Review for Test 4 on classes and objects

1. Which of the following program statements uses **sqrt** correctly?

   (A) Math.sqrt();
   (B) result = Math.sqrt;
   (C) System.out.println(Math.sqrt(16));
   (D) System.out.println(sqrt(100));

2. What is the output of the following code segment?

```
int a = 25;
int b = 16;
double result = Math.sqrt(Math.abs(b-a));
System.out.println(result);
```

   (A) 5.0
   (B) 4.0
   (C) 3.0
   (D) -3.0

3. Assume that **a** and **b** are non-equal integers.
   What is the value of **result** in the following statement?

```
int a = 25;
int b = 16;
double result = Math.sqrt(Math.min(a,b)-Math.max(b,a));
```

   (A) -5.0
   (B) -4.0
   (C) 5.0
   (D) 4.0

4. Consider the following statement:

   Blank is a computer science tool that involves using program features without knowledge of how the program features are implemented.

   This statement explains

   (A) inheritance.
   (B) encapsulation.
   (C) composition.
   (D) information hiding.
   (E) polymorphism.

5. An object is a

(A) single instance of a given data structure template.
(B) collection of primitive data types.
(C) user-defined data type.
(D) data structure template or blue print.

6. An object method (or instance method) is called by using

(A) the method identifier only.
(B) the object identifier only.
(C) the class identifier, followed by a period and the method identifier.
(D) an object identifier, followed by a period and the method identifier.

7. The Java keyword **new** is used to construct

(A) objects only.
(B) classes only.
(C) objects and classes.
(D) neither objects nor classes.

8. By Java program writing convention, which of the following is true?

(A) Object identifiers start with a capital letter and class identifiers start with a lower-case letter.
(B) Object identifiers start with a lower-case letter and class identifiers start with a capital letter.
(C) Object identifiers start with a capital letter and class identifiers start with a capital letter.
(D) Object identifiers start with a lower-case letter and class identifiers start with a lower-case letter.

For questions 13-20 use the following **Bank** class information. It contains the headings of the methods in the **Bank** class, along with a description.

public Bank()
// default constructor starts checking and savings account with zero dollars.

public Bank(double c, double s)
// constructor creates an object with **c** dollars in checking and **s** dollars in savings.

public double getChecking()
// returns the checking account balance

public double getSavings()
// returns the savings account balance

public double getCombined()
// returns the combined balance of the checking and savings account

public void changeChecking(double amount)
// alters the balance of the checking account by the **amount** parameter

public void changeSavings(double amount)
// alters the balance of the savings account by the **amount** parameter

public void closeChecking()
// alters the checking account balance to zero

public void closeSavings()
// alters the savings account balance to zero

13. Access to methods of the **Bank** class requires

(A) using a statement, like **Bank.getSavings();**
(B) using a statement, like **Bank.getSavings;**
(C) the creation of one or more **Bank** objects.
(D) using the **get** method.

14. A class is _________ .

(A) a data type
(B) a variable
(C) a constant
(D) another name for an object

15. An object is_________.

(A) a data type
(B) a variable
(C) a constant
(D) another name for a class

16. The **new** keyword must be used with

(A) every class method call.
(B) every object method call.
(C) the construction of each object.
(D) the construction of each class.

17. What is the output of the following program segment?

```
Bank tom;
tom = new Bank();
Bank sue;
sue = new Bank();
tom.changeChecking(1000);
sue.changeChecking(1500);
System.out.println("sue:  " + sue.getSavings());
System.out.println("tom:  " + tom.getSavings());
```

(A)
```
tom:  1000.0
sue:  1500.0
```

(B)
```
sue:  1500.0
tom:  1000.0
```

(C)
```
sue:  0.0
tom:  0.0
```

(D) Error message

18. What is the output of the following program segment?

```
Bank tom;
tom = new Bank(7500.0, 5000.0);
Bank sue;
sue = new Bank(4000.0, 3500.0);
System.out.println("tom:  " + tom.getChecking() + "  " + tom.getSavings());
System.out.println("sue:  " + sue.getChecking() + "  " + sue.getSavings());
tom.closeChecking();  tom.closeSavings();
sue.closeChecking();  sue.closeSavings();
```

