**Template Lembar Kerja Individu dan Kelompok**

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
|  |  |  |
| **[Nomor Soal] Identifikasi Masalah:** | | |
| 1. Uraikan permasalahan dan variabel 2. Rincikan sumber informasi yang relevan (buku / webpage) 3. Uraikan rancangan solusi yang diusulkan (jika ada). 4. Analisis susunan solusi, parameter solusi (jika ada). | | |
| **[Nomor Soal] Analisis dan Argumentasi** | | |
| 1. Uraikan rancangan solusi yang diusulkan. 2. Analisis solusi, kaitkan dengan permasalahan. | | |
| **[Nomor Soal] Penyusunan Algoritma dan Kode Program** | | |
| 1. Rancang desain solusi atau algoritma 2. Tuliskan kode program dan luaran 3. Beri komentar pada kode 4. Uraikan luaran yang dihasilkan 5. Screenshot/ Capture potongan kode dan hasil luaran | | |
| **[Nomor Soal] Kesimpulan** | | |
| 1. Analisa 2. Susunlah kesimpulan berdasarkan permasalahan, algoritma, dan kode program! 3. Apakah dasar alasan pengambilan keputusan Anda untuk kasus ini? 4. Evaluasi 5. Apa konsekuensi dari skenario pemprograman ini? 6. Evaluasi input, proses, dan luaran yang dihasilkan! (jika ada) 7. Kreasi 8. Apakah ada pengetahuan baru yang dikembangkan dan konsep baru sebagai usulan solusi? 9. Konstruksikan hubungan antara variabel yang berbeda dengan konsep yang anda ketahui! (jika ada) | | |
|  | | |

Jawaban

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Adel januarti saputri**  **G1F024010** | **KELAS JAVA** | **18 SEPTEMBER 2024** |

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

1. **Nama kelas dan konstruktor tidak konsisten**:
   * Nama kelas adalah Manusia, tetapi konstruktor yang dideklarasikan bernama Manusia1. Konstruktor harus menggunakan nama yang sama dengan nama kelas, dalam hal ini seharusnya Manusia, bukan Manusia1.
2. **Parameter konstruktor salah**:
   * Konstruktor Manusia1 hanya menerima satu parameter (String nama), tetapi pada saat pembuatan objek (new Manusia1("Putri", "hitam");), Anda memberikan dua argumen ("Putri" dan "hitam"). Anda perlu memperbaiki konstruktornya agar sesuai dengan jumlah argumen yang digunakan.
3. **Atribut rambut tidak diinisialisasi**:
   * Pada konstruktor, Anda hanya mencetak nilai rambut, tetapi tidak pernah menginisialisasi nilai tersebut.
4. **Kelas Manusia1 tidak ada**:
   * Anda mencoba membuat objek dari kelas Manusia1, tetapi kelas tersebut tidak ada. Kode ini seharusnya menggunakan nama kelas yang benar, yaitu Manusia.

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

Nama Kelas dan Konstruktor Tidak Konsisten karena kode mendeklarasikan kelas dengan nama Manusia, namun konstruktor yang didefinisikan memiliki nama Manusia1. Dalam Java, konstruktor harus memiliki nama yang sama dengan kelas. Ini adalah aturan dasar dalam Java untuk membedakan konstruktor dari metode lain Nama Konstruktor Harus Sama dengan Nama Kelas: Dalam Java, nama konstruktor harus identik dengan nama kelas. Karena kelasnya bernama Manusia, maka konstruktor juga harus dinamai Manusia, bukan Manusia1. Jika kita menggunakan nama yang berbeda untuk konstruktor, program tidak akan mengenalinya sebagai konstruktor dan akan menyebabkan kesalahan kompilasi

