Eyad Ghanem 222102280

Dr. shaker el-Sappagh

Principles of programming student portfolio:

Contains:

- Assignments
- Solved Labs

1. Assignments

Assigment 1:

```
package eyad;

//part #2
      // a)  mystery has the value 5
      // b)   mystery has the value 0
      // C)   mystery has the value 1

      //A) 0.3
      //B) 3.333333
      //C)-1.57
      //D) 4.3
      //E) 0.2480
      //F) 1
      //G) 1
// part 3
   //AUTHOR: Eyad
   //FILENAME: C:\Users\mada_\eclipse-
workspace\eyad\src
   //SPECIFICATION: TempConverter
   //Description: tip calculator
   //TIME SPENT: how long it took you to complete the
assignment

import java.util.Scanner;
public class Assignment1 {

   static double bill,tip2;
   static int tip;
   static double total;
```

```
    public static void main(String[] args) {

        Scanner key= new Scanner(System.in);

        System.out.println("Enter the bill amount: ");
        bill = key.nextDouble();
        System.out.println("What percent would you like 
to tip: ");
        tip = key.nextInt();
        tip2= bill/tip;
        System.out.println(" Tip Amount: $"+ tip2);

        total = bill+tip2;
        System.out.println("Total amount with tip: $" + 
total);
    }
}
```

Assigment 4:

Multiple Choice and True/False

1-b. decision structure.
2- d. boolean expression.
3- a. NOT
4- b. logical operators.
5- NOT > AND > OR
6- d. case-sensitive string comparison.
7- a. flag
8- b. 'A' is less than 'B'
9- a. nested if statement
10- a
11- a. &&
12- c. the equals method.
13- a. ternary operator
14- b. default
15- a. Tab
16- True
17- True
18- False
19- True
20- False
21- True

Find the Error
1- x = 1
should be: x == 1
2- else
should be: else if ()
3- multiline if statement should have braces {}
code should be:
```
if(num2 == 0) {
// statement
// statement
// statement
} else {
// statement
// statement
// statement
}
```

4- relational operators like (>) can't be used in switch statement
5- x >= 1
should be: x > 1 (not including 1 or 20)
6- ||
should be: &&
7- &&
should be: ||

Assigment 5:

3.1 Multiple Choice Questions

1) The ________ statement is used to make simple decisions in Java.
A) `do/while`
B) `for`
C) `branch`
D) `if`
Answer: D

2) A Boolean expression is one that is either:
A) `true` or `false`

B) `x` or `y`
C) Positive or negative
D) None of the above
Answer: A

3) This type of operator determines whether a specific relationship exists between two values:
A) Logical
B) Mathematical
C) Unary
D) Relational
Answer: D

4) Which of the following expressions will determine whether `x` is less than or equal to `y`?
A) `x > y`
B) `x =< y`
C) `x <= y`
D) `x >= y`
Answer: C

5) Which one of the following is the not equal operator?
A) <>
B) `NOT`
C) `*&`
D) `!=`
Answer: D

6) What will be the value of `x` after the following code is executed?

```
int x = 75;
int y = 60;
if (x > y)
     x = x - y;
```

A) 75
B) 15
C) 60
D) 135
Answer: B

7) What will be the value of `ans` after the following code has been executed?

```
int ans = 10;
int x = 65;
int y = 55;
if (x >= y)
     ans = x + y;
```

A) 10
B) 120
C) 100
D) No value, there is a syntax error.
Answer: B

8) What will be the value of `ans` after the following code has been executed?

```
int x = 90, y = 55, ans = 10;
if ( x == y);
     ans *= 2;
```

A) 10
B) 145
C) 20
D) No value, there is a syntax error.

Answer: C

9) A block of code is enclosed in a set of:
A) braces `{ }`
B) parentheses `( )`
C) double quotes `" "`
D) brackets `[ ]`
Answer: A

10) A flag may have the values:
A) 0 or 1
B) +1 or -1
C) `true` or `false`
D) of any character
Answer: C

11) If `chr` is a character variable, which of the following `if` statements is written correctly?
A) `if (chr = "a")`
B) `if (chr == "a")`
C) `if (chr = 'a')`
D) `if (chr == 'a')`
Answer: D

12) In Java, when a character is stored in memory, it is actually stored as a(n):
A) Unicode number
B) ASCII character code
C) EBCDIC character code
D) Morse code
Answer: A

13) This is an international coding system that is extensive enough to represent all the characters of all the world's alphabets:
A) ASCII

B) Unicode
C) Java
D) None of the above
Answer: B

14) What will be the values of `ans`, `x`, and `y` after the following statements are executed?

```
int ans = 35, x = 50, y =50;
if ( x >= y)
{
     ans = x + 10;
     x -=y;
}
else
{
     ans = y + 10;
     y += x;
}
```

A) `ans` = 60, `x` = 50, `y` =100
B) `ans` = 60, `x` =0, `y` =50
C) `ans` = 45, `x` = 50, `y` = 0
D) `ans` = 45, `x` = 50, `y` = 50
Answer: B

15) What will be the value of `bonus` after the following code is executed?

```
int bonus, sales = 10000;
if (sales < 5000)
     bonus = 200;
else if (sales < 7500)
     bonus = 500;
else if (sales < 10000)
     bonus = 750;
else if (sales < 20000)
     bonus = 1000;
else
     bonus = 1250;
```

A) 200
B) 500
C) 750
D) 1000
E) 1250
Answer: D

16) In most editors, you are indenting by one level each time that you press this key:
A) Tab
B) Shift
C) Alt
D) Space
Answer: A

