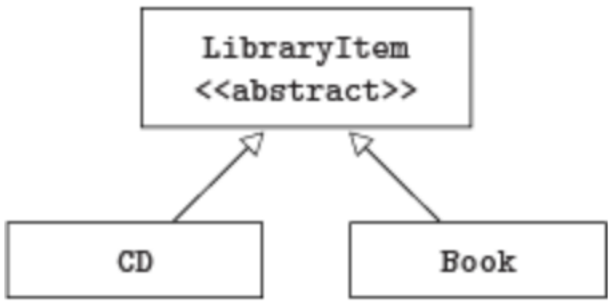
Inheritance
Multiple Choice Questions

Questions 1 and 2 refer to this hierarchy.



(1)Which of the following declarations will cause an error? You may assume that each of the classes above has a default constructor.

A) LibraryItem item = new LibraryItem();
B) Book b = new LibraryItem();
C) LibraryItem cd = new CD();

(2) Suppose that, of the three classes shown in the diagram, only the LibraryItem class has a method called getPrice defined in it, whose specification is as follows:

```
/** @return the price of this LibraryItem */
public double getPrice()
```

Now consider the following declaration that appears in a client program.

```
List<LibraryItem> libraryList = new ArrayList<LibraryItem>();
```

Assume that libraryList is initialized with LibraryItem objects. Consider the following code segment that computes the total cost of all items stored in libraryList.

```
double total = 0.0;
for (LibraryItem item : libraryList)
        total += item.getPrice();
```

The use of getPrice in the above code segment is an example of

A) An inherited method
B) An overridden method
C) Polymorphism
D) An overloaded method
E) An abstract method

(3) Consider the following class definitions.

```
        public class Dancer
        {
                //...data fields and several methods not shown

                public void leap()
                { /* implementation not shown */ }
        }

        public class BalletDancer extends Dancer
        {
                public void glide()
                { /* implementation not shown */ }
                //...other methods not shown
        }
```

A client program has the following declarations

```
        Dancer d = new BalletDancer();
        BalletDancer b = new BalletDancer();
```

Which of the following statements is(are) valid?
I.   b.leap();
II.   d.glide();
III.   d.leap();

A) I only
B) II only
C) III only
D) II and III only
[✓ E) I and III only]

Questions 4 and 5 refer to the following classes
(4)

```
public class PosInt
{
        private int value;
        /** Constructs a PosInt with the given value.
         *  Precondition: aValue > 0.
         */
        public PosInt(int aValue)
        { value = aValue; }

        public int getValue()
        { return value; }
}

public class EvenPosInt extends PosInt
{
        /** @return the next consecutive int value that is greater
         *  than the value of the current EvenPosInt
         */
        public int getNextEven()
        { /* implementation */ }
}
```

Which of the following declarations in a client class would cause an error?

I. PostInt p = new PosInt();
II. PosInt even = new EvenPosInt();
III. PosInt pe = new EvenPosInt(10);

A) I only
B) II only
C) I and II only
D) II and III only
[✓ E) I, II and III]

(5) Suppose that the following constructor is added to the EvenPosInt class:

```
public EvenPosInt(int aValue)
{ super(aValue); }
```

Which of the following is a correct /*implementation*/ for the getNextEven method?

I. return value + 2;
II. return getValue() + 2;
III. return this + 2;

A) I only
[✓ B) II only]
C) III only
D) II and III only
E) I, II and III

(6) Consider the following class declarations

```
public class ClassA
{       ...
        public void doSomething(ClassB b, ClassC c)
        {       ...
        }
}

public class ClassB extends ClassA
{
        ...
}

public class ClassC extends ClassB
{
        ...
}
```

A different class has these declarations: (You may assume that each class has a default constructor)

```
ClassA objA = new ClassA();
ClassB objB = new ClassB();
ClassC objC = new ClassC();
```

Which of the following is a correct call to doSomething?

[✓ A) objC.doSomething(objC, objC);]
B) objC.doSomething(objA, objA);
C) objB.doSomething(objC, objB);
D) objB.doSomething(objB, objB);
E) objA.doSomething(objA, objA);

Questions 7 and 8 refer to the following declarations

```
public abstract class Bird
{
        private String species;

        /** Constructor */
        public Bird(String aSpecies)
        { species = aSpecies; }

        public abstract void makeNoise();

        //Other methods not shown.
}

public class Owl extends Bird
{
        /** Constructor */
        public Owl(String aSpecies)

        { /* implementation */ }

        public void makeNoise()
        { System.out.println("hoot hoot"); }

        //Other methods not shown.
}

public class Hen extends Bird
{
        /** Constructor */
        public Hen(String aSpecies)

        { /* implementation */ }

        public void makeNoise()
        { System.out.println("cluck cluck"); }

        //Other methods not shown.
}
```

(7) Which is the correct /* implementation */ code for the constructor of the Hen class?
I. super();
II. super(aSpecies);
III. species = aSpecies;

A) I only
[✓ B) II only]
C) III only
D) II and III only
E) I, II and III

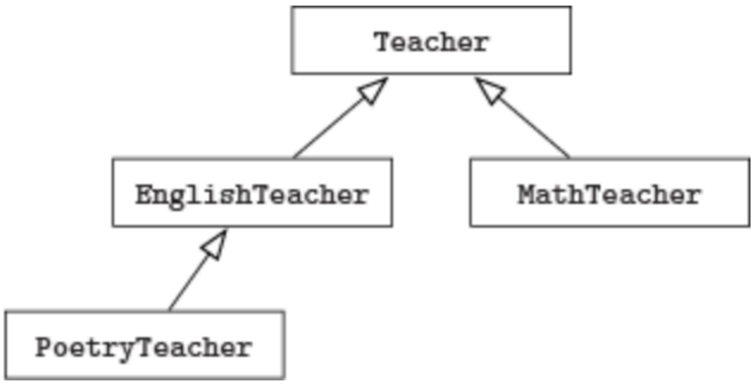
(8) A client class has an ArrayList<Bird> birds that is initialized with many different objects that are subclasses of Bord. Each subclass has an overridden makeNoise

method, similar to those shown in the Owl and Hen classes. What is the effect of executing the following code segment?

```
        for (Bird b : birds)
                b.makeNoise();
```

[✓ A) The unique noise for each particular Bird will be the output]
B) There will be a compile-time error with a message to the effect that makeNoise is abstract in class Bird.
C) There will be a compile-time error with a message to the effect that there cannot be different types in ArrayList<Bird>
D) There will be a compile-time error with a message to the effect that Bird b needs to be cast to its actual type.
E) An IndexOutOfBoundsException will be thrown.