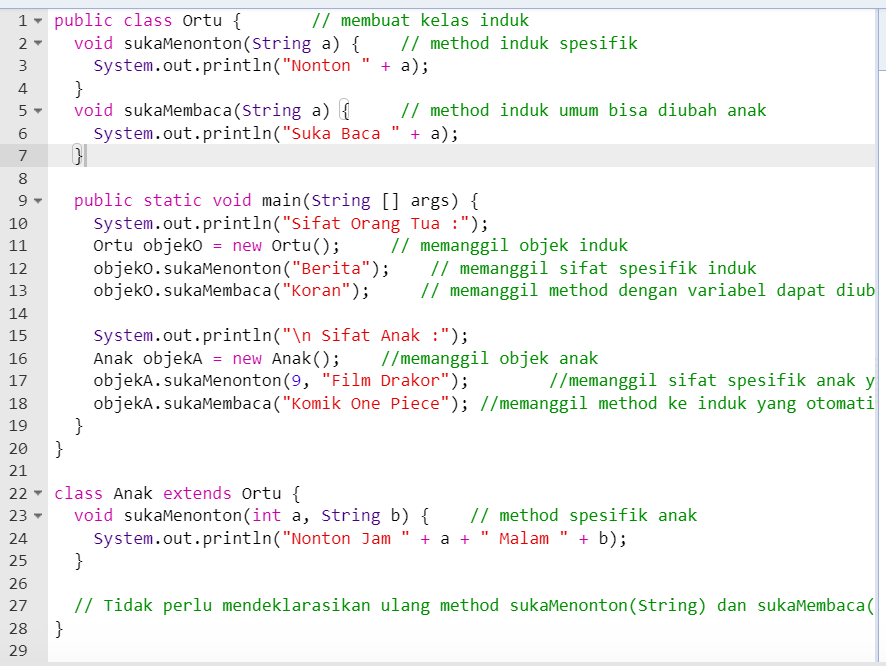


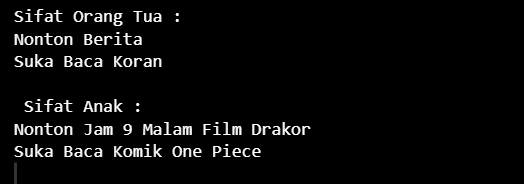
**[No.1 ] Penyusunan Algoritma dan Kode Program**

1. Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.  
Misalkan algoritma memasak mi instan:

1. Masak air
2. Buka bungkus
3. Masukkan mie
4. Masukkan bumbu
5. Hasilnya mie matang, taruh di piring
6. Mie siap disantap.
7. Kode program dan luaran





1. Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot

1. Analisa luaran yang dihasilkan   
   Luaran sudah sesuai dengan program yang disusun.   
   operator java yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data

**[No.1] Kesimpulan Analisa**

1. Constructor harus memiliki nama yang sama dengan kelas.

2. Jumlah parameter pada constructor dan argumen yang diberikan saat membuat objek harus sesuai.

3. Untuk mencetak atribut rambut dengan benar, atribut tersebut harus diinisialisasi di dalam

constructor.

**LATIHAN 2**

**[No. 2] Identifikasi Masalah:**

1.)Nama constructor harus sama dengan nama kelas (*Ortu, bukan ortu*).

2.)Constructor membutuhkan parameter yang diterima dari objek yang dibuat dalam method main.

3.)Constructor harus memiliki tanda kurung setelah namanya, meskipun tidak menerima parameter.

4.)Atribut nama dan rambut harus dideklarasikan di dalam kelas, dan diinisialisasi menggunakan *this.nama* dan *this.rambut* dalam constructor

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

1.)Nama constructor harus diubah agar sama dengan nama kelas (Ortu).

2.) Constructor harus menerima dua parameter (nama dan rambut) untuk mencetak informasi yang diinginkan.

