bangkahulu"); // Ganti dengan alamat Anda
10        System.out.println("Jenis Kelamin: Perempuan"); // Ganti dengan jenis kelamin Anda
11    }
12 }
13
```

Gambar 1.1 Kode program



```
Output    Generated Files

Halo Renni Ren Rofa Panggabean
Alamat: Gg.Bersama, Muara Bangkahulu
Jenis Kelamin: Perempuan
|
```

Gambar 1.2 Luaran

[1.] Kesimpulan

1) Analisa

- Kesimpulan berdasarkan permasalahan: Bahwasannya kesalahan sintaks dapat menyebabkan program tidak dapat dijalankan. Dengan memperbaiki kesalahan tersebut, program dapat berfungsi sesuai dengan yang diharapkan.
- Dasar alasan pengambilan keputusan: Pentingnya mematuhi aturan sintaks dalam pemrograman Java untuk menghindari kesalahan kompilasi.

2) Evaluasi

- Konsekuensi dari skenario pemrograman yaitu : jika kesalahan tidak diperbaiki, program tidak dapat dieksekusi.
- Evaluasi input, proses, dan luaran yang dihasilkan : input yang tepat diperlukan untuk menghasilkan output yang diinginkan.

3) Kreasi

- Pengetahuan baru yang dikembangkan yaitu pemahaman yang lebih baik tentang sintaks Java dan pentingnya penanganan kesalahan.

Nama & NPM	Topik:	Tanggal:
RENNI REN ROFA.PGB G1F021002	Pengenalan Java dan Tipe Data	30 Agustus 2024
[2.] Identifikasi Masalah:		
<p>1) Permasalahan: Menentukan tipe data yang tepat untuk berbagai jenis data yang diberikan Data contoh 2 :</p> <ol style="list-style-type: none"> 1. 5 2. 'L' 3. "mobil" 4. 5.0 5. 5.0f 6. -5 <p>2) Rincikan sumber informasi yang relevan : WebPage: <u>Oracle Java Documentation</u> - Sumber daya resmi untuk memahami sintaks dan struktur program Java</p> <p>3) Uraikan Rancangan Solusi yang Diusulkan:</p> <ul style="list-style-type: none"> • Identifikasi setiap tipe data yang sesuai dengan nilai yang diberikan. • Gunakan pengetahuan tentang tipe data primitif dan non-primitif di Java untuk menentukan tipe data yang paling efisien. 		
[2.] Analisis dan Argumentasi		
<p>1) Saya mengusulkan solusi dengan pemilihan tipe data yang sesuai berdasarkan nilai-nilai yang diberikan, dengan mempertimbangkan efisiensi memori dan kebutuhan presisi.</p> <p>2) Analisis Solusi, Kaitkan dengan permasalahan :</p> <p>Solusi yang diusulkan sesuai dengan karakteristik tipe data yang tepat untuk setiap nilai. Hal ini memastikan program bekerja secara efisien dan tanpa kesalahan tipe data.</p>		
[2.] Penyelesaian		
<p>1) Rekomendasi Tipe Data Yang Tepat : Dari data yang diberikan:</p> <ul style="list-style-type: none"> • 5: Tipe data yang tepat adalah <i>int</i> karena merupakan bilangan bulat. • 'L': Tipe data yang tepat adalah <i>char</i> karena merupakan satu karakter. • "mobil": Tipe data yang tepat adalah <i>String</i> karena merupakan rangkaian karakter (teks). • 5.0: Tipe data yang tepat adalah <i>double</i> karena merupakan bilangan pecahan desimal. • 5.0f: Tipe data yang tepat adalah <i>float</i> karena merupakan bilangan pecahan desimal yang diberi penanda f. • -5: Tipe data yang tepat adalah <i>int</i> karena merupakan bilangan bulat negatif. <p>2) Karakteristik Penggunaan Setiap Tipe Data:</p> <ul style="list-style-type: none"> • int: Digunakan untuk menyimpan bilangan bulat (baik positif, negatif, maupun nol). Tipe data ini biasanya berukuran 4 byte (32-bit) di Java dan memiliki rentang nilai dari -2,147,483,648 hingga 2,147,483,647. • char: Digunakan untuk menyimpan karakter tunggal. Tipe data ini berukuran 2 byte (16-bit) karena Java menggunakan Unicode, yang memungkinkan representasi karakter dari berbagai bahasa di dunia. 		