(9) Consider the following hierarchy of classes for questions 9 and 10



Which of the following is(are) true?

I. PoetryTeacher inherits all public methods of Teacher and EnglishTeacher
II. MathTeacher inherits the constructors of Teacher
III. PoetryTeacher inherits the constructors of EnglishTeacher and Teacher.

A) None
[✓ B) I only]
C) II only
D) III only
E) I, II and III

(10) A program is written to print data about teachers in a school.

```
public class TeacherStuff
{
        //Private instance variables and constructors not shown.

        public void displayTitle(Teacher t)
        { /* implementation not shown */ }

        //Other methods not shown.
}
```

A client class has the following declarations:

```
Teacher t1 = new MathTeacher();
EnglishTeacher t2 = new EnglishTeacher();
EnglishTeacher t3 = new PoetryTeacher();
```

In this client class, which of the following methods calls will cause an error?

I. displayTitle(t1);
II. displayTitle(t2);
III. displayTitle(t3);

[✓ A) None]
B) I only
C) II only
D) III only
E) I, II and III

(11) Refer to the classes below for questions 11 and 12

```
public class ClassA
{
        //default constructor not shown ...

        public void method1()
        { /* implementation of method1 */ }
}

public class ClassB extends ClassA
{
        //default constructor not shown ...

        public void method1()
        { /* different implementation from method1 in ClassA*/ }

        public void method2()
        { /* implementation of method2 */ }
}
```

Which is a true statement about the methods in ClassA and ClassB?

A) ClassB inherits method2 from ClassA.
B) ClassA inherits method2 from ClassB.
C) method1 is an example of an overloaded method.
D) method2 is an example of an overridden method.
E) method1 is an example of an overridden method.

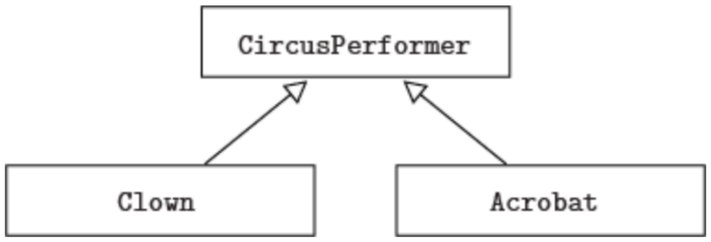
(12) Consider the following declarations in a client class.

```
        ClassA ob1 = new ClassA();
        ClassA ob2 = new ClassB();
```

Which of the following method calls will cause an error?
I. ob1.method();
II. ob1.method2();
III. ob2.method2();

A) I only
B) II only
C) III only
D) II and III only
E) I, II and III

(13) Consider the classes below, where Clown and Acrobat are subclasses of CircusPerformer.



Which of the following is a true statement about the classes shown?
A) Each of the classes- CircusPerformer, Clown, and Acrobat – can have a method doTrick that has different code.
B) Both Clown and Acrobat inherit the constructors in CircusPerformer.
C) If CircusPerformer has a private instance variable routine, neither Clown nor Acrobat can access it.
D) If CircusPerformer has a private method tumble, both Clown and Acrobat can access it.
E) If Clown has a private instance variable jokes, CircusPerformer can access it.

(14) Consider the following class declarations

```
public class Base
{
  private int myVal;

  public Base()
  {  myVal = 0;  }

  public Base(int x)
  {  myVal = x;  }
}


public class Sub extends Base
{
  public Sub()
  {  super(0);  }
}
```

Which of the following statements will NOT compile?
A) Base b1 = new Base();
B) Base b2 = new Base(5);
C) Base s1 = new Sub();
D) Sub s2 = new Sub();
[✓ E) Sub s3 = new Sub(5);]

(15)

```
public class Shape {
  public void what() { System.out.print("Shape ");}
}

public class Rectangle extends Shape {
  public void what() { System.out.print("Rectangle "); }
}

public class Square extends Rectangle { }

    public class Oval extends Shape {
      public void what() { System.out.print("Oval "); }
    }

    public class Circle extends Oval { }
```

What is the output from looping through the following array and calling what on each?
Shape[] shapes = {new Shape(), new Rectangle(), new Square(), new Circle()};

A) Shape Rectangle Rectangle Circle
B) Shape Shape Shape Shape
C) Shape Rectangle Square Circle

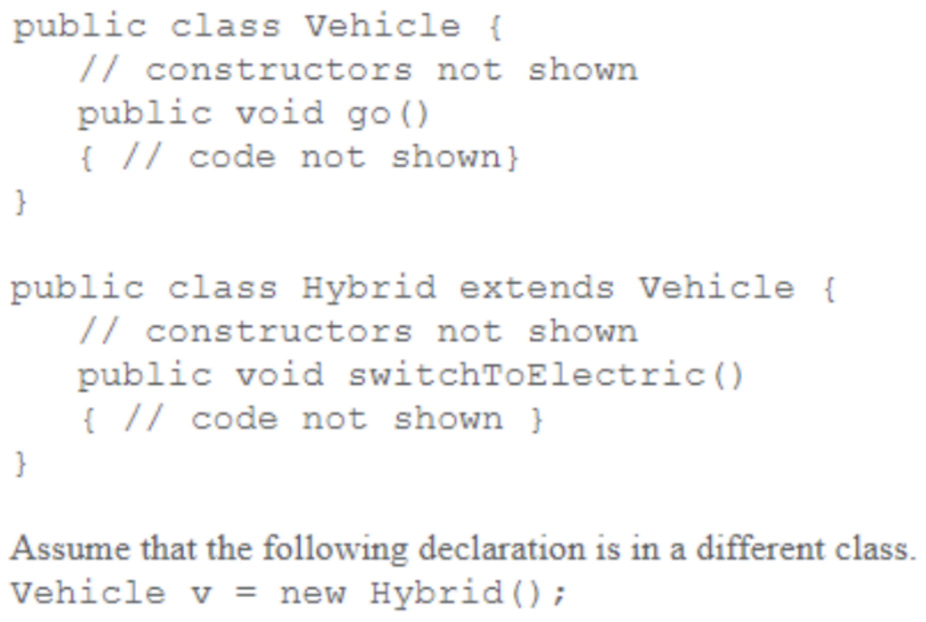
D) Shape Rectangle Rectangle Oval
E) There will be a compile time error