3.) Atribut nama dan rambut harus dideklarasikan di dalam kelas

4.)Tanda kurung () harus ditambahkan setelah nama constructor

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

1.Algoritma

a.) Mulai (Start)

b.)Deklarasikan kelas Ortu.

c.) Deklarasikan atribut nama dan rambut bertipe String.

d.) Buat constructor Ortu yang menerima dua parameter:

* nama (String)
* rambut (String)

e.) Inisialisasi atribut nama dan rambut dengan nilai dari parameter constructor.

f.)Di dalam constructor, cetak teks yang menyatakan:

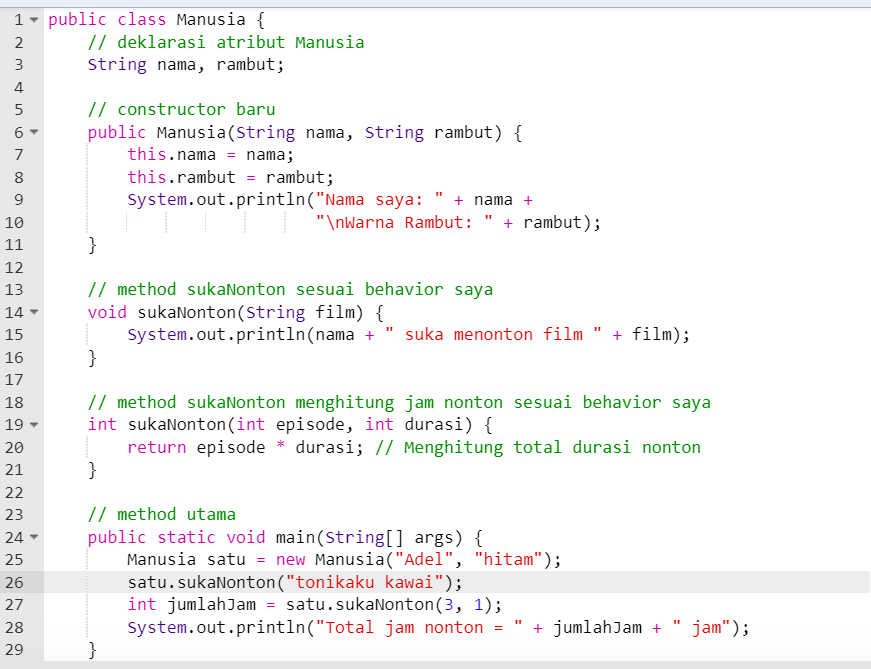
* "Nama saya : " + nama
* "Warna Rambut : " + rambut

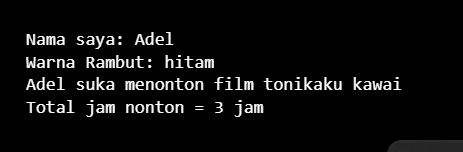
g.)Buat method main:

* Di dalam main, buat objek baru dari kelas Ortu dengan argumen "Putri" dan "hitam".

h.)Program selesai.

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.   
operator java yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data

**[No.2]Kesimpulan Analisa**

A.)Nama constructor harus diubah menjadi Ortu agar sama dengan nama kelas

B.)Atribut nama dan rambut harus dideklarasikan di dalam kelas, dan constructor harus menerima dua parameter (nama dan rambut) agar sesuai dengan argumen yang dikirim saat pembuatan objek

C.)Tanda kurung pada constructor harus ditambahkan, meskipun constructor tidak menerima parameter

**LATIHAN 3**

**[No. 3] Identifikasi Masalah:**

Pada soal masih ada pesan kesalahan yaitu kode awal memiliki beberapa kesalahan, seperti nama constructor yang tidak sesuai, method yang tidak dideklarasikan dengan benar, serta penggunaan variabel yang belum dideklarasikan

**[No.3] Analisis dan Argumentasi**

1.)Nama Constructor Konsisten dengan Nama Kelas:

* Constructor diubah dari Manusia1 menjadi Manusia agar sesuai dengan nama kelas. Ini memastikan bahwa objek dapat dibuat dengan benar dari kelas Manusia.

2.)Deklarasi Method yang Benar:

* Method sukaNonton(String film) sekarang menerima parameter film untuk menampilkan hobi menonton secara dinamis.
* Method kedua sukaNonton(int episode, int durasi) dihitung dan mengembalikan hasil perkalian antara jumlah episode dan durasi.