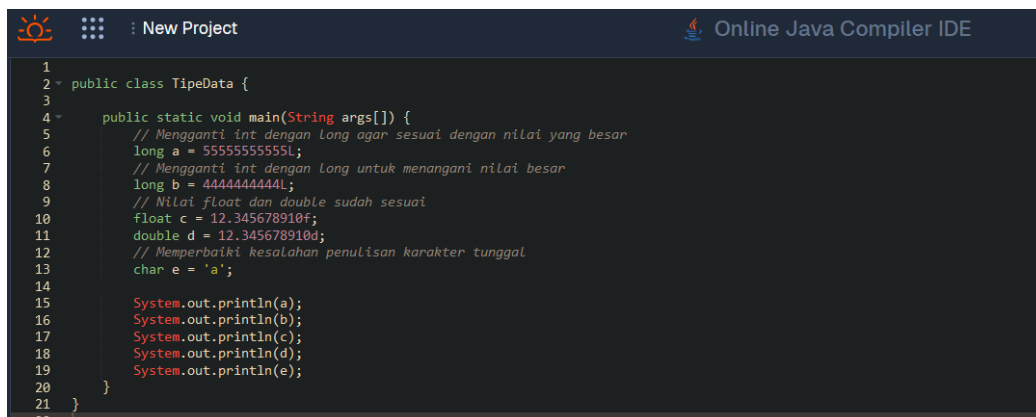
- **String**: Digunakan untuk menyimpan rangkaian karakter atau teks. String di Java bukan tipe data primitif, tetapi merupakan kelas yang digunakan untuk mengelola teks. String bisa menyimpan teks dengan panjang yang cukup besar tergantung pada memori yang tersedia.
- **double**: Digunakan untuk menyimpan bilangan desimal dengan presisi yang lebih tinggi. Tipe data ini berukuran 8 byte (64-bit) dan biasanya digunakan untuk perhitungan yang membutuhkan presisi tinggi.
- **float**: Digunakan untuk menyimpan bilangan desimal dengan presisi yang lebih rendah dibandingkan double. Tipe data ini berukuran 4 byte (32-bit) dan digunakan ketika presisi tinggi tidak terlalu dibutuhkan untuk menghemat memori.
- **int**: Sama seperti pada poin pertama, digunakan untuk bilangan bulat, termasuk nilai negatif.

[2.] Kesimpulan

- 1) Analisa
Tipe data yang dipilih untuk setiap nilai sudah sesuai dengan karakteristik dan kebutuhan presisi serta efisiensi memori. Keputusan ini diambil berdasarkan pengetahuan tentang tipe data dalam Java.
- 2) Evaluasi
Konsekuensi dari pemrograman ini adalah program akan berjalan dengan efisien dan benar karena tipe data yang tepat digunakan.
- 3) Kreasi
Pengetahuan baru: Penggunaan tipe data yang lebih tepat untuk aplikasi spesifik.

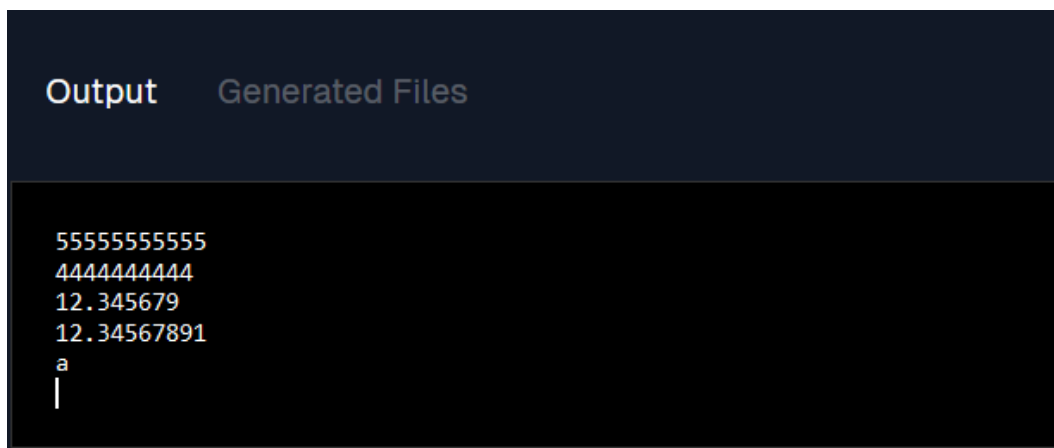
Nama & NPM	Topik:	Tanggal:
RENNI REN ROFA.PGB G1F021002	Pengenalan Java dan Tipe Data	30 Agustus 2024
[3.] Identifikasi Masalah:		
<p>1) Permasalahan: Kode yang diberikan tidak dapat dikompilasi karena terdapat kesalahan penulisan tipe data.</p> <p>Variabel :</p> <ul style="list-style-type: none"> • int a: Menyimpan bilangan bulat besar. • byte b: Menyimpan bilangan bulat kecil. • float c: Menyimpan bilangan desimal. • double d: Menyimpan bilangan desimal dengan presisi lebih tinggi. • char e: Menyimpan karakter tunggal. <p>Soal :</p> <p>3.1 Evaluasi penyebab kesalahan pada Contoh 3!</p> <p>3.2. Rekomendasikan tipe data yang sesuai untuk data tersebut</p> <p>2) Rincikan sumber informasi yang relevan : <i>WebPage: Oracle Java Documentation - Sumber daya resmi untuk memahami sintaks dan struktur program Java</i></p>		
[3.] Analisis dan Argumentasi		