17) If you prematurely terminate an `if` statement with a semicolon, the compiler will:
A) Not display an error message
B) Assume you are placing a null statement there
C) All of the above
D) None of the above
Answer: C

18) What would be the value of `bonus` after the following statements are executed?

```
int bonus, sales = 1250;
if (sales > 1000)
     bonus = 100;
if (sales > 750)
     bonus = 50;
if (sales > 500)
     bonus = 25;
else
     bonus = 0;
```

A) 100
B) 500
C) 25
D) 0
Answer: C

19) What would be the value of `bonus` after the following statements are executed?

```
int bonus, sales = 85000;
char dept = 'S';
if (sales > 100000)
     if (dept == 'R')
     bonus = 2000;
     else
     bonus = 1500;
else if (sales > 75000)
     if (dept == 'R')
     bonus = 1250;
     else
     bonus = 1000;
else
     bonus = 0;
```

A) 2000
B) 1500
C) 1250

D) 1000
Answer: D

20) Which of the following is the correct `boolean` expression to test for: `int x` being a value between, but not including, 500 and 650, or `int y` not equal to 1000?
A) `((x >= 500 && x <= 650) && (y != 1000))`
B) `((x > 500 AND x < 650) OR !(y.equal(1000)))`
C) `((x > 500 && x < 650) || (y != 1000))`
D) `((x < 500 && x > 650) || !(y == 1000))`
Answer: C

21) ________ works like this: If the expression on the left side of the `&&` operator is `false`, the expression the right side will not be checked.
A) Short-circuit evaluation
B) Reverse logic
C) Boolean logic
D) Relational evaluation
Answer: A

22) If `str1` and `str2` are both `String`s, which of the following will correctly test to determine whether `str1` is less than `str2`?

(1) `(str1 < str2)`
(2) `(str1.equals(str2) < 0)`
(3) `(str1.compareTo(str2) < 0)`
A) 1, 2, and 3 will all work
B) 2
C) 3
D) 2 and 3
Answer: C

23) To do a case insensitive compare which of the following could be used to test the equality of two strings, `str1` and `str2`?
A) `(str1.equalsIgnoreCase(str2))`

B) `(str1.compareToIgnoreCase(str2) == 0)`
C) Only A
D) A and B
Answer: D

24) What will be the value of `pay` after the following statements are executed?

```
int hours = 45;
double pay, payRate = 10.00;
pay = hours <= 40 ? hours * payRate :
     40 * payRate + (hours - 40) * payRate * 1.5;
```
A) 400.00
B) 450.00
C) 465.00
D) 475.00
Answer: D

25) What would be the value of `x` after the following statements were executed?

```
int x = 10;
switch (x)
{
     case 10:
     x += 15;
     case 12:
     x -= 5;
     break;
     default:
     x *= 3;
}
```
A) 5
B) 20
C) 25
D) 30
Answer: B

26) What will be printed when the following code is executed?

```
double x = 45678.259;
System.out.printf("%,.2f", x);
```

A) 45678.259

B) 0,045,678.26

C) 45,678.26

D) 45,678.3

Answer: C27) Which of the following will format 12.78 to display as 12.8%?

A) `System.out.printf("%2.1d%", 12.78);`

B) `System.out.printf("%.2f%%", 12.78);`

C) `System.out.printf("%1.2d%", 12.78);`

D) `System.out.printf("%.1f%%", 12.78);`

Answer: D

28) The expression tested by an `if` statement must evaluate to:

A) 0 or 1

B) +1 or -1

C) `true` or `false`

D) `t` or `f`

Answer: C

29) These operators use two operands:

A) Unary

B) Binary

C) Tertiary

D) None of the above

Answer: B

30) What is the value of `x` after the following code has been executed?

```
int x = 75;
int y = 90;
if ( x != y)
     x += y;
```
A) 75
B) 90
C) 15
D) 165
Answer: D

31) What is the value of `ans` after the following code has been executed?

```
int x = 40;
int y = 40;
int ans = 0;
if (x = y)
     ans = x + 10;
```
A) 50
B) 80
C) 30
D) No value, this is a syntax error.
Answer: D

32) What is the value of `ans` after the following code has been executed?

```
int x = 35;
int y = 20, ans = 80;
if (x < y);
     ans += y;
```
A) 80
B) 100
C) 35
D) 55
Answer: B

33) Enclosing a group of statements inside a set of braces creates a:
A) block of statements

B) `boolean` expression
C) loop
D) nothing, it is just for readability
Answer: A

34) This is a `boolean` variable that signals when some condition exists in the program:
A) Sentinel
B) Block
C) Flag
D) Case
Answer: C

35) Which of the following correctly tests the `char` variable `chr` to determine whether it is NOT equal to the character B?
A) `if (chr > 'B')`
B) `if (chr < 'B')`
C) `if (chr != 'B')`
D) `if (chr != "B")`
Answer: C

36) In an `if/else` statement, if the `boolean` expression is `false`:
A) the first statement or block is executed
B) the statement or block following the `else` is executed
C) all statements or blocks are executed
D) no statements or blocks are executed
Answer: B

37) What will be the values of `ans`, `x`, and `y` after the following statements are executed?

```
int ans = 0, x = 15, y =25;
if ( x >= y)
{
     ans = x + 10;
     x -=y;
}
else
{
     ans = y + 10;
     y += x;
}
```
A) `ans` = 0, `x` = 15, `y` = 25
B) `ans` = 25, `x` = -10, `y` = 25
C) `ans` = 35, `x` = 15, `y` = 40
D) `ans` = 25, `x` = 15, `y` = 40
Answer: C