(16) Given the following class declarations, which of the following will cause a compile time error?

```
public class Car {
  private String make;

  public Car () {
     make = "Ford"; }

   public Car(String theMake) {
     make = theMake; }

   public String getMake() {
      return make;
   }
}

public class ElectricCar extends Car {
   public ElectricCar(String theMake) {
      super(theMake); }
}
```

A) Car myCar = new ElectricCar();
B) EletricCar myCar = new ElectricCar(“Ford”);
C) Car myCar = new ElectricCar(“Toyota”);
D) Car myCar = new Car(“Ford”);
E) Car myCar = new Car();

(17)



```
public class Vehicle {
   // constructors not shown
   public void go()
   { // code not shown}
}

public class Hybrid extends Vehicle {
   // constructors not shown
   public void switchToElectric()
   { // code not shown }
}
```

Assume that the following declaration is in a different class.

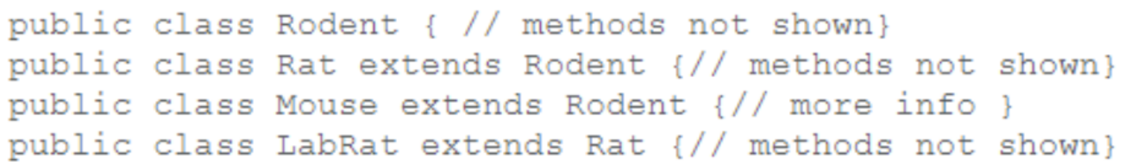
```
Vehicle v = new Hybrid();
```

Given the above class declarations, assume that the following declaration is in a different class.
Vehicle v = new Hybrid();
Which of the following will compile without error?

I. V.go();
II. V.switchToElectric();
III. ((Hybrid)v).switchToElectric();

A) I only
B) I, II and III
C) II and III only
[✓ D) I and III only]
E) II only

(18) Consider the following partial class definitions.



```
public class Rodent { // methods not shown}
public class Rat extends Rodent {// methods not shown}
public class Mouse extends Rodent {// more info }
public class LabRat extends Rat {// methods not shown}
```

Which of the following declarations would NOT cause a compile time error?
A) Rat rat = new Mouse();
B) LabRat labRat = new Rat();
C) Mouse mouse = new Rat();
[✓ D) Rodent rodent = new Rat();]
E) Rat rat = new Rodent();

(19) Consider the following class definitions:

```
public class Student {
  public String getFood() {
     return "Pizza";
  }
  public String getInfo()  {
     return "Studying";
  }
}
public class GradStudent extends Student {
  public String getFood() {
     return "Taco";
  }
  public String getInfo() {
     super.getInfo();
     return "Eating";
  }
}
```

What is printed when the following code is executed?

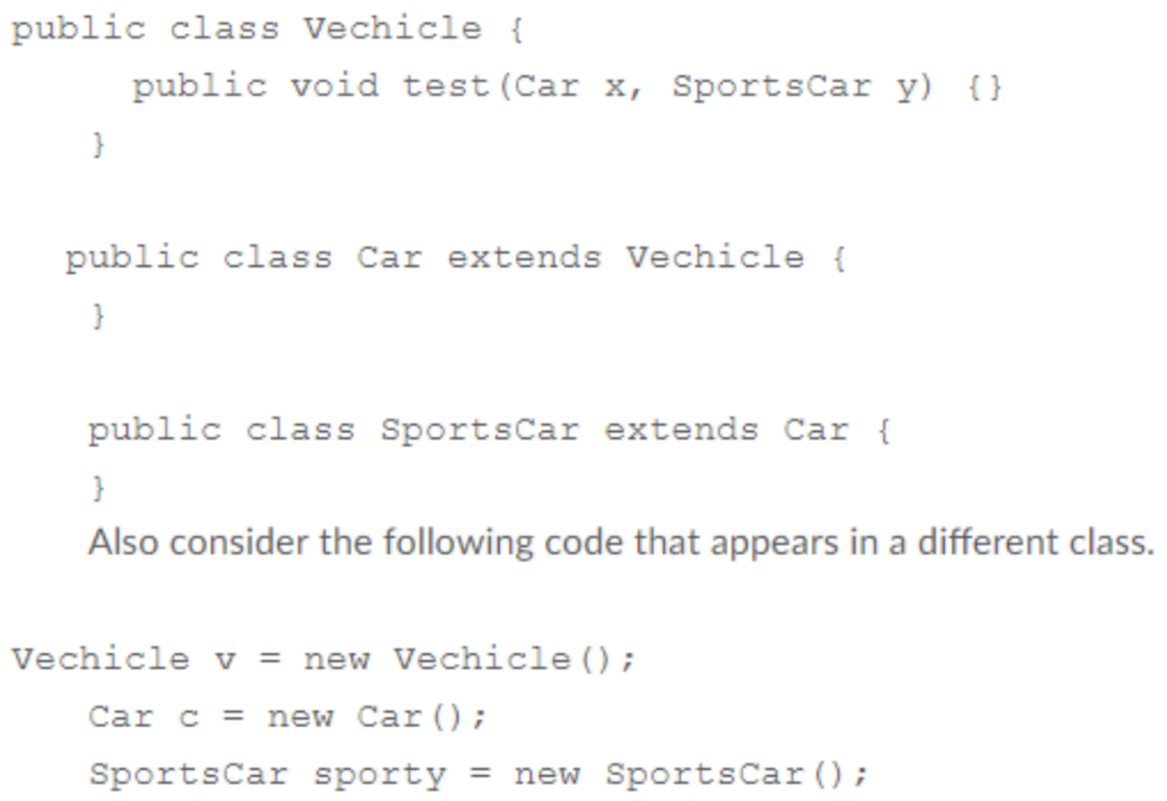
```
Student s = new GradStudent();

System.out.println(s.getInfo());
```

A) Studying
   Eating
B) Studying
C) Taco
D) Pizza
E) Eating

(20) Class C extends Class B which extends Class A. Also, all of the three classes implement a public method test(). How can a method in an object of Class C invoke the test() method defined in class A(without creating a new instance of class A)?

