29) Consider the code segment.

```
public class Boo
{
  public Boo(String s) { /* implementation not shown */ }
  public Boo() { /* implementation not shown */ }
}


public class Bar extends Boo
{
  public Bar() { /* implementation not shown */ }
  public Bar(String s)
  {
    super(s);
  }

  public void zoo()
  {
    // missing code
  }
}
```

Which of the following code segments can be used to instantiate a object?

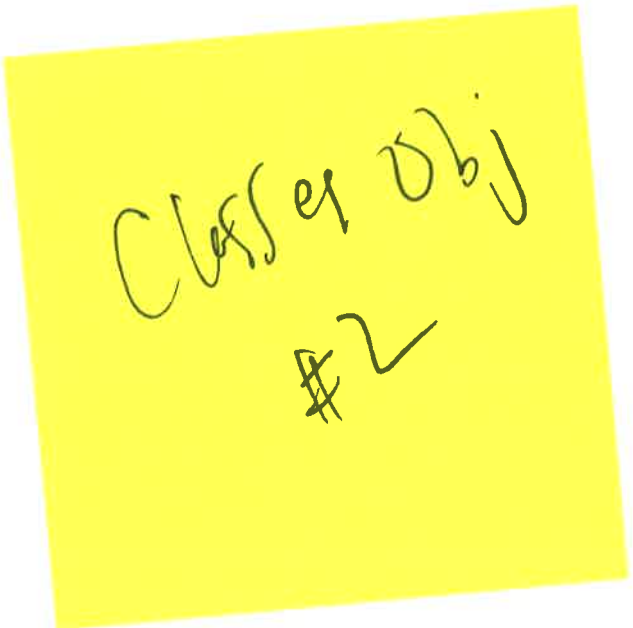
(A) `Boo f = new Bar();`

(B) `Bar f = new Boo(String s);`

(C) `Bar f = new Boo;`

(D) `Bar f = Boo(String s);`

(E) None of the above



30) Consider the following code.

```
public void add(int x)
{
  x = x + 1;

}
public void add(int x, int y)
{
  x = x + y;
}
```

The above code is an example of which of the following?

(A) Method overriding

(B) Method overloading

(C) Inheritance

(D) Encapsulation

(E) Implementation

For Questions 3 - 5, refer to following student class

```
public class Employee
{
  private String name;
  private double salary;
  private boolean isManager;
  public Employee(String name, double salary, boolean isMang)
  {
    // missing code
  }

  /* Setter and getter methods implementation not shown */


  public void displayable()
  {
    // missing code
  }

  public Employee salincrement(Employee E)
  {
    // missing code
  }
}
```

3) Which of the following code segments is a valid way of instantiating an Employee object?

I.
```
Employee E = new Employee();
```

II.
```
String n = "Tom";
double s = 20000;
boolean is_m = true;
Employee E = new Employee(n, s, is_m);
```

III.
```
Employee E = new Employee(String name,boolean ismanager,double salary);
```

(A) I only

(B) II only

(C) I and II

(D) I and III

(E) III only

4) Consider the following implementation of the constructor of the Employee class.

```
public Employee(String name, double salary, boolean isMang)
{
  this.name = name;
  this.salary = salary;
  this.isManager = isMang;
}
```

Which of the following code segments is an appropriate implementation of the displayable() method if its purpose is to print out the details to Employee?

(A)
```
System.out.print(name + " " + salary + " " + isManager + " ");
```

(B)
```
System.out.print(Employee.name + " ");
System.out.print(Employee.salary + " ");
System.out.print(Employee.isManager + " ");
```

(C)
```
System.out.println(E.getName() + " " + E.getSalary());
```

(D)
```
System.out.println(Employee.getName() + " " + Employee.getSalary());
```

(E) None of the above

5) Consider the following implementation for the `salincrement()` method.

```
public Employee salincrement(Employee E)
{
  if (E.isManager == true)
  {
    double sal =  E.getSalary() * 2;
    E.setSalary(sal);
    return E;
  }
  else
  {
    return E;
  }
}
```