3.)Penggunaan Objek dan Pemanggilan Method yang Tepat:

* Pembuatan objek Manusia satu = new Manusia("Putri", "hitam"); sekarang berjalan dengan benar karena constructor telah diperbaiki.
* Pemanggilan method sukaNonton("Drakor") dan sukaNonton(2, 2) bekerja karena parameter dan method sudah dideklarasikan dengan benar

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

1.Algoritma

a.) Mulai (Start)

b.)Deklarasikan kelas Manusia.

c.)Di dalam kelas Manusia, deklarasikan dua atribut:

* nama bertipe String
* rambut bertipe String

d.)Buat constructor Manusia(String nama, String rambut):

* Inisialisasi atribut nama dan rambut dengan nilai yang diberikan melalui parameter.
* Cetak nama dan warna rambut menggunakan System.out.println.

e.)Deklarasikan method sukaNonton(String film):

* Parameter: film bertipe String.
* Cetak hobi menonton berdasarkan nilai film yang diberikan.

f.)Deklarasikan method sukaNonton(int episode, int durasi):

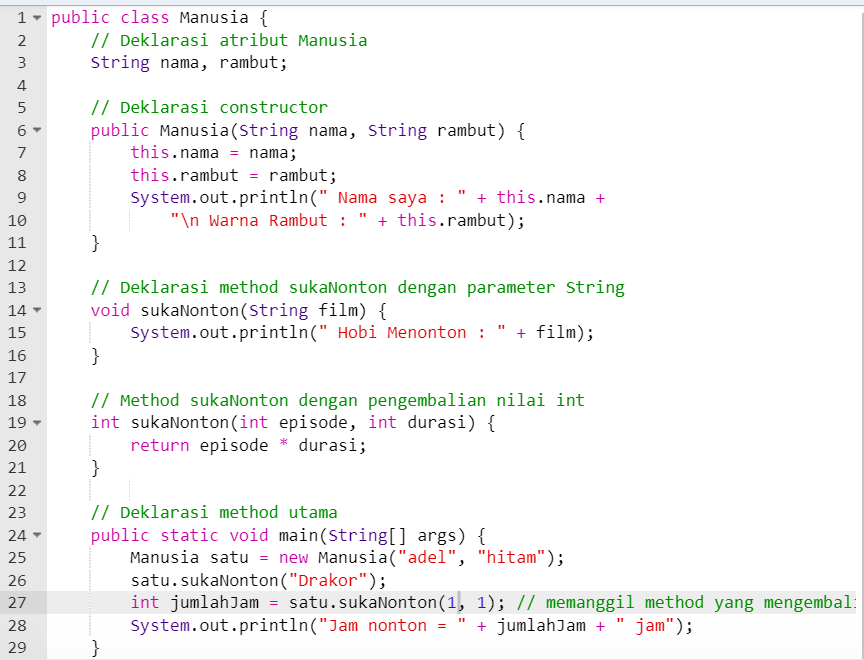
* Parameter: episode bertipe int, durasi bertipe int.
* Hitung total jam menonton: episode \* durasi.
* Kembalikan nilai hasil perhitungan.

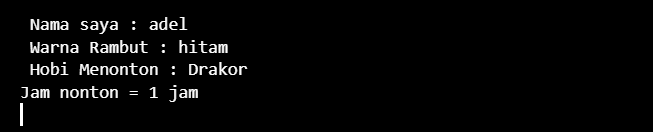
g.)Deklarasikan method utama main(String[] args):

* Buat objek Manusia satu = new Manusia("Putri", "hitam").
* Panggil method sukaNonton("Drakor") untuk mencetak hobi menonton.
* Panggil method sukaNonton(2, 2) untuk menghitung total jam menonton.
* Simpan hasil perhitungan ke dalam variabel jumlahJam.
* Cetak jumlah jam menonton ke layar.

h.)Selesai (End).

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.   
operator java yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data

**[No.3]Kesimpulan Analisa**

1.) Nama constructor harus disesuaikan dengan nama kelas.

2.) Method harus didefinisikan dengan tanda kurung (), dan variabel yang digunakan di dalam method harus dideklarasikan dengan benar.