A) Test();
B) Super.super.test();
C) Super.test();
D) this.test();
E) There is no way to call a method in a grandparent class from a grandchild class

(21) Given the following declarations.



```
public class Vechicle {
      public void test(Car x, SportsCar y) {}
    }

   public class Car extends Vechicle {
    }

    public class SportsCar extends Car {
    }
```

Also consider the following code that appears in a different class.

```
Vechicle v = new Vechicle();
    Car c = new Car();
    SportsCar sporty = new SportsCar();
```

Which of the following is a correct call to test?

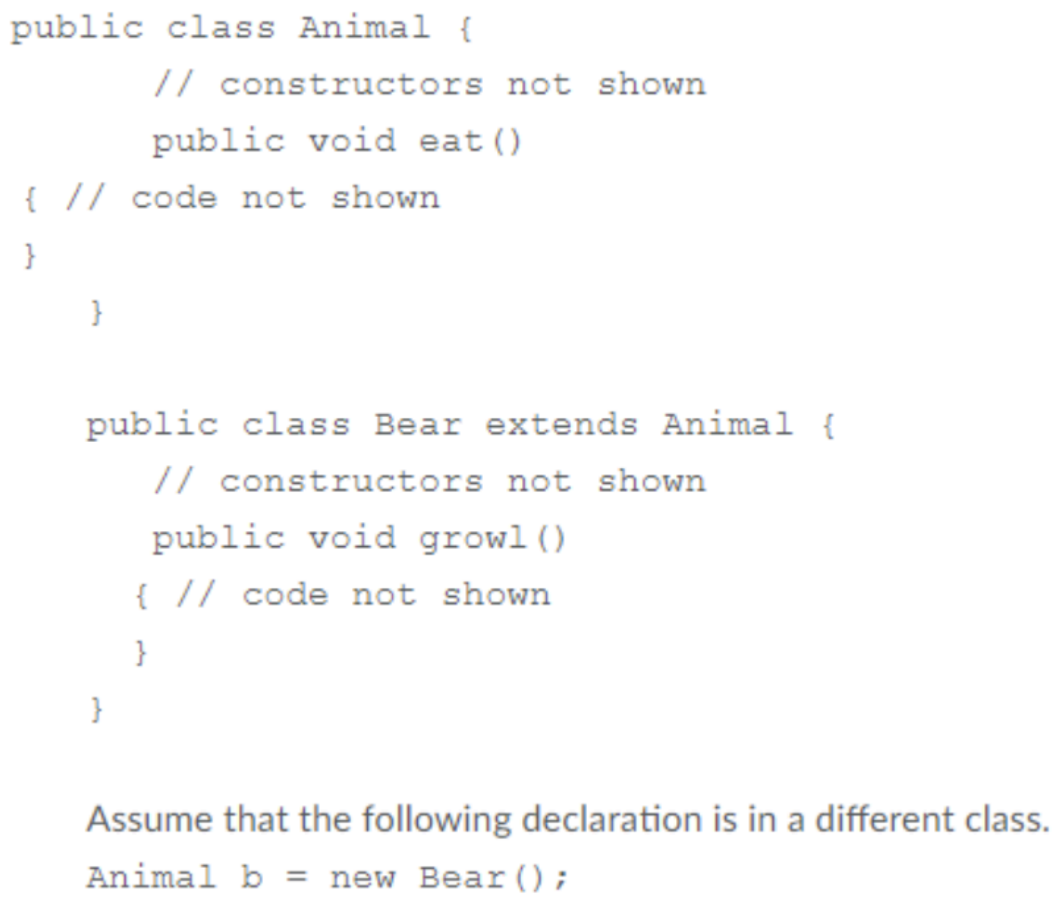
A) v.test(sporty,v);
B) sporty.test(c,c);
C) v.test(sporty, c);
D) sporty.test(sporty, v);
[✓ E) c.test(sporty, sporty);]

(22) Given the following class declarations, assume that Parent p = new Child() appears in a client program. What is the result of the call p.m1()?

```
public class Parent {
    public void m1() {
          System.out.print("pm1");
          m2();
    }
    public void m2() {
          System.out.print("pm2");
    }
 }

 public class Child extends Parent {
       public void m1()
       {
          super.m1();
          System.out.print("cm1");
       }
       public void m2()
       {
          super.m2();
          System.out.print("cm2");
       }
    }
```

[✓ A) pm1pm2cm2cm1]
B) pm1pm2
C) pm1pm2cm1cm2
D) pm1cm1
E) pm1

(23) Given the following class declarations



```
public class Animal {
      // constructors not shown
      public void eat()
 { // code not shown
 }
   }

   public class Bear extends Animal {
      // constructors not shown
      public void growl()
     { // code not shown
     }
   }
```

Assume that the following declaration is in a different class.

```
Animal b = new Bear();
```

Which of the following will compile without error?

I. b.eat();
II. b.growl();
III. ((Bear)b).growl();

A) I only
B) II only
C) III only
[✓ D) I and III only]
E) I, II and III

(24) Consider the following class declarations

```
public class C1 {
  private int num;
  private String name;

  public C1(int theNum) {
    num = theNum;
  }

  public C1(String theName) {
    name = theName;
  }
  // other methods not shown
}

public class C2 extends C1 {
   // methods not shown
}
```

Which of the following constructors are valid for C2?

```
I.  public C2 () { }
II. public C2 (int quan) {super (quan); }
III. public C2 (String label) { super(label); }
```

A) All three are valid
B) II only
C) III only
[✓ D) II and III]
E) None are valid

