**Percent of total test grade — 50**

**Directions:** Determine the answer to each of the following questions or incomplete statements, using the available space for any necessary scratch work. Then decide which is the best of the choices given and fill in the corresponding oval on the answer sheet. No credit will be given for anything written in the examination booklet (these pages). Do not spend too much time on any one problem.

Notes:

* Assume that classes listed in the Quick Reference found in Appendix have been imported where appropriate.
* Assume that declarations of variables and methods appear within the context of an enclosing class.
* Assume that method calls that are not prefixed with an object or class name and are not shown within a complete class definition appear within the context of an enclosing class.

1. What is a *mutator* ?  
     
   (A) An instance method that modifies the object’s internal state.

(B) An instance method that provides information about the state of an object, without  
 modifying it.

(C) Code that interacts with another class or objects of that class.

(D) The parent class in an inheritance relationship.

(E) Hiding the implementation details of an object from the clients of the object,

usually using the private keyword.

1. What is *encapsulation* ?  
     
   (A) An instance method that modifies the object’s internal state.

(B) An instance method that provides information about the state of an object, without  
 modifying it.

(C) Code that interacts with another class or objects of that class.

(D) The parent class in an inheritance relationship.

(E) Hiding the implementation details of an object from the clients of the object,

usually using the private keyword.

1. What is an *accessor* ?  
     
   (A) An instance method that modifies the object’s internal state.

(B) An instance method that provides information about the state of an object, without  
 modifying it.

(C) Code that interacts with another class or objects of that class.

(D) The parent class in an inheritance relationship.

(E) Hiding the implementation details of an object from the clients of the object,

usually using the private keyword.

1. What is a *client* (also called *client code*)?  
     
   (A) An instance method that modifies the object’s internal state.

(B) An instance method that provides information about the state of an object, without  
 modifying it.

(C) Code that interacts with another class or objects of that class.

(D) The parent class in an inheritance relationship.

(E) Hiding the implementation details of an object from the clients of the object,

usually using the private keyword.

1. Assume that x and y are variables of type int. The expression  
     
    !(x > y) || !(x < y)  
     
   is equivalent to which of the following?  
     
   (A) true  
   (B) false  
   (C) x == y  
   (D) x != y  
   (E) (x <= y) && (x >= y)
2. Consider the following class declaration:  
     
    public class SomeClass {  
    private int num;  
     
    public SomeClass(int n) {  
    num = n;  
    }  
     
    public void setToZero() {  
    num = 0;  
    }  
     
    public int getNum() {  
    return num;  
    }  
    }  
     
   The following code segment appears in another class.  
     
    SomeClass one = new SomeClass(7);  
    SomeClass two = new SomeClass(8);  
    SomeClass three = two;  
     
    one.setToZero();  
    two.setToZero();  
     
    System.out.println(one.getNum() + " " + two.getNum() +  
    " " + three.getNum());  
     
   What is printed as a result of executing the code segment?  
     
   (A) 7 8 8  
   (B) 0 8 8  
   (C) 0 0 8  
   (D) 0 0 0  
   (E) 0 0 7
3. Which of the following expressions evaluates to true after the code in Question 6 executes?   
     
   I one == two  
   II two == three  
   III one == three  
     
   (A) I only  
   (B) II only  
   (C) III only  
   (D) I and II  
   (E) I, II, and III

**Question 8 refers to the following code.**

public class Date

{

private int myDay;

private int myMonth;

private int myYear;

public Date( ) //default constructor

{ <implementation code> }

public Date( int mo, int day, int yr ) //constructor

{ <implementation code> }

public int month( ) //returns month of date

{ <implementation code> }

public int day( ) //returns day of date

{ <implementation code> }

public int year( ) //returns year of date

{ <implementation code> }

//string representation od Date as "m/d/y", e.g. 4/18/1985

public String toString( )

{ <implementation code> }

}

8. A client class has a display method that writes the date represented as its parameter:

// Outputs date d in the form month:day:year.

public void display (Date d)