38) What would be the value of `discountRate` after the following statements are executed?

```
double discountRate = 0.0;
int purchase = 100;
if (purchase > 1000)
     discountRate = .05;
else if (purchase > 750)
     discountRate = .03;
else if (purchase > 500)
     discountRate = .01;
```
A) .05
B) .03
C) .01
D) 0.0
Answer: D

39) What would be the value of `discountRate` after the following statements are executed?

```
double discountRate = 0.0;
int purchase = 1250;
if (purchase > 1000)
     discountRate = .05;
if (purchase > 750)
```

```
     discountRate = .03;
if (purchase > 500)
     discountRate = .01;
else
     discountRate = 0;
```
A) .05
B) .03
C) .01
D) 0
Answer: C

40) What would be the value of `discountRate` after the following statements are executed?

```
double discountRate = 0.0;
int purchase = 1250;
char cust = 'N';
if (purchase > 1000)
     if (cust == 'Y')
     discountRate = .05;
     else
     discountRate = .04;
else if (purchase > 750)
     if (cust == 'Y')
     discountRate = .04;
     else
     discountRate = .03;
else
     discountRate = 0;
```
A) .05
B) .04
C) .03
D) 0
Answer: B

41) Which of the following is the correct `boolean` expression to test for: `int x` being a value less than or equal to 500 or greater than 650, and `int y` not equal to 1000?

A) `((x >= 500 && x <650) && (y != 1000))`
B) `((x <= 500 OR x > 650) AND !(y.equal(1000)))`
C) `((x >= 500 || x < 650) || (y != 1000))`
D) `((x <= 500 || x > 650) && !(y == 1000))`
Answer: D

42) If `str1` and `str2` are both `String`s, which of the following expressions will correctly determine whether they are equal?

(1) `(str1 == str2)`
(2) `str1.equals(str2)`
(3) `(str1.compareTo(str2) == 0)`
A) 1, 2, and 3 will all work
B) 1 and 2
C) 1 and 3
D) 2 and 3
Answer: D

43) What will be printed when the following code is executed?

```
int y = 10;
if ( y == 10)
{
     int x = 30;
     x += y;
}
System.out.print("x = ");
System.out.print(x);
```
A) `x = 30`
B) `x = 40`
C) `x = 20`
D) `x` is unknown when the last statement is executed.
Answer: D

44) The `switch` statement is a:
A) Multiple alternative decision structure

B) Nested decision structure
C) Sequence structure
D) Test expression
Answer: A

45) What will be the value of `charges` after the following code is executed?

```
double charges, rate = 7.00;
int time = 180;
charges = time <= 119 ? rate * 2 :
     time / 60.0 * rate;
```

A) 7.00
B) 14.00
C) 21.00
D) 28.00
Answer: C

46) What would be the value of `discountRate` after the following statements are executed?

```
double discountRate;
char custType = 'B';
switch (custType)
{
     case 'A':
     discountRate = .08;
     break;
     case 'B':
     discountRate = .06;
     case 'C':
     discountRate = .04;
     default:
     discountRate = 0.0;
}
```

A) .08
B) .06

C) .04
D) 0.0
Answer: D

47) What will be printed when the following code is executed?

```
double x = 45678.259;
String output = String.format("%,.1f", x);
System.out.println(output);
```

A) 45678.259
B) 45,678.259
C) 45,678.26
D) 45,678.3
Answer: D

48) Which of the following will format 12.7801 to display as $12.78?
A) `System.out.printf("$%,.2f", 12.7801);`
B) `System.out.printf("%f", 12.7801);`
C) `System.out.printf("%.2f$$", 12.7801);`
D) `System.out.printf("$d", 12.7801);`
Answer: A

49) What does the following code display?

```
int d = 9, e = 12;
System.out.printf("%d %d\n", d, e);
```

A) `%d %d`
B) `9 12`
C) `%d 9`
D) `%9 %12`
Answer: B

50) What does the following code display?

```
double x = 12.3798146;
System.out.printf("%.2f\n", x);
```

A) `123798146`
B) `1238`
C) `%12.38`
D) `12.38`
Answer: D

3.2 True/False Questions

1) Programs never need more than one path of execution.
Answer: F

2) An important style rule you should adopt for writing `if` statements is to write the conditionally executed statement on the line after the `if` statement.
Answer: T

3) The `if/else` statement will execute one group of statements if its `boolean` expression is `true` or another group if its `boolean` expression is `false.`
Answer: T

4) Because the `&&` operator performs short-circuit evaluation, your `boolean` expression will usually execute faster if the subexpression that is most likely `false` is on the left of the `&&` operator.
Answer: T

5) A local variable's scope always ends at the closing brace of the block of code in which it is declared.
Answer:

6) When testing for character values, the `switch` statement does not test for the case of the character.

Answer: F

7) An important style rule you should follow when writing `if` statements is to line up the conditionally executed statement with the `if` statement.
Answer: F

8) Unicode is an international encoding system that is extensive enough to represent ALL the characters of ALL the world's alphabets.
Answer:

9) Because the `||` operator performs short-circuit evaluation, your `boolean` expression will generally be evaluated faster if the subexpression that is most likely to be `true` is on the left.
Answer: T

10) When two `String`s are compared using the `compareTo` method, the cases of the two strings are not considered.
Answer: F

11) In a `switch` statement, each of the `case` values must be unique.
Answer: T

12) In a `switch` statement, if two different values for the *CaseExpression* would result in the same code being executed, you must have two copies of the code, one after each *CaseExpression*.
Answer: F

13) The `String.format` method works exactly like the `System.out.printf` method, except that it does not display the formatted string on the screen.
Answer: T

14) The `System.out.printf` method formats a string and displays it in the console window.