(25) Consider the following two classes

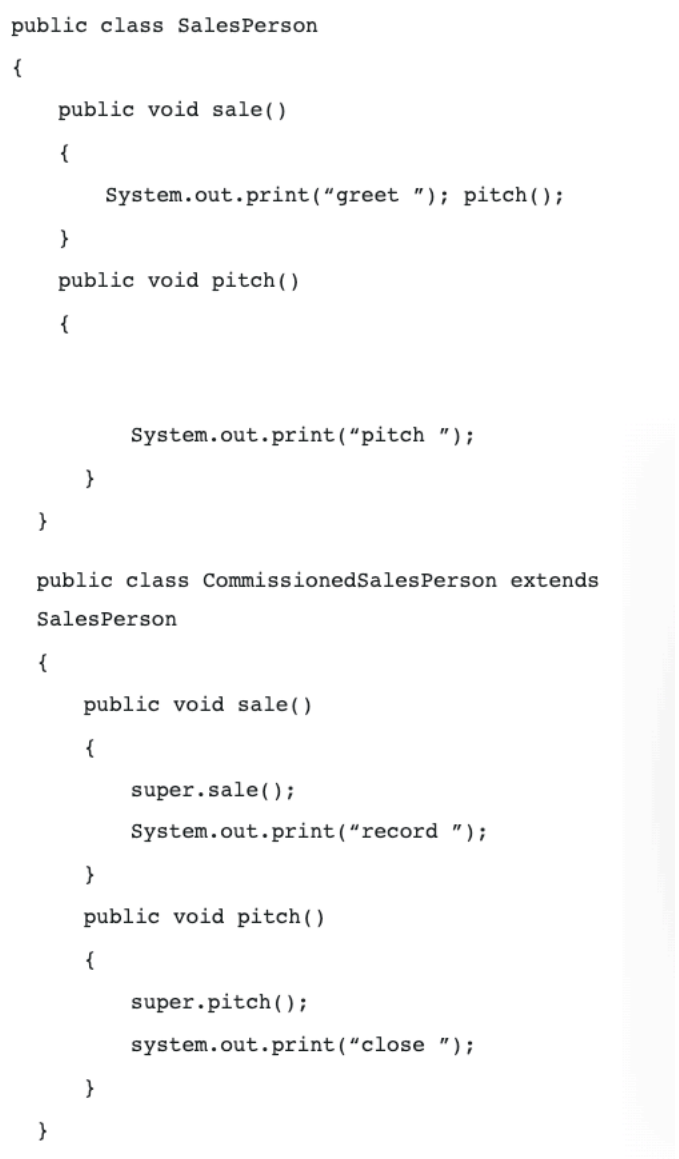
```
public class Bird
{
    public void act()
    {
        System.out.print("fly ");
        makeNoise();
    }

    public void makeNoise()
    {
        System.out.print("chirp ");
    }
}

public class Dove extends Bird
{
    public void act()
    {
        super.act();
        System.out.print("waddle ");
    }

    public void makeNoise()
    {
        super.makeNoise();
        System.out.print("coo ");
    }
}
```

Suppose the following declaration appears in a class other than Bird or Dove:
Bird pigeon = new dove();

What is printed as a result of the call pigeon.act()?

A) fly
B) fly chirp
C) fly chirp waddle
D) fly chirp waddle coo
[✓ E) fly chirp coo waddle]

(26) Consider the following two classe



```
public class SalesPerson
{
    public void sale()
    {
        System.out.print("greet "); pitch();
    }
    public void pitch()
    {


          System.out.print("pitch ");
      }
  }

  public class CommissionedSalesPerson extends
  SalesPerson
  {
      public void sale()
      {
          super.sale();
          System.out.print("record ");
      }
      public void pitch()
      {
          super.pitch();
          system.out.print("close ");
      }
  }
```

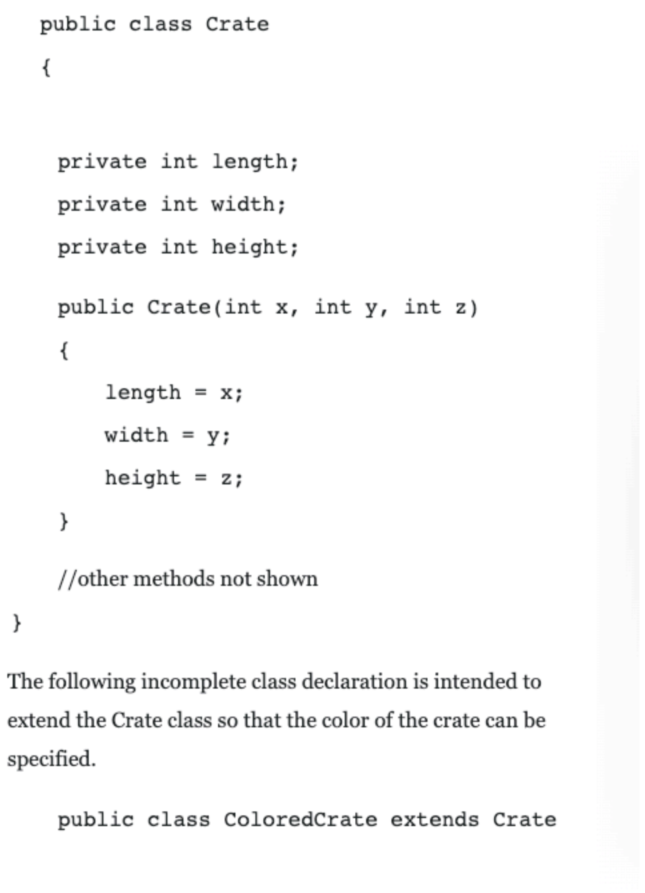
The following code segment is found in a class other than salesperson

SalesPerson vincent = new CommissionedSalesPerson();
Vincent.sale();

Which of the following is the best description of the functionality of this code segment?

A) greet pitch
B) greet pitch close
C) greet pitch record
D) greet pitch record close
[✓ E) greet pitch close record]

(27) Consider the following declaration of a class that will be used to represent dimensions of rectangular crates.



```
  public class Crate
  {

   private int length;
   private int width;
   private int height;

   public Crate(int x, int y, int z)
   {
       length = x;
       width = y;
       height = z;
   }

   //other methods not shown
}
```

The following incomplete class declaration is intended to extend the Crate class so that the color of the crate can be specified.

```
   public class ColoredCrate extends Crate
```

```
{
    private String color;
    //Constructors not shown
    //Other methods not shown
}
```

Which of the following possible constructors for `ColoredCrate` would be considered legal?

```
 I. public ColoredCrate(int a, int b, int c,
    String crateColor)
    {
      length = a;
      width = b;
      height = c;
      color = crateColor;
    }
II. public ColoredCrate(int a, int b, int c,


      String crateColor)
      {
        super (a, b, c)
        color = crateColor;
      }
  III. public ColoredCrate()
      {
        color = “”;
      }
```

A) I only
B) III only
C) I and II only
D) I and III only
E) II and III only

Questions 28-31 refer to the following class definition

```
public class Depot
{
   private String city;
   private String state;
   private String country = "USA";
   private boolean active = true;

   public Depot(String myCity, String myState, String myCountry, boolean myActive)
   {
      city = myCity;
      state = myState;
      country = myCountry;
      active = myActive;
   }

   public Depot(String myState, String myCity)
   {
      state = myState;
      city = myCity;
   }

   public Depot(String myCity, String myState, boolean myActive)
   {
      city = myCity;
      state = myState;
      active = myActive;
   }

   public String toString()
   {
      return city + ", " + state + " " + country + " " + active;
   }

   /* Additional implementation not shown */
}
```

(28) The Depot class has three constructors. Which of the following is the correct term for this practice?