{

/\* method body \*/

}

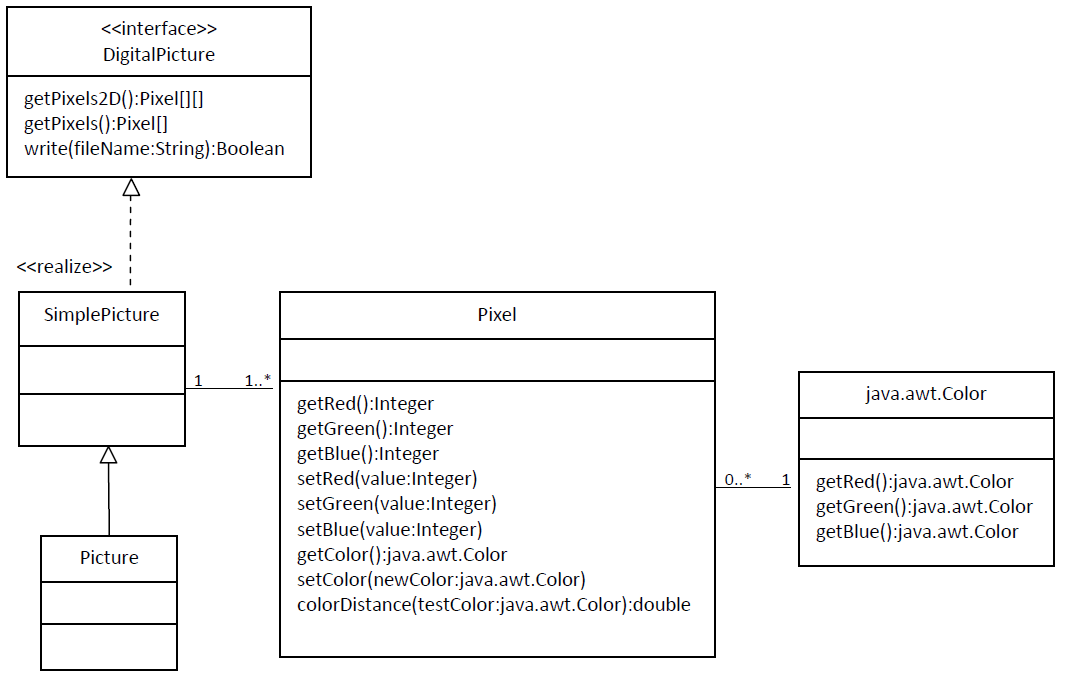
Which of the following are correct replacements for /\* method body \*/?

1. Date d = new Date(m, d, y);

System.out.println(d);

1. System.out.println(d.month() + “:” + d.day() + “:” + d.year());
2. System.out.println(d);
3. I only
4. II only
5. III only
6. II and III only
7. I, II, and III

**Question 9 refers to the following UML class diagram:**



9. Which of the following will compile without error?

1. DigitalPicture p = new Picture();
2. DigitalPicture p = new SimplePicture();
3. Picture p = new SimplePicture();

1. I, II, and III
2. II only
3. III only
4. I and II
5. II and III

**Question 10-12 refers to the following code.**

public class MyClass {

private int[][] data;

public MyClass(int x, int y) {

data = new int[x][y];

for (int i = 0; i < x; i++) {

int z = 1;

for (int j = 0; j < y; j++) {

data[i][j] = z;

z\*=i;

}

}

}

public int foo(int x, int y) {

return data[x][y];

}

}

10. What is printed to the console when the following code segment is executed?  
  
 MyClass a = new MyClass(20, 20);

System.out.println(a.foo(10, 10));

(A) Nothing is printed, since an Exception occurs  
(B) An integer value between 1 and 10  
(C) An integer value between 10 and 100  
(D) An integer value greater than 100  
(E) None of the above

1. What is printed to the console when the following code segment is executed?  
     
    MyClass a = new MyClass(20, 20);

System.out.println(a.foo(0, 10));

(A) Nothing is printed, since an Exception occurs  
(B) An integer value between 1 and 10  
(C) An integer value between 10 and 100  
(D) An integer value greater than 100  
(E) None of the above