Answer: T

Assigment 6:

MULTIPLE CHOICE

1) A. ++
2) c. x = 33, y = 9
3) b. 28
4) b. Loop
5) d. infinite loop
6) a. An iteration
7) a. Counter variable
8) b. post-test loop
9) a. Braces
10) b. 100
11) d. This is an infinite loop
12) b. Input validation
13) c. User controlled loop
14) d. The boolean condition can never be true
15) c. This is an infinite loop.
16) b. 1
17) c. counter-controlled loop
18) c. 12
19) a. 40
20) d. Sentinel
21) b. Set the accumulator where the total will be kept to an initial value, usually zero
22) c. for loop
23) d. diskOut.println("Calvin");
24) b. import java.io.*;

25) a. FileWriter fwriter = new FileWriter("MyFile.txt", true);
PrintWriter outFile = new PrintWriter(fwriter);

26) c. while (inputFile.hasNext())
{ ... }

27) d. x = 17, y = 4

28) a. 3

29) c. a way to terminate

30) c. while, for

31) d. This is an infinite loop

32) d. 210

33) d. Numbers in the range 100 – 500

34) b. 200

35) d. 5

36) b. conditional loop

37) d. all of the above

38) c. 5, 8, 11, 14,

39) a. Running total

40) b. is a special value that cannot be mistaken as a member of the list

41) b. do-while loop

42) d. Delimiter

43) c. File file = new File("MyFile.txt");
Scanner inputFile = new Scanner(file);

44) c. int number = inputFile.readInt();

45) b. The File class's exists method

True/False

1) true
2) true

3) false
4) false
5) false
6) true
7) true
8) false
9) true
10) true
11) true
12) true

Assigment 7 (Array)**Section 1:**

\

Q1: Which of the following statements about arrays are *true*?

A. An array is a group of variables containing values that all have the same type.

B. Elements are located by index or subscript.

C. The length of an array `c` is determined by the expression `c.length();`.

D. The zeroth element of array `c` is specified by `c[0]`.

a. A, C, D.
[✓ b. A, B, D.]
c. C, D.

d. A, B, C, D.

Q2: Consider the array:

s[0] = 7

s[1] = 0
s[2] = -12
s[3] = 9
s[4] = 10
s[5] = 3
s[6] = 6

The value of s[s[6] - s[5]] is:
a. 0.

b. 3.
c. 9.
d. 0.

Section 2:

Q1: A programmer must do the following before using an array:

a. declare then reference the array.

b. create then declare the array.

c. create then reference the array.

d. declare then create the arra

Q2: Consider the code segment below. Which of the following statements is *false*?

```
int[] g;

g = new int[23];
```

a. The first statement declares an array reference.

b. The second statement creates the array.

c. g is a reference to an array of integers.

d. The value of `g[3]` is `-1`.

Section 3:

Q1: Which of the following statements about creating arrays and initializing their elements is *false*?

a. The **`new`** keyword should be used to create an array.

b. When an array is created, the number of elements must be placed in square brackets following the type of element being stored.

c. The elements of an array of integers have a value of **`null`** before they are initialized.

d. A for loop is commonly used to set the values of the elements of an array.

Q2: What do the following statements do?

```
double[] array;

array = new double[14];
```

a. Create a **`double`** array containing 13 elements.

b. Create a **`double`** array containing 14 elements.

c. Create a **`double`** array containing 15 elements.

d. Declare but do not create a double array.

Q3: Which of the following initializer lists would correctly set the elements of array `n`?

a. int[] n = { 1, 2, 3, 4, 5 };

b. array n[int] = { 1, 2, 3, 4, 5 };

c. int n[5] = { 1; 2; 3; 4; 5 };

d. int n = new int(1, 2, 3, 4, 5);

7.4 Q4: Constant variables also are called ________.
a. write-only variables.

b. finals.

c. named constants.

d. All of the above.

Q5: Which of the following will *not* produce a compiler error?

a. Changing the value of a constant after it is declared.

b. Changing the value at a given index of an array after it is created.

c. Using a final variable before it is initialized.

d. All of the above will produce compiler errors.

Q6: Consider the program below:

```
public class Test
```

```
{

   public static void main( String[] args ) {

      int[] a;

      a = new int[ 10 ];


      for ( int i = 0; i < a.length; i++ )
         a[ i ] = i + 2;

      int result = 0;

      for ( int i = 0; i < a.length; i++ )
         result += a[ i ];

      System.out.printf( "Result is: %d\n", result ); }  //
   end main

}  // end class Test
```

The output of this program will be:

a. Result is: 62.

b. Result is: 64.
c. Result is: 65.
d. Result is: 67.

Q7: Consider the class below:

```
public class Test

{

   public static void main( String[] args ) {

      int[] a = {  99, 22, 11, 3, 11, 55, 44, 88, 2, -3 };


      int result = 0;


      for ( int i = 0; i < a.length; i++ )
      {

         if ( a[ i ] > 30 )
            result += a[ i ];

      }  // end for
```

```
    System.out.printf( "Result is: %d\n", result ); } //
  end main

} // end class Test
```

The output of this Java program will be:

a. Result is: 280.

b. Result is: 154.
c. Result is: 286.
d. Result is: 332.