If the code below is executed, what will the output be?

```
public static void main(String[] args)
{
  Employee E = new Employee("Tom", 200000, true);
  E.displayable();
  Employee E_DUP = E.salincrement(E);
  E_DUP.displayable();
}
```

(A) `Tom 200000.0 true`

(B) `Tom 200000.0 true Tom 400000.0 true`

(C) `Tom 200000 true Tom 400000 true`

(D) Compile time error

(E) None of the above

7) Consider the following classes.

```
public class A
{
  int x = 10;
  int y = 20;

  public void display_A()
  {
    System.out.println(x + " " + y);
  }
}
```

```
public class B
{
  int x = 20;
  int y = 10;

  public void display_B()
  {
    System.out.println(x + " " + y);
  }
}
```

```
public class C extends A, B
{
  int x = 30;
  int y = 40;
  A a = new A();
  a.display_A();
}
```

What will be the output for the above code?

(A) Run time error

(B) `10 20`

(C) Compile time error

(D) `30 40`

(E) None of the above

For Questions 13 - 15, refer to the following classes.

```
public class Author
{
  private String name;
  private int noofpub;

  public Author(String name, int noofpub)
  {
    // missing code
  }
  public String getName()
  {
    // missing code
  }
  public int getNoofpub()
  {
    // missing code
  }
}

public class Book
{
  private String title;
  private double price;
  private Author author;

  public Book(String title, double price)
  {
    /* Implementation */
  }
  public String getTitle()
  {
    /* Implementation */
  }
  public double getPrice()
  {
    /* Implementation */
  }

  /* Other methods (not shown) */
}
```

13) Consider the following implementation of the Book constructor.

```
public Book(String title, double price)
{
  this.title = title;
  this.price = price;
  this.author = new Author("Steven", 2);
}
```

What is the relationship between the Book and the Author classes?

(A) Inheritance

(B) Composition

(C) Aggregation

(D) Polymorphism

(E) None of the above

14) Consider the following implementation of the Book constructor.

```
public Book(String title, double price)
{
  this.title = title;
  this.price = price;
  this.author = new Author("Steven", 2);
}
```

Which of the following code segments is a valid way of creating an Author object?

I. `Author A = new Author("Tom", 3);`

II. `Book B = new Author("Tom", 3);`

III. `Book B = new Book("xyz", 35.0);`

(A) I only

(B) II only

(C) I and II

(D) I and III

(E) III only

22) Consider the following code.

```
public class Round extends Shape
public class Rectangle extends Shape

public class Oval extends Round
public class Square extends Rectangle
```

Assuming each class has a default constructor, which of the following declarations are valid, based on the code above?

I.
```
Shape S = new Round();
Round R = new Oval();
```

II.
```
Shape S = new Oval();
Shape S1 = new Square();
Square S2 = new Square();
```

III.
```
Square S = new Rectangle();
```

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

24) Which of the following relationships represents polymorphism?
(A) HAS-A
(B) HAS-HAS
(C) IS-A
(D) MANY-TYPES
(E) None of the above

26) Which of the following statement is true about inheritance?
(A) Inheritance is shown by using `implements` keyword.
(B) One class can inherit two classes.
(C) The `super` keyword can be used to inherit the superclass constructor.
(D) Inheritance is not commonly used in Java programming.
(E) All of the above are false.

33) Refer to the following classes.

```
public class A
{
  int i = 100;
}


public class B extends A
{
  int i = 101;
}
```

What will be the output if the following code is executed?

```
public static void main(String[] args)
{
  A a = new B();
  B b = new B();
  System.out.println(a.i + " " + b.i);
}
```

(A) `100 100`

(B) `101 100`

(C) `100 101`

(D) `101 101`

(E) None of the above

37) Refer to the following class.

```
public class X
{
  public methodA(int x)
  {
    System.out.println("x : " + x);
  }

  public methodA(int x, int y)
  {
    System.out.println("x : " + x + " " + y);
  }
}
```

Which of the following concepts does the method represent?

