25. Consider the program below.

```
public class Waco
{
  public static void main (String args[ ])
  {
    Piggy kathy = new Piggy("Kathy",1500.0);
    Piggy rachel = new Piggy("Rachel",2500.0);
    kathy.showData();
       // Line 1
    System.out.println("Name    " + rachel.name);          // Line 2
    System.out.println("Savings " + rachel.savings);       // Line 3
  }
}


class Piggy
{
  public double savings;
  public String name;

  public Piggy(String n, double s)
  {
    name = n;
    savings = s;
  }

  public void showData()
  {
    System.out.println("Name:    " + name);                // Line 4
    System.out.println("Savings:  " + savings);            // Line 5
  }
}
```

Lines 1 - 5 access data of **Piggy** objects. Which lines have access?

(A) Lines 2 and 3 only

(B) Lines 4 and 5 only

(C) Lines 1, 3, 4 and 5 only

(D) Lines 1, 2, 4 and 5 only

(E) All five lines have access

26. Consider the program below.

```
public class PiggyBusiness
{
  public static void main (String args[ ])
  {
    Piggy kathy = new Piggy("Kathy",1500.0);
    Piggy rachel = new Piggy("Rachel",2500.0);
    kathy.showData();
      // Line 1
    System.out.println("Name    " + rachel.name);      // Line 2
    System.out.println("Savings " + rachel.savings);   // Line 3
  }
}

class Piggy
{
  private double savings;
  public String name;

  public Piggy(String n, double s)
  {
    name = n;
    savings = s;
  }

  public void showData()
  {
    System.out.println("Name:    " + name);           // Line 4
    System.out.println("Savings: " + savings);        // Line 5
  }
}
```

delete

rachel.showData();

add

note

change to private

Change the 2 lines that need to be fixed so that we are using the concept of data encapsulation correctly.

See above

27. Altering the existing values of object instance variables is the job of

(A) accessor methods.
(B) mutator methods.
(C) constructors.
(D) the **main** method.

28. What happens when an object is instantiated without providing initial values for instance variables?

(A) Java provides default values like **0** for int variables and **null** for any object.
(B) The program stops during execution and waits for information.
(C) The program will not compile.
(D) The program crashes during execution.

29. Which features can you use to recognize a constructor in a class declaration?

(A) The constructor name is the same as the class name.
(B) Constructors use both the **public** and the **static** keywords.
(C) Constructors are neither void methods nor return methods.
(D) Both A and C

30. When is a constructor called?

(A) Each time the constructor identifier is used in a program statement
(B) During the instantiation of a new object
(C) During the construction of a new class
(D) At the beginning of any program execution

31. A class can have

(A) one constructor method only.
(B) one or two constructor methods.
(C) multiple constructors with the same identifier (name).
(D) multiple constructors with different identifiers (names).

32. What is an *overloaded* constructor?

(A) A constructor with too many program statements.
(B) A second constructor with the same constructor heading as the first constructor.
(C) A second constructor with a different identifier than the first constructor.
(D) A second or other multiple constructor with a different signature than ar constructor. (different number of parameters or different types of paramters)

33. What is the primary job of the constructor?

(A) Initialize instance variables and get physical space for the object.
(B) Return the value of instance variables.
(C) Alter the initial values of instance variables.
(D) Display the values of instance variables.

34. What happens if the same variable identifier is defined more than once in the same scope?

(A) The program will not compile.
(B) The program will crash during execution.
(C) The program may develop logic errors.
(D) Only the first variable will be processed, while the second variable is ignored.

35. A **class** method

(A) requires using the keyword **new**.
(B) requires using the keyword **private**.
(C) requires using the keyword **static**.
(D) requires all of the above.
(E) requires both A and C.

36. A **private** method

I. can only be accessed by methods of the same class.
II. is usually a helper method.
III. should not be a constructor.

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

37. A **public** method

I. can only access public data.
II. ✓ helps maintain encapsulation, the private data is accesed via these methods only
III. ✓ can be accessed from outside the class.