Q8: Which flag in a format specifier indicates that values with fewer digits than the field width should begin with a leading 0?

a. p.
b. l.
c. w.
d. 0.

Q9: Invalid possibilities for array indices include ________.
a. Positive integers.

b. Negative integers.

c. Zero.

d. None of the above.

Q10: Which expression adds 1 to the element of array arrayName at index i?

a. ++arrayName[i].

b. arrayName++[i].

c. arrayName[i++].

d. None of the above.

Q11: Attempting to access an array element out of the bounds of an array, causes a(n) ____________.

a. ArrayOutOfBoundsException.

b. ArrayElementOutOfBoundsException.

c. ArrayIndexOutOfBoundsException.

d. ArrayException.

Q12: Which of the following statements is *false*?

a. An exception indicates a problem that occurs while a program executes.

b. Exception handling enables you to create fault-tolerant programs that can resolve (or handle) exceptions—in many cases, this allows a program to continue executing as if no problems were encountered.

c. The **`catch`** block contains the code that might throw an exception, and the **`try`** block contains the code that handles the exception if one occurs.

d. Inside the **`catch`** block, you can use the parameter's identifier to interact with a caught exception object.

Section 4

Q1: Consider integer array values, which contains 5 elements. Which statements successfully swap the contents of the array at index 3 and index 4?

a.

```
values[ 3 ] = values[ 4 ];
values[ 4 ] = values[ 3 ];
```

b.

```
values[ 4 ] = values[ 3 ];
values[ 3 ] = values[ 4 ];
```

c.

```
int temp = values[ 3 ];
values[ 3 ] = values[ 4 ];
values[ 4 ] = temp;
```

d.

```
int temp = values[ 3 ];
values[ 3 ] = values[ 4 ];
values[ 4 ] = values[ 3 ];
```

Q2: In this question, assume a class, Book, has been defined. Which set of statements creates an array of Book objects?

a.

```
Book[] books;
```

```
books = new Book[ numberElements ];
```

b.

```
Book[] books]

books  new Book()[ numberElements ];
```

c.

```
new Book() books[];

books = new Book[ numberElements ];
```

d. All of the above.

Section 5

Q1: Assume array items contains the integer values 0, 2, 4, 6 and 8. Which of the following set of statements uses the enhanced for loop to display each value in array items?

a.

```
for ( int i = 0; i < items.length; i++ )
   System.out.prinf( "%d\n", items[ i ] );
```

b.

```
for ( int i : items )

   System.out.printf( "%d\n", items[ i ] );
```

c.

```
for ( int i : items )
   System.out.printf( "%d\n", i );
```

d.

```
for ( int i = 0 : items.length )
   System.out.printf( "%d\n", items[ i ] );
```

Q2: Which of the following tasks *cannot* be performed using an enhanced for loop?

a. Calculating the product of all the values in an array.

b. Displaying all even element values in an array.

c. Comparing the elements in an array to a specific value.

d. Incrementing the value stored in each element of the array.

Section 6

Q1: Which statement correctly passes the array items to method takeArray? Array items contains 10 elements.

a. takeArray(items[]).
b. takeArray(items).

c. takeArray(items[9]).

d. Arrays cannot be passed to methods—each item must be sent to the method separately.

Q2: Consider array items, which contains the values 0, 2, 4, 6 and 8. If method changeArray is called with the method call changeArray(item items[2]), what values are stored in items after the method has finished executing?

```
public static void changeArray( int[] passedArray, int value ) {

   passedArray[ value ] = 12;
   value = 5;

}  // end method changeArray
```

a. 0, 2, 5, 6, 12.

b. 0, 2, 12, 6, 8.
c. 0, 2, 4, 6, 5.

d. 0, 2, 4, 6, 12.

Q3: When an argument is passed by reference:

a. a copy of the argument's value is passed to the called method.

b. changes to the argument do not affect the original variable's value in the caller.

c. the called method can access the argument's value in the caller directly and modify that data.

d. the original value is removed from memory.

Section 7

Q1: What kind of application tests a class by creating an object of that class and calling the class's methods?

a. Pseudo application.

b. Debugger.

c. Tester.

d. Test harness.

Section 8

Q1: In Java, multidimensional arrays:

a. are not directly supported.

b. are implemented as arrays of arrays.

c. are often used to represent tables of values.

d. All of the above.

Q2: In array items, which expression below accesses the value at row 3 and column 4?

a. items[3].[4]

b. items[3[4]].
c. items[3][4].
d. items[3, 4].

Q3: An array with *m* rows and *n* columns is *not*:

A. An *m*-by-*n* array.

B. An *n*-by-*m* array.

C. A two-dimensional array.
D. A dual-transcripted array.

a. A and C.
b. A and D.
c. B and D.
d. B and C.

Q4: Which statement below initializes array items to contain 3 rows and 2 columns?

a. int[][] items = { { 2, 4 }, { 6, 8 }, { 10, 12 } };.
b. int[][] items = { { 2, 6, 10 }, { 4, 8, 12 } };.

c. int[][] items = { 2, 4 }, { 6, 8 }, { 10, 12 };.
d. int[][] items = { 2, 6, 10 }, { 4, 8, 12 };.

Q5: For the array in the previous question, what is the value returned by items[1][0]?

a. 4.

b. 8.
c. 12.
d. 6.