(A) Method overriding

(B) Method overloading

(C) Inheritance

(D) Polymorphism

(E) B and D

39) Refer to the following classes.

```
public class X
{
  public void methoda(int a)
  {
    System.out.println("Integer" + " ");
  }
  public void methoda(double d)
  {
    System.out.println("Double" + " ");
  }
}


public class Y extends X
{
  public void methoda(double d)
  {
    System.out.println("Double in class Y" + " ");
  }
}
```

What will be the output if the following code is executed?

```
public static void main(String[] args)
{
  Y y = new Y();
  y.methoda(100);
  y.methoda(100.0);
}
```

(A) `100 100.0`

(B) `Integer Double`

(C) `Integer Double in class Y`

(D) `Double Double in class Y`

(E) `Double in class Y Integer`

7. Given two classes, `Animal` and `Mammal`, which of the following situations would make the statement

```
Animal a = new Mammal("Elephant");
```

valid?

(A) `Mammal` extends `Animal` and `Mammal` has a constructor with one parameter of the `String` type.
(B) `Mammal` extends `Animal` and `Animal` has a constructor with one parameter of the `String` type.
(C) `Animal` has a method `Mammal` that takes one parameter of the `String` type.
(D) `Animal` has a public data member `String Mammal`.
(E) None of the above

11. Suppose a class `Particle` has the following variables defined:

```
public class Particle
{
  public static final int STARTPOS = 100;
  private double velocity;
  < other code not shown >
}
```

Which of the following is true?

(A) `velocity` can be passed as an argument to one of `Particle`'s methods, but `STARTPOS` cannot.
(B) Java syntax rules wouldn't allow us to use the name `startPos` instead of `STARTPOS`.
(C) A statement `double pos = STARTPOS + velocity;` in one of `Particle`'s methods would result in a syntax error.
(D) Java syntax rules wouldn't allow us to make `velocity` public.
(E) A statement `STARTPOS += velocity;` in one of `Particle`'s methods would result in a syntax error.

18. Consider the following method from `ClassX`:

```
private int modXY(int x, int y)
{
  r = x / y;
  return x % y;
}
```

If `ClassX` compiles with no errors, which of the following statements must be true?

I. `modXY` has a side effect since `r` is not a local variable in `modXY`.
II. `r` must be an instance variable in the superclass of `ClassX`.
III. `r` must have the type `double`.

(A) I only
(B) II only
(C) III only
(D) I and II
(E) None

20. Consider the following class definitions:

```
public class Country
{
  public String toString() { return "Country"; }
}

public class Brazil extends Country
{
  public String toString() { return "Brazil"; }
}
```

What is the output of the following code segment?

```
Country country1 = new Country();
Country country2 = new Brazil();
System.out.print(country1 + " " + country2 + " ");
country1 = country2;
System.out.print(country1);
```

(A) Country Country Country
(B) Country Country Brazil
(C) Country Brazil Country
(D) Country Brazil Brazil
(E) Brazil Brazil Brazil

Questions 24-26 involve reasoning about classes and objects used in an implementation of a library catalog system. An object of the class `BookInfo` represents information about a particular book, and an object of the class `LibraryBook` represents multiple copies of a book on the library's shelves:

```
public class BookInfo
{
  private String myTitle;
  private String myAuthor;
  private int myNumPages;

  < Constructors not shown >

  public String toString()
  {
    return myTitle + " by " + myAuthor;
  }

  public String getTitle() { return myTitle; }
  public int getNumPages() { return myNumPages; }
}

public class LibraryBook
{
  private BookInfo myInfo;
  private int myNumCopies;   // Number of copies on shelf

  < Constructors not shown >

  public int getNumCopies() { return myNumCopies; }
  public void setNumCopies(int numCopies)
                          { myNumCopies = numCopies; }
  public BookInfo getInfo() { return myInfo; }

  // postcondition: if there are copies on shelf, decrements
  //                the number of copies left and returns true;
  //                otherwise returns false
  public boolean checkOut() { < code not shown > }
}
```

24. If `catalog` is declared in a client class as

```
LibraryBook [] catalog;
```

which of the following statements will correctly display *title* by *author* of the third book in `catalog`?