A) Overriding
B) Procedural abstraction
C) Encapsulation
D) Polymorphism
E) [✓ Overloading]

(29) Consider the following code segment in another class.

```
Depot station = new Depot("Oakland", "California");
System.out.println(station);
```

What is printed as a result of executing the code segment?

A) [✓ California, Oakland USA true]
B) California, Oakland USA active
C) USA, California USA true
D) Oakland, California USA active
E) Oakland, California USA true

(30) Consider the following class definition

public class WhistleStop extends Depot

Which of the following constructors compiles without error?

I.
```
public WhistleStop()
{
    super();
}
```

II.
```
public WhistleStop()
{
    super("Waubaushene", "Ontario", "Canada", true);
}
```

III.
```
public WhistleStop(String city, String province)
{
    super(city, province);
}
```

A) I only
B) II only
C) III only
[✓ D) II and III only]
E) I, II and III

(31) Assume a correct no-argument constructor has been added to the WhistleStop class. Which of the following code segments compiles without error?

I.
```
ArrayList<Depot> stations = new ArrayList<Depot>();
stations.add(new WhistleStop());
stations.add(new Depot("Mansonville", "Quebec", "Canada", false));
```

II.
```
ArrayList<WhistleStop> stations = new ArrayList<Depot>();
stations.add(new WhistleStop());
stations.add(new Depot("Orinda", "California", true));
```

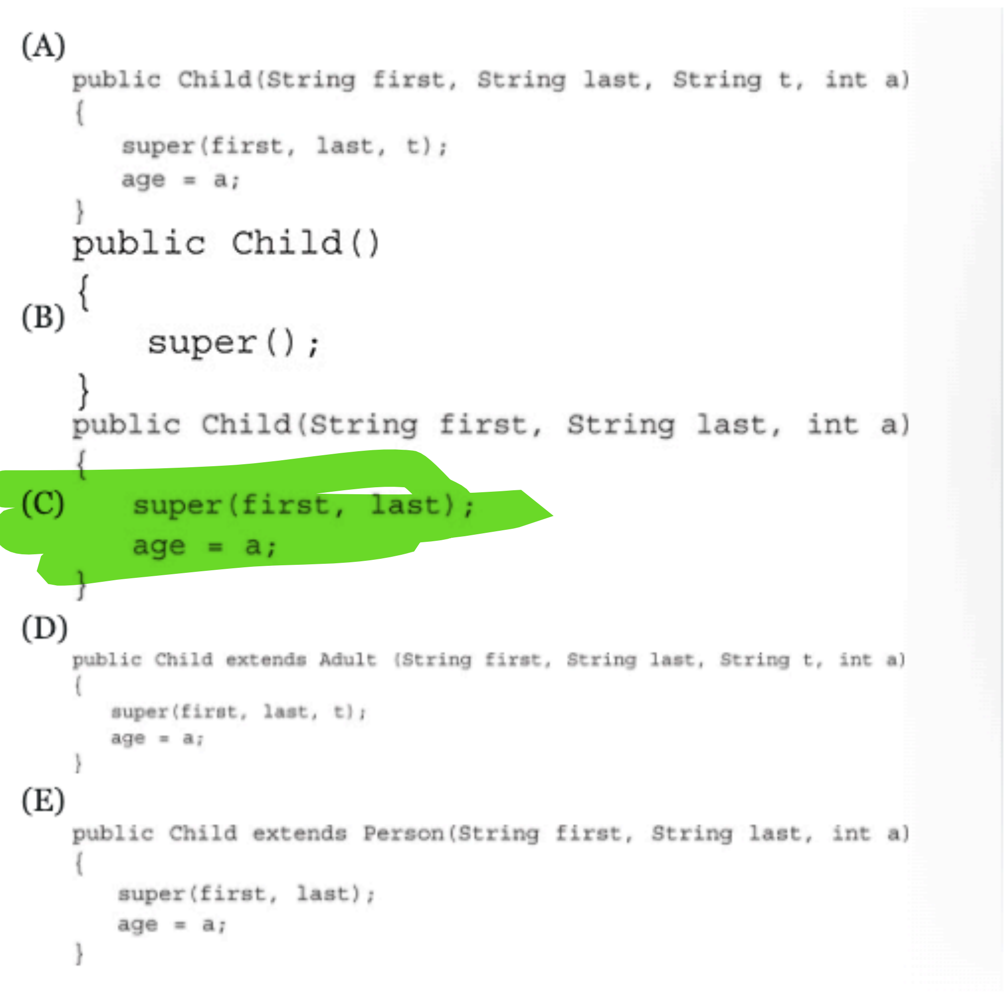
III.
```
ArrayList<WhistleStop> stations = new ArrayList<WhistleStop>();
stations.add(new WhistleStop());
stations.add(new Depot("Needham", "Massachusetts", true));
```

[✓ A) I only]
B) I and II only
C) I and III only
D) II and III only
E) I, II and III

Questions 32-35 refer to the following class

```
public class Person
{
   private String firstName;
   private String lastName;

   public Person(String fName, String lName)
   {
      firstName = fName;
      lastName = lName;
   }

   public String getName()
   {
      return firstName + " " + lastName;
   }
}

public class Adult extends Person
{
   private String title;

   public Adult(String fname, String lName, String myTitle)
   {
      super(fname, lName);
      title = myTitle;
   }

   /*  toString method to be implemented in question 27  */
}

public class Child extends Person
{
   private int age;

   /*  Constructor to be implemented in question 26  */
}
```

(32) Which of the following is a correct implementation of a child class constructor?



(A)

```
public Child(String first, String last, String t, int a)
{
   super(first, last, t);
   age = a;
}
```

(B)

```
public Child()
{
   super();
}
```

[✓ (C)]

```
public Child(String first, String last, int a)
{
   super(first, last);
   age = a;
}
```

(D)

```
public Child extends Adult (String first, String last, String t, int a)
{
   super(first, last, t);
   age = a;
}
```

(E)

```
public Child extends Person(String first, String last, int a)
{
   super(first, last);
   age = a;
}
```

(33) Which of the following is a correct implementation of the Adult class toString method?