3.)Perlu adanya konsistensi dalam penggunaan parameter pada method, serta deklarasi variabel yang sesuai di seluruh kelas

**LATIHAN 4**

**[No. 4] Identifikasi Masalah**

1. **Duplikasi Method main pada Kelas Ortu dan Anak**:
   * Pada kode ini, terdapat dua main method, satu di kelas Ortu dan satu di kelas Anak. Ini tidak menyebabkan kesalahan teknis, tetapi hanya satu main method yang akan dieksekusi saat program dijalankan. Memiliki main method di dua tempat bisa membingungkan dan tidak diperlukan jika hanya ingin menjalankan satu titik masuk.
2. **Overriding Method sukaMenonton(String a) di Kelas Anak**:
   * Method sukaMenonton(String a) dideklarasikan dua kali dalam kelas Anak. Kedua method ini memiliki tanda tangan yang sama (menerima parameter String), yang menyebabkan salah satu method menimpa yang lain (overriding). Dalam hal ini, salah satu dari method tersebut akan diabaikan karena mereka memiliki parameter yang sama, dan ini bisa membingungkan.
3. **Pemanggilan Method sukaMenonton dengan Parameter Berbeda**:
   * Pada kelas Anak, method sukaMenonton(int a, String b) memiliki parameter tambahan int yang tidak ada di kelas Ortu. Ini merupakan bentuk **method overloading**, yang merupakan fitur yang valid, tetapi cara pemanggilan dan deklarasinya harus konsisten.
4. **Tidak Konsisten dalam Menangani Polimorfisme**:
   * Di dalam kelas Anak, meskipun method sukaMenonton(int a, String b) ditambahkan, method sukaMenonton(String a) yang ada di Ortu juga di-declare ulang dalam Anak. Ini sebenarnya tidak perlu karena method dari kelas induk akan diwariskan secara otomatis jika tidak di-overriding. Jika tidak ada perubahan dalam method, lebih baik tidak mendeklarasikan ulang method tersebut di kelas Anak.

**[No.4] Analisis dan Argumentasi**

1.)Penggunaan Satu main Method:

* Menjaga hanya satu main method di kelas Ortu menyederhanakan eksekusi program.

2.) Penghapusan Method yang Tidak Perlu:

* Menghapus method yang tidak diperlukan mengurangi kebingungan dan potensi error.

3.)Pemanfaatan Overloading dan Overriding yang Benar:

* Memastikan bahwa method yang di-overload atau di-override benar-benar memerlukan perubahan atau penambahan fungsionalitas

**[No.4 ] Penyusunan Algoritma dan Kode Program**

1.)algoritma

**a.) inisialisasi Program**

* Buat *objek Ortu dan objek Anak*.

**b.)** **Eksekusi Kelas Ortu**

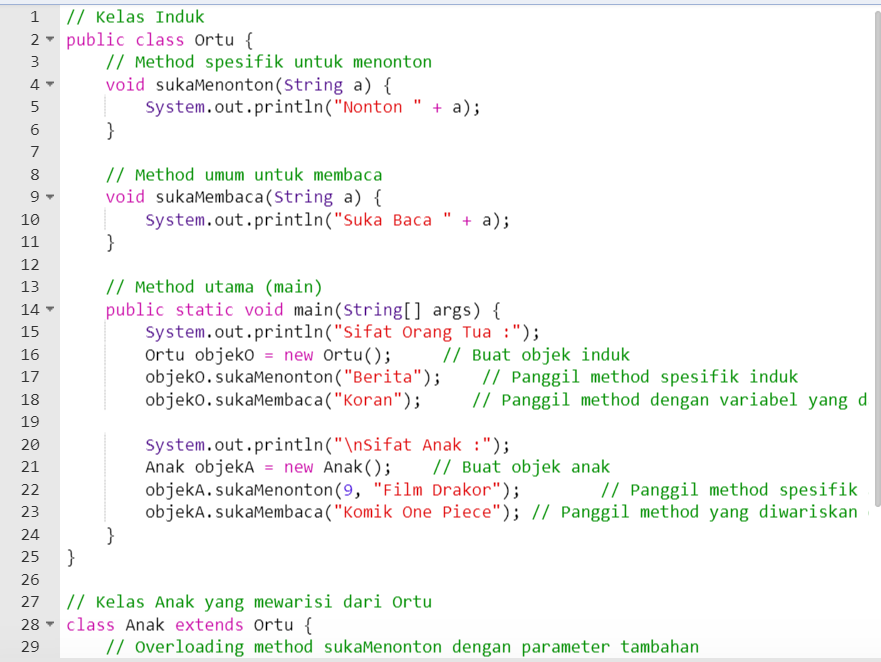
* Tampilkan teks "*Sifat Orang Tua* :"
* Buat objek Ortu dengan nama *objekO.*
* Panggil method sukaMenonton pada objekO dengan parameter "*Berita*".
* Panggil method sukaMembaca pada objekO dengan parameter "*Koran*".

