1. What are the *3 cornerstones of OOP*?

2. What 2 things do *objects* store?

3. What makes objects unique?

4. Explain *encapsulation*.

5. Java encapsulates data and action modules that access the data in one container, called an \_\_\_\_\_\_\_\_\_\_\_\_.

6. Object members that perform some task are called \_\_\_\_\_\_\_\_\_\_\_\_\_.

7. Object members that store data are called \_\_\_\_\_\_\_\_\_\_\_\_\_.

8. Create the shortest possible class, called *Qwerty* that can compile.

9. Look at programs ***Java0801.java***. Why is the **CardDeck** class not declared **public**?

// Java0801.java

// CardDeck Case Study #01

// This shows a minimal class declaration.

// This class has no practical value, but it compiles and executes.

public class Java0801

{

 public static void main(String args[])

 {

 System.out.println("\nCard Deck Case Study 01\n");

 CardDeck d = new CardDeck();

 System.out.println();

 }

}

class CardDeck

{

}

10. Look at program ***Java0803.java***. This program does compile and execute.

 Even so, what is wrong with it?

// Java0803.java

// CardDeck Case Study #03

// <CardDeck> variables are accessed directly by the <main> method.

// This program violates encapsulation, even though it compiles, and executes.

// This approach greatly compromises program reliability.

public class Java0803

{

 public static void main(String args[])

 {

 System.out.println("\nCard Deck Case Study 03\n");

 CardDeck d = new CardDeck();

 d.cardGame = "Poker";

 d.numDecks = 1;

 d.numPlayers = 5;

 d.cardsLeft = 208;

 System.out.println("Name of Card Game: " + d.cardGame);

 System.out.println("Number of Decks: " + d.numDecks);

 System.out.println("Number of Players: " + d.numPlayers);

 System.out.println("Number of Cards Left: " + d.cardsLeft);

 System.out.println();

 }

}

class CardDeck

{

 String cardGame;

 int numDecks;

 int numPlayers;

 int cardsLeft;

}

11. What is the essence of *encapsulation*?

12. Look at program ***Java0813.java***. Why will this program not compile?

// Java0813.java

// Cube Casestudy #4 - Stage #4 adds a <move> method, which updates the cube's coordinates

// and draws a cube at the new location.

import java.awt.\*;

import java.applet.\*;

public class Java0813 extends Applet

{

 public void paint(Graphics g)

 {

 Cube cube = new Cube(g,50,50,50);

 for (int x = 50; x < 750; x += 50)

 cube.move(g,x,300);

 }

}

class Cube

{

 private int tlX; // topleft X coordinate of the Cube's position

 private int tlY; // topleft y coordinate of the Cube's position

 private int size; // the size of the cube along one edge

 public Cube(Graphics g)

 {

 tlX = 50;

 tlY = 50;

 size = 50;

 }

 public Cube(Graphics g, int x, int y, int s)

 {

 tlX = x;

 tlY = y;

 size = s;

 }

 public void draw(Graphics g)

 {

 int tlX2 = tlX + size/3;

 int tlY2 = tlY + size/3;

 g.setColor(Color.black);

 g.drawRect(tlX,tlY,size,size);

 g.drawRect(tlX2,tlY2,size,size);

 g.drawLine(tlX,tlY,tlX2,tlY2);

 g.drawLine(tlX+size,tlY,tlX2+size,tlY2);

 g.drawLine(tlX,tlY+size,tlX2,tlY2+size);

 g.drawLine(tlX+size,tlY+size,tlX2+size,tlY2+size);

 }

 public void move(Graphics g, int x, int y)

 {

 tlX = x;

 tlY = y;

 draw(g);

 }

}

 13. If a class member is declared **private**, where can it be accessed?

14. Data attributes are usually declared \_\_\_\_\_\_\_.

15. If a class member is declared **public**, where can it be accessed?

16. Methods are usually declared \_\_\_\_\_\_\_.

**Continued on Next Page.**

17. Look at the 4 *get* methods in program ***Java0805.java***.

 Is it possible for any of these methods to alter the class data?

// Java0805.java

// CardDeck Case Study #05 - The <CardDeck> class now has four "get" methods to return the data values of <CardDeck> objects. Note that Java assigns initial values to object data.

public class Java0805

{

 public static void main(String args[])

 {

 System.out.println("\nCard Deck Case Study 05\n");

 CardDeck d = new CardDeck();

 System.out.println("Name of Card Game: " + d.getGame());

 System.out.println("Number of Decks: " + d.getDecks());

 System.out.println("Number of Players: " + d.getPlayers());

 System.out.println("Number of Cards Left: " + d.getCards());

 System.out.println();

 }

}

class CardDeck

{

 private String cardGame;

 private int numDecks;

 private int numPlayers;

 private int cardsLeft;

 public String getGame()

 {

 return cardGame;

 }

 public int getDecks()

 {

 return numDecks;

 }

 public int getPlayers()

 {

 return numPlayers;

 }

 public int getCards()

 {

 return cardsLeft;

 }

}

18. Java assigned default values when a new object is instantiated.

 If an attribute is an **int**, what value is it assigned?

19. Refer to the previous question. If an attribute is a **String**, what value is it assigned?

20. Refer to your answers to the previous 2 questions. Do these 2 mean the same thing?

**Continued on Next Page.**

21. Look at the 4 *set* methods in program ***Java0806.java***.

 Is it possible for any of these methods to alter the class data?

// Java0806.java

// CardDeck Case Study #06 - The <CardDeck> class adds four "set" methods to alter the data attributes of <CardDeck> objects.

public class Java0806

{

 public static void main(String args[])

 {

 System.out.println("\nCard Deck Case Study 06\n");

 CardDeck d = new CardDeck();

 d.setGame("Bridge");

 d.setDecks(1);

 d.setPlayers(4);

 d.setCards(52);

 System.out.println("Name of Card Game: " + d.getGame());

 System.out.println("Number of Decks: " + d.getDecks());

 System.out.println("Number of Players: " + d.getPlayers());

 System.out.println("Number of Cards Left: " + d.getCards());

 System.out.println();

 }

}

class CardDeck

{

 // Data attributes

 private String cardGame;

 private int numDecks;

 private int numPlayers;

 private int cardsLeft;

 // Get return Methods

 public String getGame() { return cardGame; }

 public int getDecks() { return numDecks; }

 public int getPlayers() { return numPlayers; }

 public int getCards() { return cardsLeft; }

 // Set void Methods

 public void setGame(String cG)

 {

 cardGame = cG;

 }

 public void setDecks(int nD)

 {

 numDecks = nD;

 }

 public void setPlayers(int nP)

 {

 numPlayers = nP;

 }

 public void setCards(int cL)

 {

 cardsLeft = cL;

 }

}