**Template Lembar Kerja Individu dan Kelompok**

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **NAMA: MUHAMMAD NABIL**  **NPM: G1F024068** | **Kelas, Objek, Method** | **18 September 2024** |
| **Unit 1 Kelas (Class)** | | |
| **Contoh 1:** public class Manusia {  // deklarasi kelas     //deklarasi atribut Manusia dalam variabel     String nama, rambut;          //deklarasi constructor     public Manusia1 (String nama) {             System.out.println(" Nama saya : "+ nama +              "\n Warna Rambut : " + rambut);         }          //deklarasi method utama     public static void main( String[] args) {             Manusia1 satu = new Manusia1("Putri", "hitam"); }  }  Luaran 1: Exception in thread "main" java.lang.Error: Unresolved compilation problem:      The constructor Manusia1(String, String) is undefined     at Manusia1.main(Manusia1.java:13)  **Latihan 1:** 1.1. Perbaiki pesan kesalahan Contoh 1! 1.2. Cermati contoh 1. susun kode menggunakan constructor dengan parameter data pribadi anda!  **Jawaban:**  1.1    public class classmethod {    String nama, rambut;    public classmethod(string nama, string rambut) {  this.nama = nama;  this.rambut = rambut;  System.***out***.println(" Nama saya : " + this.nama +  "\n Warna Rambut : " + this.rambut);  }    public static void main(string[] args) {  classmethod satu = new classmethod("Putri", "hitam");    }  }  Output:    1.2    public class classmethod {    private String nama;  private String rambut;  public classmethod(String nama, String rambut) {  this.nama = nama;  this.rambut = rambut;  System.*out*.println("Nama saya : " + this.nama +  "\n Warna Rambut : " + this.rambut);  }  public static void main(String[] args) {  classmethod satu = new classmethod("Nabil", "hitam");  }  }  Output:  **Contoh 2:**Salin dan tempel kode program berikut ke Eclipse atau JDoodle.  public class Ortu {     //deklarasi constructor  (variabel constructor)     public ortu   {           //nama dan rambut adalah variabel constructor         System.out.println(" Nama saya : "+ nama +          "\n Warna Rambut : " + rambut);     }     public static void main (String[] args) {         Ortu satu = new Ortu("Putri", "hitam"); }   }  **Luaran 2:** Exception in thread "main" java.lang.Error: Unresolved compilation problem:      The constructor Ortu(String, String) is undefined      at Ortu.main(Ortu.java:9)  **Latihan 2:** 2.1.  Evaluasi penyebab kesalahan dan perbaiki kode tersebut! 2.2.  Apabila nanti Anda akan memiliki keturunan, analisa sifat (atribut) dan constructor sebagai Ortu apa yang akan diturunkan (gunakan data karakter pribadi anda) ? 2.3.  Rancanglah kode program untuk sifat (atribut) dan constructor overloaded dari Latihan 2.2!  **Jawaban:**  2.1  Evaluasi:  1.Nama kelas ortu sebaiknya menggunakan huruf kapital di awal sesuai dengan konvensi penamaan kelas di Java. Jadi, seharusnya Ortu.  2.Nama constructor harus sama dengan nama kelas, yaitu Ortu, tetapi di kode Anda tertulis ortu. Ini menyebabkan kesalahan karena Java membedakan huruf besar dan kecil.  3. Tipe data untuk parameter di constructor sudah benar, yaitu String dan int, tetapi harus dipastikan bahwa penulisan konsisten.  2.2  Nama : Putri  Warna rambut : Hitam  Umur: 25  Hobi : Menonton  2.3      public class ortu {  public ortu(String nama, String rambut, int umur, String hobi ) {  //nama dan rambut adalah variabel constructor  System.*out*.println(" Nama saya : "+ nama +  "\n Warna Rambut : " + rambut+  "\n Umur: " + umur +  "\n Hobi: " + hobi);  }  public static void main (String[] args) {  ortu satu = new ortu("Putri", "hitam", 25, "menonton");  }  }  Output: | | |
| **Unit 3 Method** | | |
| **Contoh 3:**Salin dan tempel kode program berikut ke Eclipse atau JDoodle.  public class Manusia {     //deklarasi atribut Manusia dalam variabel     String nama, rambut;          //deklarasi constructor     public Manusia1(String nama, String rambut) {             System.out.println(" Nama saya : "+ nama +              "\n Warna Rambut : " + rambut);         }          //deklarasi method     void sukaNonton  {         System.out.println(" Hobi Menonton : " + film);     }          int sukaNonton {         episode\*durasi;     }                  //deklarasi method utama     public static void main( String[] args) {             Manusia satu = new Manusia("Putri", "hitam");             satu.sukaNonton("Drakor");             int jumlahJam = satu.sukaNonton(2, 2);             System.out.println("Jam nonton = " +jumlahJam + " jam"); }  }  **Luaran 3:** Exception in thread "main" java.lang.Error: Unresolved compilation problems:      The method sukaNonton(String) is undefined for the type Manusia1     The method sukaNonton(int, int) is undefined for the type Manusia1     at Manusia1.main(Manusia1.java:23)  **Latihan 3:** 3.1.  