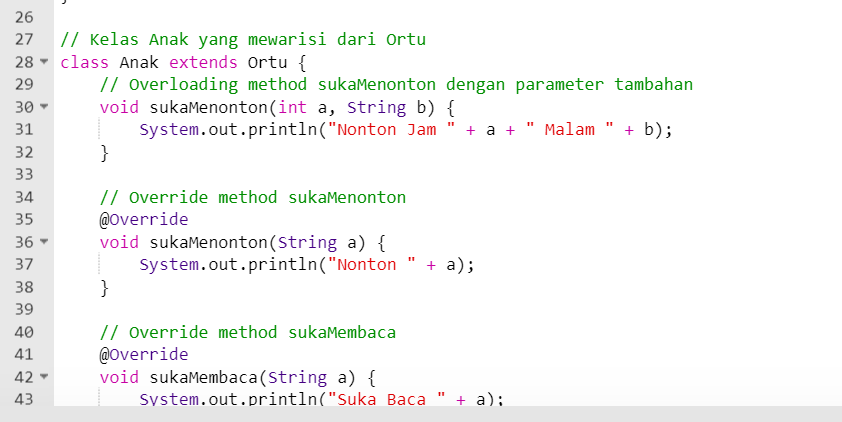
**C.)** **Eksekusi Kelas Anak**

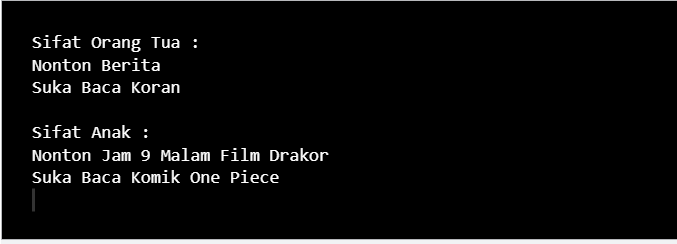
* Tampilkan teks "*Sifat Anak* :"
* Buat objek Anak dengan nama objekA.
* Panggil method sukaMenonton pada objekA dengan parameter 9 dan "*Film Drakor*".
* Panggil method sukaMembaca pada objekA dengan parameter "*Komik One Piece*".

**d.)** **Program Selesai**

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.   
operator java yang ditampilkan telah sesuai dengan kebutuhan dan permintaan data

**[No.4]Kesimpulan Analisa**

1.)Kelas Ortu dan Anak menggunakan konsep pewarisan OOP dengan baik, namun ada beberapa redundansi dalam implementasi method.

2.)Penggunaan method overload dan override harus tepat dan efektif sesuai dengan kebutuhan fungsionalitas tambahan atau perubahan.

3.)Perbaikan kode dapat fokus pada mengurangi pengulangan dan meningkatkan keterbacaan serta pemeliharaan

**REFLEKSI**

pengalaman belajar kami dari matkul komputer dan pemograman ini adalah dapat belajar bahasa pemograman dasar pada bahasa pemograman pada java yang dimana kami di arahkan untuk belajar latihan pre test pada web e-learning untuk melatih kemampuan kami dalam menganalisis **Kelas, Objek, Method**