(A)
```
public String toString()
{
    System.out.println(title + " " + super.toString());
}
```
(B)
```
public String toString()
{
    return title + " " + super.toString();
}
```
(C)
```
public String toString()
{
    return title + " " + firstName + " " + lastName;
}
```
(D)
```
public void toString()
{
    System.out.println(title + " " + super.getName());
}
```
[✓ (E)]
```
public String toString()
{
    return title + " " + super.getName();
}
```

(34) Consider the following method declaration in a different class

```
public void findSomeone(Adult someone)
```

Assume the following variables have been correctly instantiated and initialized with appropriate values.

```
Person p;
Adult a;
Child c;
```

Which of the following method calls compiles without error?

I. `findSomeone(p);`

II. `findSomeone(a);`

III. `findSomeone(c);`

IV. `findSomeone((Adult) p);`

V. `findSomeone((Adult) c);`

A) II only
B) I and II only
[✓ C) II and IV only]

D) II, IV and V only
E) II, III and IV only

Questions 35-36 refer to the following two classes.

```
public class Vehicle
{
   private int fuel;

   public Vehicle(int fuelAmt)
   {
      fuel = fuelAmt;
   }

   public void start()
   {
      System.out.println("Vroom");
   }

   public void changeFuel(int change)
   {
      fuel = fuel + change;
   }

   public int getFuel()
   {
      return fuel;
   }
}
public class Boat extends Vehicle
{
   public Boat(int fuelAmount)
   {
       super(fuelAmount);
   }

   public void useFuel()
   {
      super.changeFuel(-2);
   }

   public void start()
   {
      super.start();
      useFuel();
      System.out.println("Remaining Fuel " + getFuel());
   }
}
```

(35) Assume the following declaration appears in a client program.

Boat yacht = new Boat(20);

What is printed as a result of executing the call yacht.start();

(A) `Vroom`

(B) `Remaining Fuel 18`

(C) `Remaining Fuel 20`

[✓ (D) `Vroom`
`Remaining Fuel 18`]

(E) `Vroom`
`Remaining Fuel 20`

(36) Which of the following statements results in a compile-time error?

```
I.   Boat sailboat = new Boat(2);
II.  Vehicle tanker = new Boat(20);
III. Boat tugboat = new Vehicle(10);
```

A) I only
B) II only
[✓ C) III only]
D) II and III only
E) None of these options will result in a compile time error

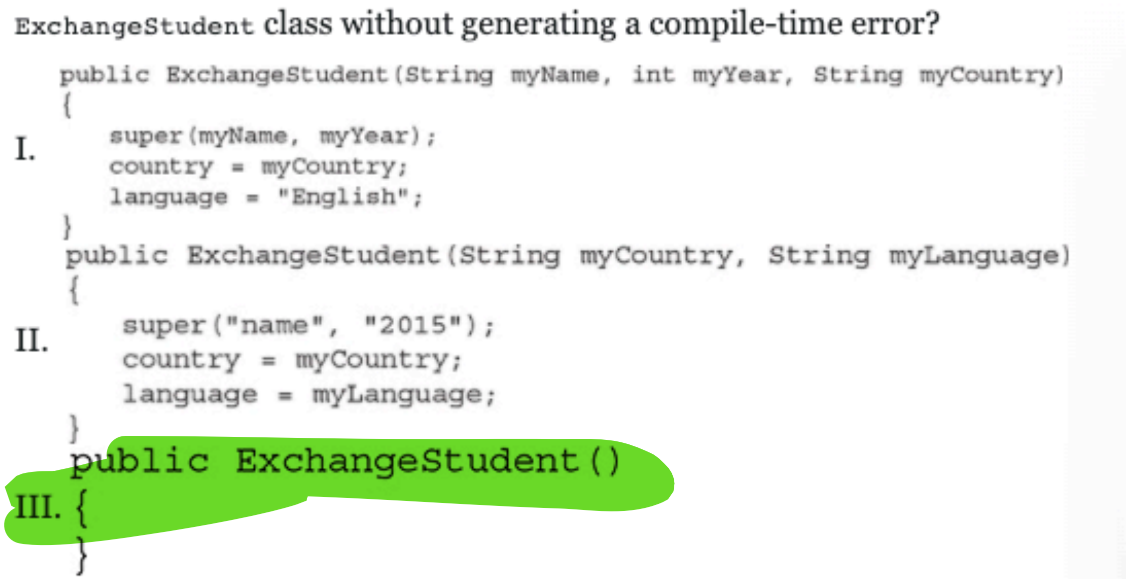
(37) Consider the following class used to represent a Student

```
public class Student
{
   private String name;
   private int year;

   public Student()
   {
      name = "name";
      year = 0;
   }

   public Student(String myName, int myYear)
   {
      name = myName;
      year = myYear;
   }
}
```

Consider the `ExchangeStudent` class that extends the `Student` class.

```
public class ExchangeStudent extends Student
{
   private String country;
   private String language;

   public ExchangeStudent(String myName, int myYear,
                          String myCountry, String myLanguage)
   {
      super(myName, myYear);
      country = myCountry;
      language = myLanguage;
   }
}
```

Which of the following constructors could be included in the ExchangeStudent class without generating a compile-time error?


`ExchangeStudent` class without generating a compile-time error?

I.
```
public ExchangeStudent(String myName, int myYear, String myCountry)
{
   super(myName, myYear);
   country = myCountry;
   language = "English";
}
```

II.
```
public ExchangeStudent(String myCountry, String myLanguage)
{
   super("name", "2015");
   country = myCountry;
   language = myLanguage;
}
```

III.
```
public ExchangeStudent()
{
}
```

A) I only
B) II only
C) III only
D) I and III only
E) I, II and III

Questions 38-39 refer to the following classes.
Consider the following class declarations

```
public class Building
{
   private int sqFeet;
   private int numRooms;

   public Building(int ft, int rms)
   {
      sqFeet = ft;
      numRooms = rms;
   }

   public int getSqfeet()
   {  return sqFeet;  }

   public String getSize()
   {
      return numRooms + " rooms and " + sqFeet + " square feet";
   }

   /* Additional implementation not shown */
}

public class House extends Building
{
   private int numBedRooms;
   public House(int ft, int rms, int bedrms)
   {
      super(ft, rms);
      numBedRooms = bedrms;
   }

   public String getSize()
   {
      return super.getSqFeet() + " square feet and " + numBedRooms + " bedrooms.";
   }
}
```

(38) Assume that ArrayList<Building> list has been correctly instantiated and populated with Building objects. Which of the following code segments will result in the square feet in each building being printed?