Q6: Which of the following statements creates a multidimensional array with 3 rows, where the first row contains 1 element, the second row contains 4 elements and the final row contains 2 elements?

a. int[][] items = { { 1, null, null, null }, { 2, 3, 4, 5 }, { 6, 7, null, null } };.
b. int[][] items = { { 1 }, { 2, 3, 4, 5 }, { 6, 7 } };.

c. int[][] items = { { 1 }, { 2, 3, 4, 5 }, { 6, 7 }, { });.

d. int[][] items = { { 1 }, { 4 }, { 2 } };.

Q7: Which of the following sets of statements creates a multidimensional array with 3 rows, where the first row contains 1 value, the second row

contains 4 items and the final row contains 2 items? a.

```
int[][] items;

items = new int[ 3 ][ ? ];
items[ 0 ] = new int[ 1 ];
items[ 1 ] = new int[ 4 ];
items[ 2 ] = new int[ 2 ];
```

b.

```
int[][] items;

items = new int[ 3 ][ ];
items[ 0 ] = new int[ 1 ];
items[ 1 ] = new int[ 4 ];
items[ 2 ] = new int[ 2 ];
```

c.

```
int[][] items;

items = new int[ ? ][ ? ];
items[ 0 ] = new int[ 1 ];
items[ 1 ] = new int[ 4 ];
items[ 2 ] = new int[ 2 ];
```

d.

```
int[][] items;

items[ 0 ] = new int[ 1 ];
items[ 1 ] = new int[ 4 ];
items[ 2 ] = new int[ 2 ];
```

Q8: The preferred way to traverse a two-dimensional array is to use ___________.

a. a do while statement.

b. a for statement.

c. two nested for statements.

d. three nested for statements.

Q9: Which set of statements totals the items in each row of two-dimensional array items, and displays each total?

a.

```
int total = 0;


for ( int row = 0; row < items.length; row++ ) {

   total = 0;


   for ( int column = 0; column < a[ row ].length; column++ )
      total += a[ row ][ column ];


   System.out.printf( "%d\n", total ); }
```

b.

```
int total = 0;


for ( int row = 0; row < items.length; row++ ) {

   for ( int column = 0; column < a[ row ].length; column++ )
      total += a[ row ][ column ];


   System.out.printf( "%d\n", total ); }
```

c.

```
int total = 0;


for ( int row = 0; row < items.length; row++ ) {

   for ( int column = 0; column < a[ column ].length; column++ ) total
      += a[ row ][ column ];


   System.out.printf( "%d\n", total ); }
```

d.

```
int total = 0;

for ( int row = 0; row < items.length; row++ ) {
   total = 0;

   for ( int column = 0; column < a[ column ].length; column++ ) total
      += a[ row ][ column ];

   System.out.printf( "%d\n", total ); }
```

Section 8

Q1: Which set of statements totals the values in two-dimensional int array items?

a.

```
int total = 0;

for ( int subItems : items ) for
   ( int item : subItems )
      total += item;
```

b.

```
int total = 0;

for ( int item: int[] subItems : items )
   total += item;
```

c.

```
int total = 0;

for ( int[] subItems : items )
   for ( int item : items )
      total += item;
```

d.

```
int total = 0;
```

```
for ( int[] subItems : items )
   for ( int item : subItems )

      total += item;
```

Section 9

Q1: An argument type followed by a(n) ________ in a method’s parameter list indicates that the method receives a variable number of arguments of that particular type.

a. square brackets ([]).

b. ellipsis (…).

c. varargs keyword.

d. All of the above are acceptable to indicate a variable number of arguments.

Section 10

Q1: Which command below runs TestProgram, and passes in the values files.txt and 3?

a. java TestProgram files.txt 3.

b. java TestProgram files.txt, 3.

c. java TestProgram "files.txt", "3".

d. java TestProgram (the arguments files.txt and 3 were passed in when the application was compiled).

Q2: Which method call converts the value in variable stringVariable to an integer?

a. Convert.toInt(stringVariable).

b. Convert.parseInt(stringVariable).

c. Integer.parseInt(stringVariable).

d. Integer.toInt(stringVariable).

Section 11

Q1: Class Arrays methods sort, binarySearch, equals and fill are overloaded for primitive-type arrays and Object arrays. In addition, methods __________ and __________ are overloaded with generic versions.

a. sort, binarySearch.

b. sort, fill.

c. binarySearch, equals.

d. binarySearch, fill.

7.13 Q2: Class Arrays provides method __________ for comparing arrays.

a. compare.

b. compares. c. equal.

d. equals.

Assigment 8 (Objects and classes):

6.1 Multiple Choice Questions

1) One or more objects may be created from a(n):

A) field

B) class

C) method

D) instance

2) Class objects normally have ________ that perform useful operations on their data, but primitive variables do not.

A) fields

B) instances

[✓ C) methods]

D) relationships

3) In the cookie cutter metaphor, think of the ________ as a cookie cutter and ________ as the cookies.

A) object; classes

[✓ B) class; objects]

C) class; fields

D) attribute; methods

4) Which of the following are classes from the Java API?

A) Scanner

B) Random

C) PrintWriter

[✓ D) All of the above]

5) When you are working with a ________, you are using a storage location that holds a piece of data.

[✓ A) primitive variable]

B) reference variable

C) numeric literal

D) binary number

6) What is stored by a reference variable?

A) A binary encoded decimal

[✓ B) A memory address]

C) An object

D) A string

7) Most programming languages that are in use today are:

A) procedural

B) logic

C) object-oriented

D) functional

8) Java allows you to create objects of this class in the same way you would create primitive variables.