1. What is printed to the console when the following code segment is executed?

MyClass a = new MyClass(20, 20);

System.out.println(a.foo(10, 0));

(A) Nothing is printed, since an Exception occurs  
(B) An integer value between 1 and 10  
(C) An integer value between 10 and 100  
(D) An integer value greater than 100  
(E) None of the above

**Questions 13-14 refer to the following declarations.** public class Point {  
 private double myX;  
 private double myY;  
  
 // Creates a Point with coordinates (x,y)  
 public Point(double x, double y) {  
 /\* *implementation not shown* \*/  
 }  
  
 // other methods not shown  
 }  
  
 public class LineSegment {  
 private Point start;  
 private Point end;  
  
 // Creates a LineSegment between the coordinates at the

// ‘start’ Point to the ‘end’ Point  
 public LineSegment(Point start, Point end) {  
 /\* *implementation not shown* \*/  
 }  
  
 // other methods not shown  
 }

1. Which of the following would be the best specification (method header) for a new LineSegment method named distanceToPoint that calculates the shortest distance from the point to the LineSegment ?  
     
   (A) public double distanceToPoint()  
     
   (B) public void distanceToPoint(Point p, double d)  
     
   (C) public double distanceToPoint(Point p)  
     
   (D) public void distanceToPoint()  
     
   (E) public double distanceToPoint(Point p, Point start,   
    Point end)
2. In a Java program which of the following correctly declares and initializes an object LineSegment seg from the coordinates (1.0, 2.0) to (5.0, 6.0) ?  
     
   (A) LineSegment seg = new LineSegment();  
    seg.start = new Point(1.0, 2.0);  
    seg.end = new Point(5.0, 6.0);  
     
   (B) LineSegment seg = new LineSegment();  
    seg.start = new Point();  
    seg.start.myX = 1.0;  
    seg.start.myY = 2.0;  
    seg.end = new Point();  
    seg.end.myX = 5.0;  
    seg.end.myY = 6.0;  
     
   (C) LineSegment seg =  
    new LineSegment(1.0, 2.0, 5.0, 6.0);  
     
   (D) LineSegment seg =  
    new LineSegment((1.0, 2.0), (5.0, 6.0));  
     
   (E) Point p1 = new Point(1.0, 2.0);  
    Point p2 = new Point(5.0, 6.0);  
    LineSegment seg = new LineSegment(p1, p2);

**Questions 15-16 refer to the following declarations.**public class Employee {

private String myName;

private int myEmployeeNumber;

private double mySalary;  
 private double myTax;

public Employee(String name, int num, double salary, double tax)  
 {

/\* implementation not shown \*/

}

public String getName() { return myName; }

public double getSalary() { return mySalary; }

public int getEmployeeNumber() { return myEmployeeNumber; }

public double getTax() { return myTax; }

public void changeSalary(int newSalary) {  
 mySalary = newSalary;  
 }

public double computePay() {

return mySalary – myTax; // take-home pay after taxes.

}

}

public class Consultant extends Employee {

private static final double BONUS = 5000;

public Consultant(String name, int num, double sal, double tax) {

/\* implementation not shown \*/

}

public double computePay() {

/\* implementation code \*/

}

}

1. Which of the following statements is true about the changeSalary method?

I. It is an instance method.

II. It is an accessor method.  
 III. It is a mutator method.   
  
 (A) I only

(B) II only

(C) I and II  
 (D) I and III  
 (E) I, II, and II

1. The computePay method in the Consultant class should override the computePay method to add BONUS to mySalary after subtracting myTax . Which of the following replacements for /\* implementation code \*/ will achieve the desired behavior?  
     
   I return super.computePay() + BONUS;  
     
   II super.computePay();  
    return getSalary() + BONUS;  
     
   III return getSalary() - getTax() + BONUS;  
     
   (A) I only  
   (B) II only  
   (C) III only  
   (D) I and II  
   (E) I and III

**END OF SECTION I.**