Evaluasi penyebab kesalahan dan perbaiki kode tersebut! 3.2.  Berdasarkan Latihan 2.2. Anda sudah punya kode program untuk atribut dan constructor sebagai Ortu.         Kembangkanlah kode program untuk method dari Ortu dengan data perilaku pribadi Anda yang menggunakan:         a) method overloading,         b) method dengan return value         c) method tanpa return value  **Jawaban:**  3.1    3.2    a)Method Overloading adalah konsep dalam pemrograman di mana dua atau lebih metode dalam satu class memiliki nama yang sama tetapi berbeda dalam parameter (jumlah, tipe, atau urutan parameter). Ini memungkinkan programmer untuk menggunakan nama metode yang sama untuk melakukan tugas yang mirip, tetapi dengan cara yang sedikit berbeda berdasarkan input yang diberikan.  b) Method dengan Return Value adalah metode yang mengembalikan nilai setelah melakukan operasi tertentu. Tipe data dari nilai yang dikembalikan harus dinyatakan dalam deklarasi metode. Jika metode tidak mengembalikan nilai, maka tipe datanya harus void.  c)Method tanpa Return Value adalah metode yang tidak mengembalikan nilai apa pun setelah dieksekusi. Tipe data dari metode ini harus dideklarasikan sebagai void. Metode ini biasanya digunakan untuk melakukan aksi tertentu, seperti mencetak output ke layar atau mengubah status objek. | | |
| **Unit 4 Extends** | | |
| **Contoh 4:**Salin dan tempel kode program berikut ke JDoodle. Kemudian catat waktu eksekusinya.  public class Ortu {       // membuat kelas induk     void sukaMenonton(String a) {    // method induk spesifik       System.out.println("Nonton " + a);     }     void sukaMembaca(String a) {     // method induk umum bisa diubah anak       System.out.println("Suka Baca " + a);     }  public static void main(String [] args) {     System.out.println("Sifat Orang Tua :");      Ortu objekO = new Ortu();     // memanggil objek induk     objekO.sukaMenonton("Berita");    // memanggil sifat spesifik induk     objekO.sukaMembaca("Koran");     // memanggil method dengan variabel dapat diubah          System.out.println("\n Sifat Anak :");         Anak objekA = new Anak();    //memanggil objek anak     objekA.sukaMenonton(9, "Film Drakor");        //memanggil sifat spesifik anak yang diturunkan induk     objekA.sukaMembaca("Komik One Piece"); //memanggil method ke induk yang otomatis diturunkan tanpa deklarasi ulang di anak }    }  class Anak extends Ortu {   void sukaMenonton(int a, String b) {         System.out.println("Nonton Jam " + a + " Malam " + b);   }       void sukaMenonton(String a) {            // method induk spesifik         System.out.println("Nonton " + a);   }   void sukaMembaca(String a) {     // method induk umum bisa diubah anak         System.out.println("Suka Baca " + a);   } public static void main(String [] args) {     System.out.println("Sifat Orang Tua :");      Ortu objekO = new Ortu();     // memanggil objek induk     objekO.sukaMenonton("Berita");        // memanggil sifat spesifik induk     objekO.sukaMembaca("Koran");     // memanggil method dengan variabel dapat diubah                  System.out.println("\n Sifat Anak :");         Anak objekA = new Anak();    //memanggil objek anak     objekA.sukaMenonton(9, "Film Drakor");        //memanggil sifat spesifik anak yang diturunkan induk     objekA.sukaMembaca("Komik One Piece"); //memanggil method ke induk yang otomatis diturunkan tanpa deklarasi ulang di anak **}    }**  **Luaran 4:** Sifat Orang Tua : Nonton Berita Suka Baca Koran  Sifat Anak : Nonton Jam 9 Malam Film Drakor Suka Baca Komik One Piece  **Latihan 4:** 4.1. Evaluasi method yang dimiliki Contoh 4  pada class Anak extends Ortu dengan method di class Ortu.        Simpulkan hasil evaluasi Anda agar method ini menjadi efisien! 4.2. Setelah dirunning di JDoodle, catat waktu eksekusinya.         Susun kembali kode program yang dapat mengefisienkan waktu eksekusi!  **Jawaban:**  4.1  1.Method Overloading: Class Ortu memiliki dua metode yang sama, sukaMenonton, tetapi dengan parameter yang berbeda. Ini adalah contoh method overloading yang efektif karena memungkinkan penggunaan metode yang sama untuk melakukan tugas yang berbeda berdasarkan input yang diberikan.  2.Inheritance: Class Anak mengextends class Ortu, yang berarti semua metode yang ada di class Ortu juga ada di class Anak. Ini membuat kode lebih terstruktur dan mudah dipahami.  3.Effisiensi:  Method Overloading: Memungkinkan penggunaan metode yang sama dengan parameter yang berbeda, sehingga kode menjadi lebih singkat dan mudah dibaca.  Inheritance: Memungkinkan penggunaan metode yang sudah ada di class induk tanpa perlu deklarasi ulang, sehingga mengurangi duplikasi kode.  4.2 | | |
| **Kesimpulan** | | |
| Class Method dalam bahasa pemrograman Java adalah metode yang didefinisikan di dalam class dan dapat dipanggil melalui objek dari class tersebut. | | |
|  | | |