A) Random

B) String

C) PrintWriter

D) Scanner

9) A UML diagram does not contain:

A) the class name

B) the method names

C) the field names

D) object names

10) Data hiding, which means that critical data stored inside the object is protected from code outside the object, is accomplished in Java by:

A) using the public access specifier on the class methods

B) using the private access specifier on the class methods

C) using the private access specifier on the class definition

D) using the private access specifier on the class fields

11) For the following code, which statement is NOT true?

```
public class Sphere
```

```
{
private double radius;
public double x;
private double y;
private double z;
}
```

A) x is available to code that is written outside the Circle class.

B) radius is not available to code written outside the Circle class.

C) radius, x, y, and z are called members of the Circle class.

D) z is available to code that is written outside the Circle class.

12) You should not define a class field that is dependent upon the values of other class fields:

A) in order to avoid having stale data

B) because it is redundant

C) because it should be defined in another class

D) in order to keep it current

13) What does the following UML diagram entry mean?

```
+ setHeight(h : double) : void
```

A) this is a public attribute named Height and is a double data type

B) this is a private method with no parameters and returns a double data type

C) this is a private attribute named Height and is a double data type

D) this is a public method with a parameter of data type double and does not return a value

14) Methods that operate on an object's fields are called:

A) instance variables

B) instance methods

C) public methods

D) private methods

15) The scope of a private instance field is:

A) the instance methods of the same class

B) inside the class, but not inside any method

C) inside the parentheses of a method header

D) the method in which they are defined

16) A constructor:

A) always accepts two arguments

B) has return type of void

C) has the same name as the class

D) always has an access specifier of private

17) Which of the following statements will create a reference, str, to the String, "Hello, World"?

A) String str = "Hello, World";

B) string str = "Hello, World";

C) String str = new "Hello, World";

D) str = "Hello, World";

18) Two or more methods in a class may have the same name as long as:

A) they have different return types

B) they have different parameter lists

C) they have different return types, but the same parameter list

D) you cannot have two methods with the same name

19) Given the following code, what will be the value of finalAmount when it is displayed?

```
public class Order
{
private int orderNum;
private double orderAmount;
private double orderDiscount;
public Order(int orderNumber, double orderAmt,
double orderDisc)
{
orderNum = orderNumber;
orderAmount = orderAmt;
orderDiscount = orderDisc;
}
public int getOrderAmount()
{
return orderAmount;
}
public int getOrderDisc()
{
return orderDisc;
}
}
public class CustomerOrder
{
public static void main(String[] args)
{
int ordNum = 1234;
double ordAmount = 580.00;
double discountPer = .1;
Order order;
double finalAmount = order.getOrderAmount() —
order.getOrderAmount() * order.getOrderDisc();
System.out.printf("Final order amount = $%,.2f\n",
finalAmount);
}
}
```

A) 528.00

B) 580.00

C) There is no value because the constructor has an error.

D) There is no value because the object order has not been created.

20) A class specifies the _________ and _________ that a particular type of object has.

A) relationships; methods

B) fields; object names

C) fields; methods

D) relationships; object names

21) This refers to the combining of data and code into a single object.

A) Data hiding

B) Abstraction

C) Object

D) Encapsulation

22) Another term for an object of a class is:

A) access specifier

B) instance

C) member

D) method

23) In your textbook the general layout of a UML diagram is a box that is divided into three sections. The top section has the _________; the middle section holds _________; the bottom section holds _________.

A) class name; attributes or fields; methods

B) class name; object name; methods

C) object name; attributes or fields; methods

D) object name; methods; attributes or fields

24) For the following code, which statement is NOT true?

```
public class Circle
{
private double radius;
public double x;
private double y;
}
```

A) x is available to code that is written outside the Circle class.

B) radius is not available to code written outside the Circle class.

C) radius, x, and y are called members of the Circle class.

D) y is available to code that is written outside the Circle class.

25) It is common practice in object-oriented programming to make all of a class's:

A) methods private

B) fields private

C) fields public

D) fields and methods public

26) After the header, the body of the method appears inside a set of:

A) brackets, []

B) parentheses, ()

C) braces, {}

D) double quotes, ""

27) In UML diagrams, this symbol indicates that a member is private:

A) *

B) #

C) -

D) +

28) In UML diagrams, this symbol indicates that a member is public.

A) /

B) @

C) -

D) +

29) In a UML diagram to indicate the data type of a variable enter:

A) the variable name followed by the data type

B) the variable name followed by a colon and the data type

C) the class name followed by the variable name followed by the data type

D) the data type followed by the variable name

30) When an object is created, the attributes associated with the object are called:

A) instance fields

B) instance methods

C) fixed attributes

D) class instances

31) When an object is passed as an argument to a method, what is passed into the method's parameter variable?

A) the class name

B) the object's memory address

C) the values for each field

D) the method names

32) A constructor is a method that:

A) returns an object of the class.

B) never receives any arguments.

C) with the name ClassName.constructor.

D) performs initialization or setup operations.

33) The scope of a public instance field is:

A) only the class in which it is defined

B) inside the class, but not inside any method

C) inside the parentheses of a method header

D) the instance methods and methods outside the class

34) Which of the following statements will create a reference, str, to the string, "Hello, world"?

