1. Diberikan kode java sebagai berikut **[15]** :

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | public class Student {  // variables  private int npm;  private string name;  private string address;  private double ipk;  private Faculty faculty;  }  public class Faculty{  // variables  private int id;  private string name;  private description;  } |

Jawablah pertanyaan berikut :

* 1. Tuliskan contoh method *constructor* untuk kelas **Student** dan kelas **Faculty** ! [5]

|  |
| --- |
| *Jawaban* :  public class Student {  // variables  private int npm;  private String name;  private String address;  private double ipk;  private Faculty faculty;  public Student(int npm, String name, String address, double ipk, Faculty faculty)  {  this.npm = npm;  this.name = name;  this.address = address;  this.ipk = ipk;  this.faculty = faculty;  }  }  public class Faculty{  // variables  private int id;  private String name;  private String description;  public Faculty(int id, String name, String description)  {  this.id = id;  this.name = name;  this.description = description;    }  } |

* 1. Tambahkan *method* yang diperlukan agar masing-masing variabel pada kelas **Student** dan kelas **Faculty** dapat diakses (diambil dan diset nilainya) oleh kelas lain. [5]

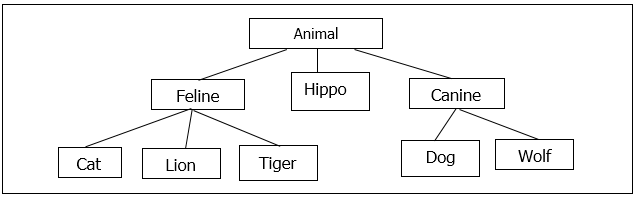
|  |
| --- |
| *Jawaban* :  public class Student {  // variables  private int npm;  private String name;  private String address;  private double ipk;  private Faculty faculty;  ....  public void setNpm(int npm)  {  this.npm = npm;  }  public int getNpm()  {  return npm;  }  public void setName(String name)  {  this.name = name;  }  public String getName()  {  return name;  }  public void setAddress(String address)  {  this.address = address;  }  public String getAddress()  {  return address;  }  public void setIpk(double ipk)  {  this.ipk = ipk;  }  public double getIpk()  {  return ipk;  }  public void setFaculty(Faculty faculty)  {  this.faculty = faculty;  }  public String getFaculty()  {  return faculty;  }  }  public class Faculty{  // variables  private int id;  private String name;  private String description;  ....  public void setId(int id)  {  this.id = id;  }  public double getId()  {  return id;  }  public void setName(String name)  {  this.name = name;  }  public String getName()  {  return name;  }  public void setDescription(String description)  {  this.description = description;  }  public String getDescription()  {  return description;  }  } |

* 1. Tuliskan contoh kode java untuk menginstansiasi objek berikut : [5]

|  |  |
| --- | --- |
| **Faculty**  “1, Akuntansi, Jurusan Akuntansi ”  “2, JTK, Jurusan Teknik Komputer” | **Student**  “3490136, Wahyu Hidayat, Cimahi, 3.5, JTK”  “3490212, Susi Susilowati, Bandung, 3.2, Akuntansi” |

|  |
| --- |
| *Jawaban :*  Faculty fakultas1 = new Faculty(1, “Akuntansi”, “Jurusan Akuntansi ”);  Faculty fakultas2 = new Faculty(2, “JTK”, “Jurusan Teknik Komputer”);  Student student1 = new Student(3490212, “Susi Susilowati”, “Bandung”, 3.2, “Akuntansi”);  Student student2 = new Student(3490136, “Wahyu Hidayat”, “Cimahi”, 3.5, “JTK ”); |

1. Perhatikan hirarki kelas berikut ini **[25]**:



* 1. **Tuliskan** kerangka kode java untuk mengimplementasikan kelas pada **poin a** ! [7]

|  |
| --- |
| *Jawaban :*  public abstract class Animal {  …  }  public class Feline extends Animal {  …  }  public class Hippo extends Animal {  …  }  public class Canine extends Animal {  …  }  public class Cat extends Feline {  …  }  public class Lion extends Feline {  …  }  public class Tiger extends Feline {  …  }  public class Dog extends Canine {  …  }  public class Wolf extends Canine {  …  } |

* 1. **Tuliskan** contoh penerapan *method overloading* dan *method overriding* **–** nya ! [8]

|  |
| --- |
| *Jawaban :*  public abstract class Animal {  public void eat() {  System.out.println(“I’m eat everything”);  }  }  public class Feline extends Animal {  **//overriding**  public void eat() {  System.out.println(“I’m eat meats”);  }  **// overloading**  public int tailLength(int a, int b)  {  return a + b;  }  public int tailLength(int a, int b, int c)  {  return a + b + c;  }  }  public class Hippo extends Animal {  //overriding  public void eat() {  System.out.println(“I’m eat grass”);  }  }  public class Canine extends Animal {  //overriding  public void eat() {  System.out.println(“I’m eat meats”);  }  }  public class Cat extends Feline {  //overriding  public void eat() {  System.out.println(“I’m eat fish”);  }  }  public class Lion extends Feline {  //overriding  public void eat() {  System.out.println(“I’m eat meats”);  }  }  public class Tiger extends Feline {  //overriding  public void eat() {  System.out.println(“I’m eat meats”);  }  }  public class Dog extends Canine {  //overriding  public void eat() {  System.out.println(“I’m eat bone”);  }  }  public class Wolf extends Canine {  //overriding  public void eat() {  System.out.println(“I’m eat meats”);  }  } |

* 1. **Jelaskan** penerapan *multiple inheritance* pada kasus tersebut dengan ditambahkannya suatu interface ! [10]

|  |
| --- |
| *Jawaban :*  Misalkan Class Cat extends Feline di implements kan dengan Interface Pet. Sehingga bentuk implementasinya sebagai berikut :  public Interface Pet {  void Health();  }  public class Cat extends Feline implements Pet {  public void Health(){  system.out.println(“I feel healthy”);  }  } |