(A) I only
(B) II only
(C) III only
(D) II & III only

(E) I, II & III

38. A **return** method

(A) indicates the data type of the return value.
(B) uses the keyword **return.**
(C) must be declared **private.**
(D) All of the above.
(E) A & B only

39. Consider the following code segment from a runner class and class declaration.

```
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);

public class CardDeck
{
        public String cardGame;
        public int numDecks;
        public int numPlayers;
        public int cardsLeft;
}
```

How does this class declaration violate the goal of encapsulation?

(A) There is no constructor in the **CardDeck** class.
(B) Read access to the data attributes is possible from outside the class.
(C) Write access to the data attributes is possible from outside the class.
(D) The data attributes are not initialized when the **CardDeck** object is instantiated.
(E) Both choices B and C

40. The scope of a variable

(A) specifies its data value range.
(B) is a list of methods that access the variable.
(C) is the segment of a program where the variable can be accessed.
(D) is defined by the constructor.

41. What happens if the same variable identifier is defined more than once in the same scope?

(A) The program will not compile.
(B) The program will crash during execution.
(C) The program may develop logic errors.
(D) Only the first variable will be processed, while the second variable is ignored.

42. Consider the following code segment and class declaration.

```
Widget widget = new Widget(12);
System.out.println(widget.getWidgets());

public class Widget
{
        private int numWidgets;

        public Widget(int numWidgets)
        {
                numWidgets = numWidgets;
        }

        public int getWidgets()
        {
                return numWidgets;
        }
}
```

What will be the output as a result of executing the code segment?

(A) A compile error message.
(B) A runtime error message.
(C) 0
(D) 12

43. A **class** method

(A) requires using the keyword **new**.
(B) requires using the keyword **private**.
(C) requires using the keyword **static**.
(D) requires all of the above.
(E) requires both A and C.

44. An *object* method

(A) requires using the keyword **new**.
(B) requires using the keyword **private**.
(C) requires using the keyword **static**.
(D) requires all of the above.
(E) requires both A and B.

45. A **private** method

I. can only be accessed by methods of the same class.
II. is usually a helper method.
III. should not be a constructor.

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

46. A *default* constructor is a

(A) *no-parameter (or no-args)* constructor, which is called during the instantiation of a new object.
(B) *parameter* method, which is called during the instantiation of a new object.
(C) *no-parameter* method.
(D) *parameter* method

47. Consider the following code segment and class declaration.

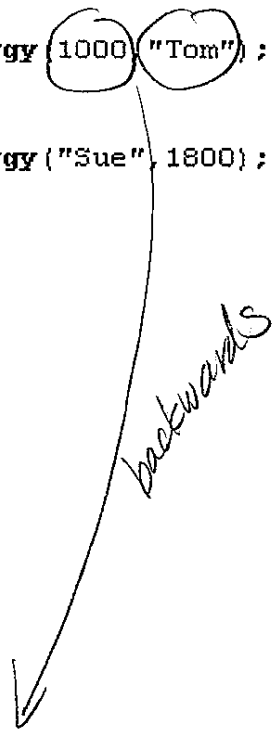
```
Widget widget = new Widget(12);
System.out.println(widget.getWidgets());

public class Widget
{
        private int numWidgets;

        public Widget(int numWidgets)
        {
            this.numWidgets = numWidgets;
        }
        public int getWidgets()
        {
            return numWidgets;
        }
}
```

What will be the output as a result of executing the code segment?

(A) A compile error message.
(B) A runtime error message.
(C) 0
(D) 12

51. What is the output of this program?

```
public class Java0806a
{
    public static void main(String args[])
    {
        Piggy tom = new Piggy(1000,"Tom");
        tom.showData();

        Piggy sue = new Piggy("Sue",1800);
        sue.showData();
    }
}


class Piggy
{
    double savings;
    String name;

    public Piggy()
    {
        savings = 0;
        name = "";
    }

    public Piggy(String n,double s)
    {
        name = n;
        savings = s;
    }

    public void showData()
    {
        System.out.println("Name:    " + name);
        System.out.println("Savings: " + savings);
    }
}
```



(A)
Name:
Savings: 0.0
Name:
Savings: 0.0

(B)
Name: Tom
Savings: 1000.0
Name:
Savings: 0.0

(C)
Name:
Savings: 0.0
Name: Sue
Savings: 1800.0

(D)
Name: Tom
Savings: 1000.0
Name: Sue
Savings: 1800.0

(E) Error