(A)
```
sue:  7500.0  5000.0
tom:  4000.0  3500.0
```

(B)
```
tom:  7500.0  5000.0
sue:  4000.0  3500.0
```

(C)
```
sue:  0.0
tom:  0.0
```

(D) Error message

19. Consider the two segments below.
Do both segments properly construct a **tom** object?

```
// segment 1
Bank tom;
tom = new Bank(7500, 5000);
```

```
// segment 2
Bank tom = new Bank(7500, 5000);
```

(A) Segment 1 is correct and segment 2 is not correct.
(B) Segment 1 is incorrect and segment 2 is correct.
(C) Both segments are incorrect.
(D) Both segments are correct.

20. Consider the two segments below.
Do both segments properly construct a **tom** object?

```
// segment 1
Bank tom;
tom = new Bank(7,500.0, 5,000.0);
```

```
// segment 2
Bank tom = new Bank(7,500.0, 5,000.
```

(A) Segment 1 is correct and segment 2 is not correct.
(B) Segment 1 is incorrect and segment 2 is correct.
(C) Both segments are incorrect.
(D) Both segments are correct.

21. Consider the following program segment.

```
for (int k = 1; k <= 10; k++)
{
    double var1 = Math.random() * 100;
    int var2 = (int) var1 + 10;
    System.out.print(var2 + "  ");
}
```

Which of the following are possible outputs of executing the program segment?

(A) 46 78 11 18 99 87 42 21 43 78
(B) 91 100 88 60 85 71 22 37 75 49
(C) 10 17 46 56 18 99 21 44 89 75
(D) All of the above.

22. What is the range of possible random integers generated by the program segment in quest

(A) [10 . . . 100]
(B) [10 . . . 109]
(C) [11 . . . 109]
(D) [10 . . . 110]

23. Which of the following program segments generates a random integer in the range [1
9999] ?

(A)

int range = 9999 - 1000 + 1;
int randInt = (int) Math.random() * range +
1000;

(B)

int range = 9999 - 1000 + 1;
int randInt=(int) (Math.random())
*** range + 1000;**

(C)

int randInt = (int) (Math.random() * 9999) +
1000;

(D)

int range = 9999 - 1000 + 1;
int randInt = (int) (Math.random() * r
1000;

24. Which of the following statements is true about the use of parameters with Java methods?

(A) Methods without parameters can compile, but will not execute correctly.
(B) All method declarations require parameters.
(C) Many methods use parameters.
(D) The use of parameters is optional to increase program readability.

25. Which of the following statements is true about a method declaration with multiple param

(A) All parameters must be the same data type.
(B) All parameters must be different data types.
(C) Parameter data types may be the same or they may be different.
(D) The parameter declarations depend on the method call.

19. Which of the following method headings uses proper parameter declarations?

(A) **public static void guess(double rate, double hours, int deductions);**
(B) **public static void guess(double rate, hours, int deductions);**
(C) **public static void guess(rate, hours, deductions);**
(D) **public static void guess(7.85, 42.5, 3);**

20. Which of the following method calls might use parameters correctly?

(A) **guess(double rate, double hours, int deductions);**
(B) **guess(double rate, hours, int deductions);**
(C) **guess(int rate, hours, deductions);**
(D) **guess(7.85, 42.5, 3);**

21. What distinguishes a call to a **return** method?

(A) The method call is the only part of a complete program statement.
(B) The method call provides a value, which is used in the program statement.
(C) The method call includes the **void** keyword.
(D) The method call includes the **return** keyword.

22. Consider the following code segment 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);

class CardDeck
{
        String cardGame;
        int numDecks;
        int numPlayers;
        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

23. What is the consequence of declaring data attributes **private**?

I. Access to the data attributes is read/write to members of the same class.
II. Access to the data attributes is read-only to members outside the class.
III. Access to the data attributes is completely restricted to members outside the class.