<p>1) Saya mengusulkan solusi dengan mengubah 'int a' menjadi 'long a'. Ubah 'byte b' menjadi 'int b'. serta gunakan tanda kutip tunggal untuk 'char e' dengan satu karakter tunggal.</p> <p>2) Alasan solusi diatas karena dengan mengganti int dengan long, kita bisa menyimpan bilangan lebih besar. Tipe data int memiliki rentang dari -2^{31} hingga $2^{31}-1$, sementara long memiliki rentang yang jauh lebih besar. Byte hanya dapat menyimpan nilai dari -128 hingga 127, sehingga harus diganti dengan int. Tipe char hanya bisa menyimpan satu karakter, jadi 'abc' akan menyebabkan kesalahan.</p> <p>3) Parameter Solusi:</p> <ul style="list-style-type: none"> • Gunakan tipe data yang sesuai dengan rentang nilai. • Memastikan penulisan literal karakter tunggal. 		
[3.] Penyusunan Algoritma dan Kode Program		
<p>1) Algoritma :</p> <ul style="list-style-type: none"> • Identifikasi nilai yang tidak sesuai dengan tipe data • Ganti tipe data yang sesuai dengan nilai yang diberikan. • Patikan literal 'cahar' hanya mengandung satu karakter. <p>2) Tuliskan kode program dan luaran</p> <p>a) Luaran yang dihasilkan:</p> <pre>5555555555 4444444444 12.345679 12.34567891 a</pre>		

b) Capture Kode Program dan luaran:



```
1 public class TipeData {
2
3
4     public static void main(String args[]) {
5         // Mengganti int dengan Long agar sesuai dengan nilai yang besar
6         long a = 5555555555L;
7         // Mengganti int dengan Long untuk menangani nilai besar
8         long b = 4444444444L;
9         // Nilai float dan double sudah sesuai
10        float c = 12.345678910f;
11        double d = 12.345678910d;
12        // Memperbaiki kesalahan penulisan karakter tunggal
13        char e = 'a';
14
15        System.out.println(a);
16        System.out.println(b);
17        System.out.println(c);
18        System.out.println(d);
19        System.out.println(e);
20    }
21 }
22 }
```

Gambar 3.1 Kode program



```
5555555555
4444444444
12.345679
12.34567891
a
|
```

Gambar 3.2 Luaran

[3.] Kesimpulan

- 1) Analisa
Kesalahan yang terjadi karena penggunaan tipe data yang tidak sesuai dengan nilai yang diberikan dan penulisan literal char yang salah.
- 2) Evaluasi
 - Input dan tipe data harus sesuai untuk menghindari kesalahan kompilasi.
 - Penggunaan tipe data yang salah dapat menyebabkan program gagal untuk dikompilasi dan dijalankan
- 3) Kreasi
 - a) Pengetahuan Baru: Pentingnya memahami batasan rentang tipe data dalam Java.