I.
```
System.out.println(catalog[2]);
```

II.
```
System.out.println(catalog[2].getInfo());
```

III.
```
System.out.println(catalog[2].getInfo().toString());
```

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

25. Which of the following code segments will correctly complete the `checkOut()` method of the `LibraryBook` class?

I.
```
if (getNumCopies() == 0)
  return false;
else
{
  setNumCopies(getNumCopies() - 1);
  return true;
}
```

II.
```
int n = getNumCopies();
if (n == 0)
  return false;
else
{
  setNumCopies(n - 1);
  return true;
}
```

III.
```
if (myNumCopies == 0)
  return false;
else
{
  myNumCopies--;
  return true;
}
```

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

26. Consider the following method from another class, a client of `LibraryBook`:

```
// postcondition: returns the total number of pages in
//                all books in catalog that are on the shelves
public int totalPages(LibraryBook [] catalog)
{
  int count = 0;
  int k;

  for (k = 0; k < catalog.length; k++)
  {
    < statement >
  }
  return count;
}
```

Which of the following replacements for < *statement* > completes the method as specified?

(A) `count += catalog[k].myNumCopies * catalog[k].myInfo.numPages;`
(B) `count += catalog[k].getNumCopies() * catalog[k].getNumPages();`
(C) `count += catalog[k].(myNumCopies * myInfo.getNumPages());`
(D) `count += catalog[k].getNumCopies() * catalog[k].getInfo().getNumPages();`
(E) None of the above

29. Brad has derived his class from a library class `JPanel`. `JPanel`'s `paintComponent` method displays a blank picture in a panel. Brad redefined `JPanel`'s `paintComponent` to display his own picture. Brad's class compiles with no errors, but when he runs the program, only a blank background is displayed. Which of the following hypotheses CANNOT be true in this situation?

(A) Brad misspelled "paintComponent" in his method's name.
(B) Brad specified an incorrect return type for his `paintComponent` method.
(C) Brad chose the wrong type for a parameter in his `paintComponent` method.
(D) Brad specified two parameters for his `paintComponent` method, while `JPanel`'s `paintComponent` takes only one parameter.
(E) Brad has a logic error in his `paintComponent` code which prevents it from generating the picture.

```
public class SortX
{
  public static void sort(Comparable[] items)
  {
    int n = items.length;
    while (n > 1)
    {
      sortHelper(items, n - 1);
      n--;
    }
  }

  private static void sortHelper(Comparable[] items, int last)
  {
    int k, m = last;
    for (k = 0; k < last; k++)
    {
      if (items[k].compareTo(items[m]) > 0)
        m = k;
    }
    Comparable temp = items[m];
    items[m] = items[last];
    items[last] = temp;
  }
}
```

32. The sorting algorithm implemented in the `sort` method can be best described as:

(A) Selection Sort
(B) Insertion Sort
(C) Quicksort
(D) Mergesort
(E) Incorrect implementation of a sorting algorithm

33. Suppose `names` is an array of `String` objects:

```
String[] names = {"Dan", "Alice", "Claire", "Evan", "Boris"};
```

If `SortX.sort(names)` is running, what is the order of the values in `names` after two complete iterations through the `while` loop in the `sort` method?

(A) `"Boris", "Alice", "Claire", "Dan", "Evan"`
(B) `"Alice", "Claire", "Boris", "Dan", "Evan"`
(C) `"Alice", "Boris", "Claire", "Evan", "Dan"`
(D) `"Alice", "Claire", "Dan", "Evan", "Boris"`
(E) None of the above

34. If `items` contains five values and `SortX.sort(items)` is called, how many times, total, will `items[k].compareTo(items[m])` be called in the `sortHelper` method?

(A) 5
(B) 10
(C) 15
(D) 25

6. Consider the following class:

```
public class Rectangle
{
  private int width, height;

  public Rectangle(int w, int h) { width = w; height = h; }
  public int getHeight() { return height; }
  public int getWidth() { return width; }
  public int getArea() { return width * height; }
}
```

Suppose we want to modify this class to make it implement the `Comparable` interface. `Rectangle` objects should be compared based on their area: the rectangle with the smaller area should be deemed smaller. Which of the following methods do we have to add?