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

24. What is the essence of *encapsulation*?

(A) Declare data attributes in a class as **private**.
(B) Declare data attributes in a class as **public**.
(C) Contain the data and the methods that access that data inside the same class.
(D) Contain all the data in one class and all the methods in another class.
(E) None of the above

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
   }
}
```

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

____

____

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

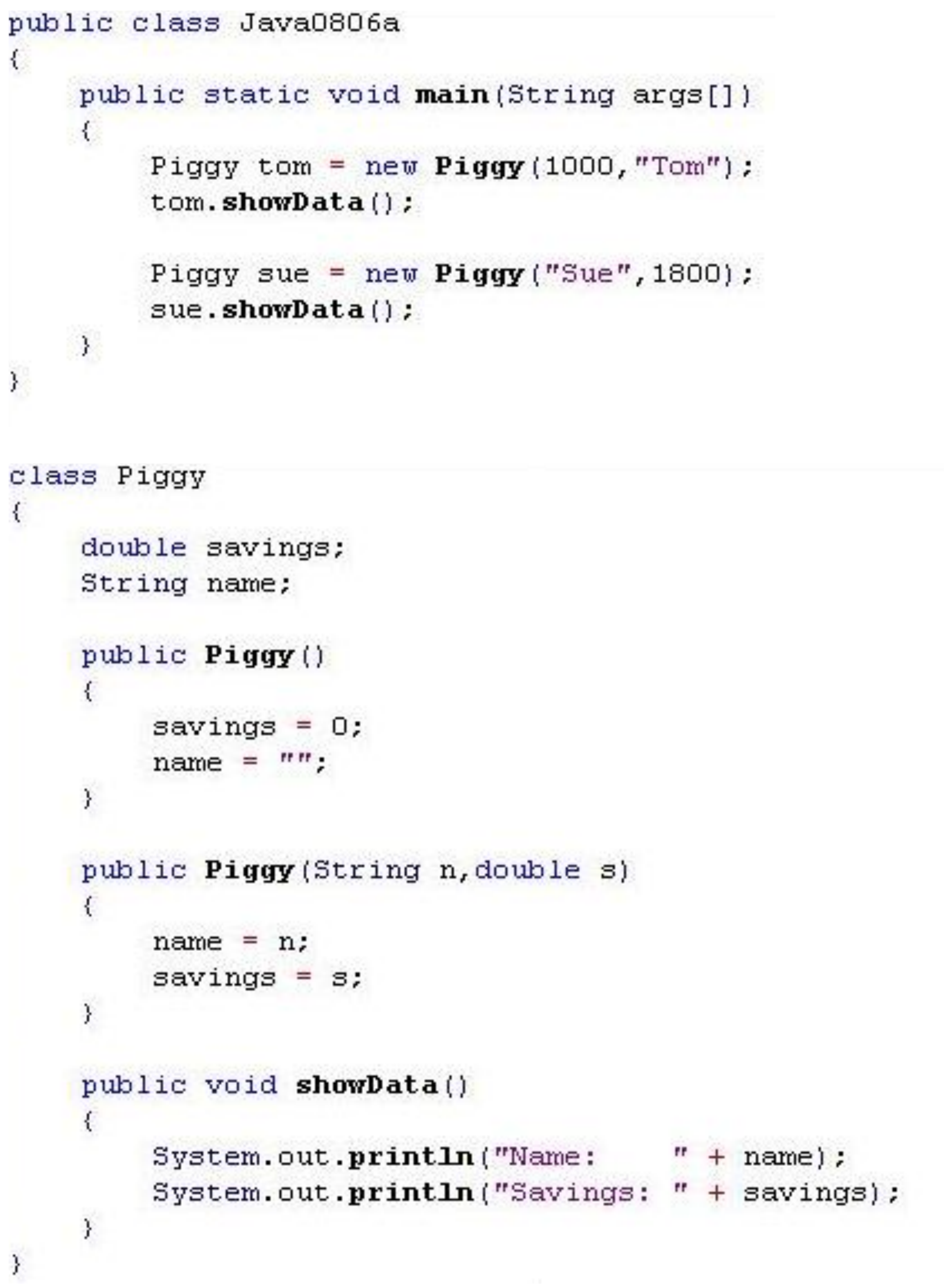
47. Consider the following code segment and class declaration.

```
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