I.
```
for (int i = 0; i < list.size(); i++)
    System.out.println(list.getSqFeet(i));
```

II.
```
for (int i = 0; i < list.size(); i++)
    System.out.println(list[i].getSqFeet());
```

III.
```
for (int i = 0; i < list.size(); i++)
    System.out.println(list.get(i).getSqFeet());
```
IV.
```
for (Building b : list)
    System.out.println(list.getSqFeet());
```
V.
```
for (Building b : list)
    System.out.println(b.getSqFeet());
```

A) I and IV only
B) I and V only
C) II and IV only
D) III and IV only
[✓ E) III and V only]

(39) Consider the following code segment

```
Building b1 = new Building(2000, 3);
Building b2 = new House(2500, 8, 4);
Building b3 = b2;
Building[] buildings = new Building[3];
buildings[0] = b1;
buildings[1] = b2;
buildings[2] = b3;
```

What will be printed by the following code segment?

```
for (int i = 0; i < buildings.length; i++)
    System.out.println(buildings[i].getSize());
```

(A)
```
3 rooms and 2000 square feet
8 rooms and 2500 square feet
8 rooms and 2500 square feet
```

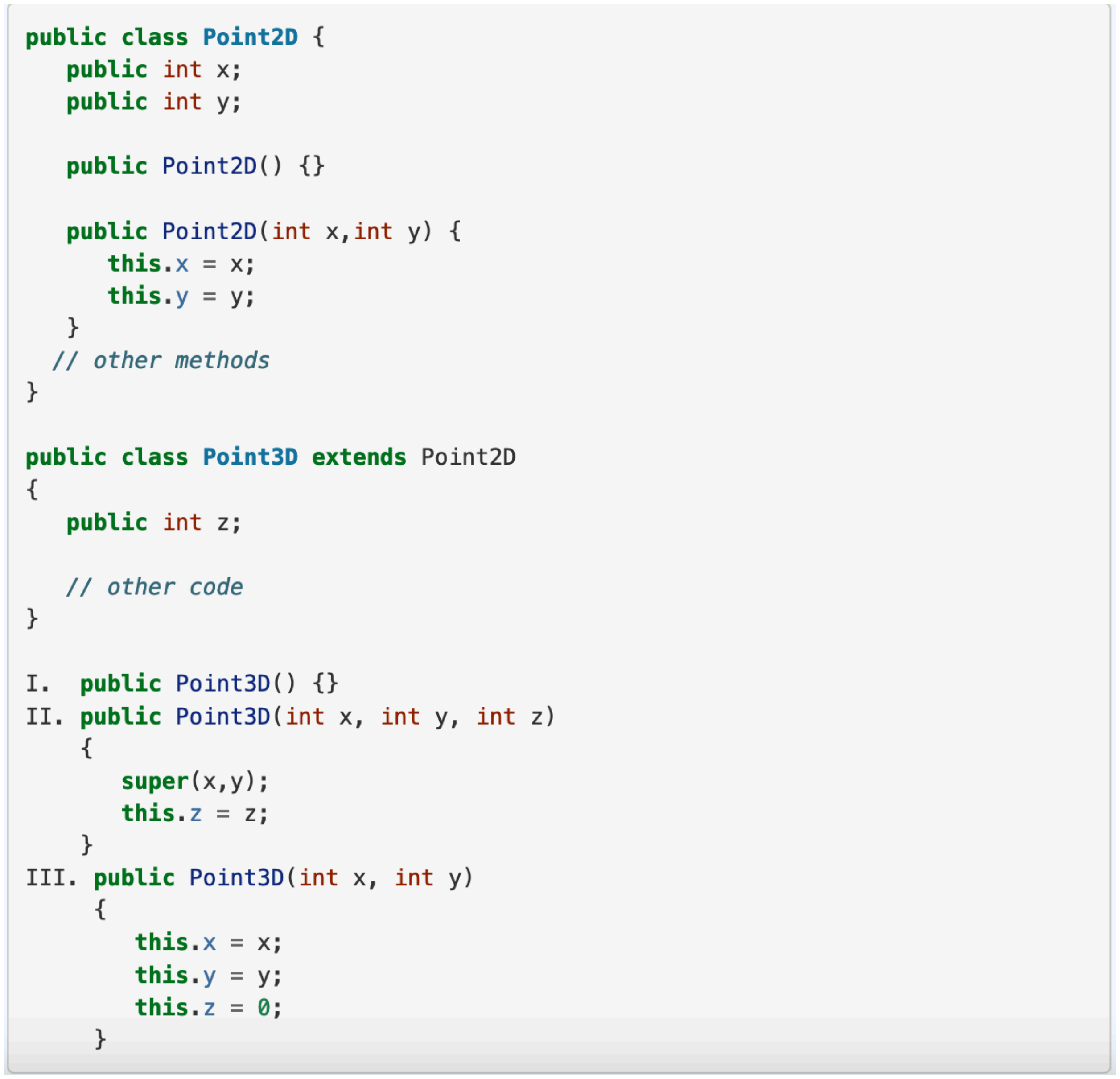
[✓ (B)]
```
3 rooms and 2000 square feet
2500 square feet and 4 bedrooms
2500 square feet and 4 bedrooms
```

(C)
```
2000 square feet and 3 bedrooms
2500 square feet and 4 bedrooms
2500 square feet and 4 bedrooms
```

(D)
```
3 rooms and 2000 square feet
2500 square feet and 4 bedrooms
3 rooms and 2000 square feet
```

(E) There is an error. Nothing will be printed.

(40) If you have the following classes, which of the following constructors would be valid for Point3D?



```java
public class Point2D {
   public int x;
   public int y;

   public Point2D() {}

   public Point2D(int x,int y) {
      this.x = x;
      this.y = y;
   }
  // other methods
}

public class Point3D extends Point2D
{
   public int z;

   // other code
}

I.  public Point3D() {}
II. public Point3D(int x, int y, int z)
    {
       super(x,y);
       this.z = z;
    }
III. public Point3D(int x, int y)
     {
        this.x = x;
        this.y = y;
        this.z = 0;
     }
```

A) II only
B) III only
[✓ C) I, II and III]
D) I and II only
E) I only