 - b) Konstruksi Hubungan: Hubungan antara tipe data dengan rentang nilainya sangat penting dalam pemrograman untuk memastikan program dapat berjalan dengan baik tanpa kesalahan.

Nama & NPM	Topik:	Tanggal:
RENNI REN ROFA.PGB G1F021002	Pengenalan Java dan Tipe Data	30 Agustus 2024

[4.] Identifikasi Masalah:

- 1) **Permasalahan:** Memahami bagaimana berbagai tipe data dapat dikonversi satu sama lain dan dampaknya terhadap data yang dikonversi.

Variabel : 'int a', 'double b', 'byte x'.

Soal :

1. Rekomendasikan konversi tipe data pada Latihan 2 ke bentuk tipe data lain yang kompatibel.
2. Simpulkan alasan jenis konversi tipe data tersebut!

- 2) **Sumber informasi yang relevan :**

Buku pemrograman Java atau halaman web tentang konversi tipe data di Java, seperti dokumentasi resmi Oracle.

[4.] Analisis dan Argumentasi

- 1) **Saya mengusulkan solusi** dengan merekomendasikan konversi tipe data yang aman dan mempertimbangkan kemungkinan hilangnya data, khususnya saat mengonversi tipe data yang lebih besar ke yang lebih kecil..

- 2) **Dengan rekomendasi yang ada analisislah** apakah data tetap utuh atau mengalami overflow/underflow selama konversi dan evaluasi hasil keluaran apakah sesuai dengan yang diharapkan berdasarkan tipe data

- 3) **Penyelesaian :**

Rekomendasi Konversi Tipe Data:

1. **5**

- Tipe data awal: int
- Rekomendasi konversi:
 - byte, short, long, float, double

2. **'L'**

- Tipe data awal: char
- Rekomendasi konversi:
 - int, long, float, double

3. **"mobil"**

- Tipe data awal: String
- Rekomendasi konversi:
 - Tidak ada konversi langsung ke tipe data primitif.

4. **5.0**

- Tipe data awal: double
- Rekomendasi konversi:
 - float, int, long

5. **5.0f**

- Tipe data awal: float
- Rekomendasi konversi:

<div style="text-align: right; margin-right: 20px;">▪ double, int, long</div> <p>6. -5</p> <ul style="list-style-type: none"> ○ Tipe data awal: int ○ Rekomendasi konversi: <ul style="list-style-type: none"> ▪ byte, short, long, float, double <p>4) Alasan Jenis Konversi :</p> <ul style="list-style-type: none"> • Konversi dari tipe data yang lebih besar ke yang lebih kecil (misalnya, dari double ke int): Ini dilakukan ketika presisi tidak diperlukan, tetapi rentang nilai yang lebih kecil cukup untuk aplikasi. Misalnya, ketika nilai desimal diabaikan. • Konversi dari tipe data yang lebih kecil ke yang lebih besar (misalnya, dari int ke double): Ini dilakukan untuk meningkatkan rentang dan presisi penyimpanan data tanpa kehilangan informasi. • Konversi antara char dan tipe numerik (int, long, dll.): Konversi ini mengubah karakter menjadi nilai ASCII-nya yang setara dalam tipe numerik.
<p>[4.] Kesimpulan</p>
<p>1) Analisa Konversi tipe data harus dilakukan dengan hati-hati untuk menghindari hilangnya informasi yang penting.</p> <p>2) Evaluasi</p> <ul style="list-style-type: none"> • Input yang lebih besar dari batas tipe data target akan menyebabkan overflow atau underflow. • Proses konversi harus mempertimbangkan jenis operasi yang akan dilakukan setelah konversi. <p>3) Kreasi Sebuah program dapat dirancang untuk mengidentifikasi dan memberi peringatan ketika konversi tipe data mungkin menyebabkan hilangnya data atau ketidaksesuaian hasil.</p>

Refleksi

Minggu ini, belajar tentang konversi tipe data dalam pemrograman Java memberikan pemahaman lebih dalam mengenai cara kerja tipe data primitif dan konversinya. Pemahaman tentang bagaimana konversi bisa mengubah nilai, seperti overflow atau hilangnya bagian desimal, sangat penting untuk menghindari kesalahan dalam program. Selain itu, memahami batasan dan karakteristik dari tiap tipe data membantu dalam pengambilan keputusan yang lebih baik dalam pemrograman.