(A)

```
public boolean compareTo(int area)
{
  return getArea() < area;
}
```

(B)

```
public int compareTo(int area)
{
  return getArea() - area;
}
```

(C)

```
public boolean compareTo(Rectangle other)
{
  return getArea() < other.getArea();
}
```

(D)

```
public int compareTo(Rectangle other)
{
  return getArea() - other.getArea();
}
```

(E)

```
public int compareTo(Object other)
{
  return getArea() - ((Rectangle)other).getArea();
}
```

(A) Overloaded methods must be made either all public or all private.
(B) Overloaded methods are defined in the same class and have the same name.
(C) Overloaded methods may have the same number of parameters.
(D) One of the overloaded methods may take no parameters.
(E) Overloaded methods cannot be differentiated based only on the names chosen for their parameters.

10. Consider the following class:

```
public class Sphere
{
  public static final double pi = 3.14159;

  public static double volume(int r)
  {
    return 4 / 3 * pi * Math.pow(r, 3);
  }
}
```

Which of the following statements about this code is true?

(A) The class will not compile because no constructors are defined.
(B) The class will not compile because `pi` cannot be declared `public`.
(C) The class will not compile because the `volume` method is declared `static`.
(D) `Math.pow(r, 3)` cannot be used because `r` is an `int`.
(E) The class compiles with no errors but the `volume` method returns a significantly smaller value than the expected $\frac{4}{3}\pi r^3$.

20. The class `PlayList` provides methods that allow you to represent and manipulate a list of tunes, but you are not concerned with how these operations work or how the list is stored in memory. You only know how to initialize and use `PlayList` objects and have no direct access to the implementation of the `PlayList` class or its private data members. This is an example of:

(A) encapsulation
(B) overriding
(C) inheritance
(D) dynamic binding
(E) method overloading

21. Which of the following statements about constructors is NOT true?

(A) All constructors must have the same name as the class they are in.
(B) Constructors' return type must be declared `void`.
(C) A class may have a constructor that takes no parameters.
(D) A constructor is called when a program creates an object with the `new` operator.
(E) A constructor of a subclass can call a constructor of its superclass using the Java reserved word `super`.

```
public class Airplane
{
  private int fuel;

  public Airplane() { fuel = 0; }
  public Airplane(int g) { fuel = g; }

  public void addFuel() { fuel++; }
  public void display() { System.out.print(fuel + " "); }
}

public class Jet extends Airplane
{
  public Jet(int g) { super(2*g); }
}
```

What is the result when the following code is compiled and run?

```
Airplane plane = new Airplane(4);
Airplane jet = new Jet(4);
plane.display();
plane.addFuel();
plane.display();
jet.display();
jet.addFuel();
jet.display();
```

(A) A syntax error, "undefined addFuel," is reported for the `jet.addFuel();` statement.
(B) A run-time error, `ClassCastException`, occurs when `jet.addFuel()` is attempted.
(C) The code compiles and runs with no errors; the output is `4 5 5 6`
(D) The code compiles and runs with no errors; the output is `4 5 8 9`
(E) The code compiles and runs with no errors; the output is `8 9 9 10`

3. Consider the following two classes.

```
public class A
{
  public void show()
  {
    System.out.print("A");
  }
}


public class B extends A
{
  public void show()
  {
    System.out.print("B");
  }
}
```

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

```
A obj = new B();
obj.show();
```

(A) `A`
(B) `B`
(C) `AB`
(D) `BA`

24. Consider the following class.

```
public class SomeMethods
{
  public void one(int first)
  {  /* implementation not shown */  }

  public void one(int first, int second)
  {  /* implementation not shown */  }

  public void one(int first, String second)
  {  /* implementation not shown */  }
}
```

Which of the following methods can be added to the `SomeMethods` class without causing a compile-time error?

