1. The following Java applications contain errors. Point out the statement(s) that contain errors. Explain what each of the errors is, and how it can be fixed.

|  |  |
| --- | --- |
| public class **OOPExercises** {  public static void main(String[] args) {  A objA = new A();  System.out.println("in main(): ");  System.out.println("objA.a = "+objA.a);  objA.a = 222;  }  } | **Point out the error(s) and how they can be fixed.** |
| public class **A** {  private int a = 100;  public void setA( int value) {  a = value;  }  public int getA() {  return a;  }  } //class A |

|  |  |
| --- | --- |
| public class **OOPExercises** {  public static void main(String[] args) {  System.out.println("in main(): ");  System.out.println("objA.a = "+getA() );  setA(123);  }  } | **Point out the error(s) and how they can be fixed.** |
| public class **A** {  private int a = 100;  public void setA( int value) {  a = value;  }  public int getA() {  return a;  }  } //class A |

|  |  |
| --- | --- |
| public class **OOPExercises** {  public static void main(String[] args) {  A objA = new A( );  double result;  result = objA.getA( );  System.out.println("objA.a = "+ result);  }  } | **Point out the error(s) and how they can be fixed.** |
| public class **A** {  private int a = 100;  public void setA( int value) {  a = value;  }  public int getA() {  return a;  }  } //class A |

|  |  |
| --- | --- |
| public class **B extends A** {  private int a = 222;  public static void main(String[] args) {  System.out.println("in main(): ");  System.out.println("a = "+a );  a = 123;  }  } | **Point out the error(s) and how they can be fixed.** |
| public class **A** {  private int a = 100;  public void setA( int value) {  a = value;  }  public int getA() {  return a;  }  } //class A |

1. Show the output of the following applications.

|  |  |
| --- | --- |
| public class **OOPExercises** {  public static void main(String[] args) {  A objA = new A();  B objB = new B();  System.out.println("in main(): ");  System.out.println("objA.a = "+objA.getA());  System.out.println("objB.b = "+objB.getB());  objA.setA (222);  objB.setB (333.33);  System.out.println("objA.a = "+objA.getA());  System.out.println("objB.b = "+objB.getB());  }  } | **Output:** |
| public class **A** {  int a = 100;  public A() {  System.out.println("in the constructor of class A: ");  System.out.println("a = "+a);  a = 333;  System.out.println("a = "+a);  }  public void setA( int value) {  a = value;  }  public int getA() {  return a;  }  } //class A | |
| public class **B** {  double b = 123.45;  public B() {  System.out.println("-----in the constructor of class B: ");  System.out.println("b = "+b);  b = 3.14159;  System.out.println("b = "+b);  }  public void setB( double value) {  b = value;  }  public double getB() {  return b;  }  } //class B | |



|  |  |
| --- | --- |
| public class **OOPExercises** {  public static void main(String[] args) {  //A objA = new A();  B objB = new B();  System.out.println("in main(): ");  //System.out.println("objA.a = "+objA.getA());  System.out.println("objB.b = "+objB.getB());  //objA.setA (222);  objB.setB (333.33);  //System.out.println("objA.a = "+objA.getA());  System.out.println("objB.b = "+objB.getB());  }  } | **Output:** |
| public class **A** {  int a = 100;  public A() {  System.out.println("in the constructor of class A: ");  System.out.println("a = "+a);  a = 333;  System.out.println("a = "+a);  }  public void setA( int value) {  a = value;  }  public int getA() {  return a;  }  } //class A | |
| public class **B extends A** {  double b = 123.45;  public B() {  System.out.println("-----in the constructor of class B: ");  System.out.println("b = "+b);  b = 3.14159;  System.out.println("b = "+b);  }  public void setB( double value) {  b = value;  }  public double getB() {  return b;  }  } //class B | |



|  |  |
| --- | --- |
| public class **OOPExercises** {  static int a = 555;    public static void main(String[] args) {  A objA = new A();  B objB = new B();  System.out.println("in main(): ");  System.out.println("a = "+a);  a = 444;  System.out.println("objB.a = "+objB.getA());  objA.setA (77777);  objB.rollBackA();  System.out.println("After roll back -----");  System.out.println("a = "+a);  System.out.println("objA.a = "+objA.getA());  System.out.println("objB.a = "+objB.getA());  }  } | **Output:** |
| public class **A** {  int a = 100;  public A() {  //System.out.println("in the constructor of class A: ");  //System.out.println("a = "+a);  a = 333;  //System.out.println("a = "+a);  }  public void setA( int value) {  a = value;  }  public int getA() {  return a;  }  } //class A | |
| public class **B** extends A {  private int a = 123;  public B() {  a = 2222;  }  public void rollBackA () {  a = super.getA();  }  public void setA( int value) {  a = value;  }  public int getA() {  return a;  }  } //class B | |



|  |  |
| --- | --- |
| public class **OOPExercises** {  static int a = 555;    public static void main(String[] args) {  A objA = new A();  B objB1 = new B();  A objB2 = new B();  C objC1 = new C();  B objC2 = new C();  A objC3 = new C();  objA.display();  objB1.display();  objB2.display();  objC1.display();  objC2.display();  objC3.display(); }  } | **Output:** |
| public class **A** {  int a = 100;  public void display() {  System.out.printf("a in A = %d\n", a);  }  } //class A | |
| public class **B** extends A {  private int a = 123;  public void display() {  System.out.printf("a in B = %d\n", a);  }  } //class B | |
| public class **C** extends B {  private int a = 543;  public void display() {  System.out.printf("a in C = %d\n", a);  }  } //class C | |

1. UML Diagrams
   1. Draw a UML class diagram (with associations) to show the design of the Java application in EX 2.2.
   2. The partial design of a Java application for a child care center is given in the following UML diagram. Note that the diagram is not complete. How do you represent the following relationships in the design: *father*, *mother*, and *guardian*? Revise the diagram to include those relationships in the design.

**Person**

- lastName: String

- firstName: String

- father: Person

- mother: Person

+ setLastName(String)

+ getLastName( ): String

.

.

.

**Child**

- guardian: Person

- age: int

- height: int

- weight: double

+ setGuardian(Person)

+ getGuardian( ): Person

.

.

.



* 1. Implement the design in EX 3.2 as a Java application. Add the *set* and *get* methods for each of the attributes. Note that Child is a subclass of Person.