I.
```
public void one(int value)
{  /* implementation not shown */  }
```

II.
```
public void one(String first, int second)
{  /* implementation not shown */  }
```

III.
```
public void one(int first, int second, int third)
{  /* implementation not shown */  }
```

(A) I only

(B) I and II only

(C) I and III only

(D) II and III only

(E) I, II, and III

26. Consider the following two methods, which appear within a single class.

```
public static void changeIt(int[] arr, int val, String word)
{
  arr = new int[5];
  val = 0;
  word = word.substring(0, 5);

  for (int k = 0; k < arr.length; k++)
  {
    arr[k] = 0;
  }
}


public static void start()
{
  int[] nums = {1, 2, 3, 4, 5};
  int value = 6;
  String name = "blackboard";

  changeIt(nums, value, name);

  for (int k = 0; k < nums.length; k++)
  {
    System.out.print(nums[k] + " ");
  }

  System.out.print(value + " ");
  System.out.print(name);
}
```

What is printed as a result of the call `start()` ?

(A) `0 0 0 0 0 0 black`

(B) `0 0 0 0 0 6 blackboard`

(C) `1 2 3 4 5 6 black`

(D) `1 2 3 4 5 0 black`

(E) `1 2 3 4 5 6 blackboard`

Questions 2-3 refer to the following information.

Consider the following partial class declaration.

```
public class SomeClass
{
  private int myA;
  private int myB;
  private int myC;

  // Constructor(s) not shown

  public int getA()
  {  return myA;  }

  public void setB(int value)
  {  myB = value;  }
}
```

2. The following declaration appears in another class.

```
SomeClass obj = new SomeClass();
```

Which of the following code segments will compile without error?

(A)
```
int x = obj.getA();
```

(B)
```
int x;
obj.getA(x);
```

(C)
```
int x = obj.myA;
```

(D)
```
int x = SomeClass.getA();
```

(E)
```
int x = getA(obj);
```

---

3. Which of the following changes to `SomeClass` will allow other classes to access but not modify the value of `myC` ?

(A) Make `myC` public.

(B) Include the method:
```
public int getC()
{  return myC;  }
```

(C) Include the method:
```
private int getC()
{  return myC;  }
```

(D) Include the method:
```
public void getC(int x)
{  x = myC;  }
```

(E) Include the method:
```
private void getC(int x)
{  x = myC;  }
```

7. Consider the following class declaration.

```
public class Person
{
  private String myName;
  private int myYearOfBirth;

  public Person(String name, int yearOfBirth)
  {
    myName = name;
    myYearOfBirth = yearOfBirth;
  }

  public String getName()
  {  return myName;  }

  public void setName(String name)
  {  myName = name;  }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

Assume that the following declaration has been made.

```
Person student = new Person("Thomas", 1995);
```

Which of the following statements is the most appropriate for changing the name of `student` from `"Thomas"` to `"Tom"` ?

(A) `student = new Person("Tom", 1995);`

(B) `student.myName = "Tom";`

(C) `student.getName("Tom");`

(D) `student.setName("Tom");`

(E) `Person.setName("Tom");`

8. Consider the following class declaration.

```
public class Student
{
  private String myName;
  private int myAge;

  public Student()
  {  /* implementation not shown */  }

  public Student(String name, int age)
  {  /* implementation not shown */  }

  // No other constructors
}
```

Which of the following declarations will compile without error?

I. `Student a = new Student();`

II. `Student b = new Student("Juan", 15);`

III. `Student c = new Student("Juan", "15");`

(A) I only

(B) II only

(C) I and II only

(D) I and III only

(E) I, II, and III

22. Consider the following `Book` and `AudioBook` classes.

```
public class Book
{
  private int numPages;
  private String bookTitle;

  public Book(int pages, String title)
  {
    numPages = pages;
    bookTitle = title;
  }

  public String toString()
  {
    return bookTitle + " " + numPages;
  }

  public int length()
  {
    return numPages;
  }
}


public class AudioBook extends Book
{
  private int numMinutes;

  public AudioBook(int minutes, int pages, String title)
  {
    super(pages, title);
    numMinutes = minutes;
  }

  public int length()
  {
    return numMinutes;
  }

  public double pagesPerMinute()
  {
    return ((double) super.length()) / numMinutes;
  }
}
```

Consider the following code segment that appears in a class other than `Book` or `AudioBook`.

```
Line 1:  Book[] books = new Book[2];
Line 2:  books[0] = new AudioBook(100, 300, "The Jungle");
Line 3:  books[1] = new Book(400, "Captains Courageous");
Line 4:  System.out.println(books[0].pagesPerMinute());
Line 5:  System.out.println(books[0].toString());
Line 6:  System.out.println(books[0].length());
Line 7:  System.out.println(books[1].toString());
```

Which of the following best explains why the code segment will not compile?

(A) Line 2 will not compile because variables of type `Book` may not refer to variables of type `AudioBook`.

(B) Line 4 will not compile because variables of type `Book` may only call methods in the `Book` class.

(C) Line 5 will not compile because the `AudioBook` class does not have a method named `toString` declared or implemented.

(D) Line 6 will not compile because the statement is ambiguous. The compiler cannot determine which `length` method should be called.

(E) Line 7 will not compile because the element at index 1 in the array named `books` may not have been initialized.

34. Consider the following class declarations.

```
public class Point
{
  private double x;  // x-coordinate
  private double y;  // y-coordinate

  public Point()
  {
    x = 0;
    y = 0;
  }

  public Point(double a, double b)
  {
    x = a;
    y = b;
  }

  // There may be instance variables, constructors, and methods that are not shown.
}


public class Circle
{
  private Point center;
  private double radius;

  /** Constructs a circle where (a, b) is the center and r is the radius.
   */
  public Circle(double a, double b, double r)
  {
    /* missing code */
  }
}
```

Which of the following replacements for `/* missing code */` will correctly implement the `Circle` constructor?

I.
```
center = new Point();
radius = r;
```

II.
```
center = new Point(a, b);
radius = r;
```

III.
```
center = new Point();
center.x = a;
center.y = b;
radius = r;
```

(A) I only

(B) II only

(C) III only

(D) II and III only

(E) I, II, and III

16. Consider the following class declaration.

```
public class SomeClass
{
  private int num;

  public SomeClass(int n)
  {
    num = n;
  }

  public void increment(int more)
  {
    num = num + more;
  }

  public int getNum()
  {
    return num;
  }
}
```

The following code segment appears in another class.

```
SomeClass one = new SomeClass(100);
SomeClass two = new SomeClass(100);
SomeClass three = one;

one.increment(200);

System.out.println(one.getNum() + "   " + two.getNum() + "   " +
                   three.getNum());
```

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

(A) 100 100 100

(B) 300 100 100

(C) 300 100 300

(D) 300 300 100

(E) 300 300 300

---

5. When designing a class hierarchy, which of the following should be true of a superclass?

(A) A superclass should contain the data and functionality that are common to all subclasses that inherit from the superclass.

(B) A superclass should be the largest, most complex class from which all other subclasses are derived.

(C) A superclass should contain the data and functionality that are only required for the most complex class.

(D) A superclass should have public data in order to provide access for the entire class hierarchy.

(E) A superclass should contain the most specific details of the class hierarchy.

20. Consider the following instance variables and method that appear in a class representing student information.

```
private int assignmentsCompleted;
private double testAverage;

public boolean isPassing()
{ /* implementation not shown */ }
```

A student can pass a programming course if at least one of the following conditions is met.

- The student has a test average that is greater than or equal to 90.
- The student has a test average that is greater than or equal to 75 and has at least 4 completed assignments.

Consider the following proposed implementations of the `isPassing` method.

I.
```
if (testAverage >= 90)
  return true;
if (testAverage >= 75 && assignmentsCompleted >= 4)
  return true;
return false;
```

II.
```
boolean pass = false;
if (testAverage >= 90)
  pass = true;
if (testAverage >= 75 && assignmentsCompleted >= 4)
  pass = true;
return pass;
```

III.
```
return (testAverage >= 90) ||
       (testAverage >= 75 && assignmentsCompleted >= 4);
```

Which of the implementations will correctly implement method `isPassing` ?

(A) I only

(B) II only

(C) I and III only

(D) II and III only

(E) I, II, and III

30. 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);`