(1) String str = new String("Hello, world");

(2) String str = "Hello, world";

A) 1

B) 2

C) 1 and 2

D) neither 1 or 2

35) Overloading means multiple methods in the same class:

A) have the same name, but different return types

B) have different names, but the same parameter list

C) have the same name, but different parameter lists

D) perform the same function

36) Given the following code, what will be the value of finalAmount when it is displayed?

```
public class Order
```

```
{
private int orderNum;
private double orderAmount;
private double orderDiscount;
public Order(int orderNumber, double orderAmt,
double orderDisc)
{
orderNum = orderNumber;
orderAmount = orderAmt;
orderDiscount = orderDisc;
}
public double finalOrderTotal()
{
return orderAmount - orderAmount *
orderDiscount;
}
}
public class CustomerOrder
{
public static void main(String[] args)
{
Order order;
int orderNumber = 1234;
double orderAmt = 580.00;
double orderDisc = .1;
order = new Order(orderNumber, orderAmt, orderDisc);
double finalAmount = order.finalOrderTotal();
System.out.printf("Final order amount = $%,.2f\n",
finalAmount);
}
```

}

A) 528.00

B) 580.00

C) 522.00

D) There is no value because the object order has not been created.

37) A class's responsibilities include:

A) the things a class is responsible for doing

B) the things a class is responsible for knowing

C) both A and B

D) neither A nor B

38) Instance methods do not have this key word in their headers:

A) public

B) static

C) private

D) protected

39) Which of the following is NOT involved in finding the classes when developing an object-oriented application?

A) Describe the problem domain.

B) Identify all the nouns.

C) Write the code.

D) Refine the list of nouns to include only those that are relevant to the problem.

40) This is a group of related classes.

A) archive

B) package

C) collection

D) attachment

41) Quite often you have to use this statement to make a group of classes available to a program.

A) import

B) use

C) link

D) assume

42) Look at the following statement.

import java.util.Scanner;

This is an example of

A) a wildcard import

B) an explicit import

C) unconditional import

D) conditional import

10

43) Look at the following statement.

import java.util.*;

This is an example of:

A) a wildcard import

B) an explicit import

C) unconditional import

D) conditional import

44) The following package is automatically imported into all Java programs.

A) java.java

B) java.default

C) java.util

D) java.lang

6.2 True/False Questions

1) An object can store data.

T

2) A class in not an object, but a description of an object.

T

3) An access specifier indicates how the class may be accessed.

T

4) A method that stores a value in a class's field or in some other way changes the value of a field is known as a mutator method.

T

5) Instance methods should be declared static.

F

6) A constructor is a method that is automatically called when an object is created.

T

7) Shadowing is the term used to describe where the field name is hidden by the name of a local or parameter variable.

T

8) The public access specifier for a field indicates that the attribute may not be accessed by statements outside the class.

F

9) A method that gets a value from a class's field but does not change it is known as a mutator method.

F

10) Instance methods do not have the key word static in their headers.

T

11) The term "default constructor" is applied to the first constructor written by the author of a class.

F

12) When a local variable in an instance method has the same name as an instance field, the instance field hides the local variable.

F

13) The term "no-arg constructor" is applied to any constructor that does not accept arguments.

T

14) The java.lang package is automatically imported into all Java programs.

T

Assigment 9:

Multiple Choice Questions

1) When an "is a" relationship exists between objects, it means that the specialized object has:
A) some of the characteristics of the general class, but not all, plus additional characteristics
B) some of the characteristics of the general object, but not all
C) none of the characteristics of the general object
D) all the characteristics of the general object, plus additional characteristics

2) Which of the following statements declares `Salaried` as a subclass of `PayType`?
A) public class Salaried extends PayType
B) public class Salaried implements PayType
C) public class Salaried derivedFrom(Paytype)
D) `public class PayType derives Salaried`

3) If `ClassA` extends `ClassB`, then:
A) public and private members of `ClassB` are public and private, respectively, in `ClassA`
B) public members in `ClassB` are public in `ClassA`, but private members in `ClassB` cannot be directly accessed in `ClassA`
C) neither public or private members in `ClassB` can be directly accessed in `ClassA`
D) private members in `ClassB` are changed to protected members in `ClassA`

4) In an inheritance relationship:
A) The superclass constructor always executes before the subclass constructor
B) The subclass constructor always executes before the superclass constructor
C) The constructor with the lowest overhead always executes first regardless of inheritance
D) The unified constructor always executes first regardless of inheritance

5) In UML diagrams, inheritance is shown:
A) With a line that has an open arrowhead at one end that points to the superclass
B) With a line that has an open arrowhead at one end that points to the subclass
C) With a line that has a closed arrowhead at one end that points to the superclass
D) With a line that has a closed arrowhead at one end that points to the subclass

6) What key word can you use to call a superclass constructor explicitly?
A) goto
B) this
C) super
D) extends

7) What is wrong with the following code?
```
public class ClassB extends ClassA
{
	public ClassB()
	{
		int init = 10;
		super(40);
	}
}
```
A) Nothing is wrong with the code.
B) The method `super` is not defined.
C) The call to the method `super` must be the first statement in the constructor.
D) No values may be passed to super.

8) Look at the following code and determine what the call to `super` will do.
```
public class ClassB extends ClassA
{
	public ClassB()
	{
		super(10);
	}
}
```
A) This cannot be determined form the code shown.
B) It will call the constructor of `ClassA` that receives an integer as an argument.

C) It will call the method named `super` and pass the value 10 to it as an argument.
D) The method `super` will have to be defined before we can say what will happen.

9) If a subclass constructor does not explicitly call a superclass constructor:
A) it must include the code necessary to initialize the superclass fields
B) the superclass fields will be set to the default values for their data types
C) Java will automatically call the superclass's default or no-arg constructor immediately after the code in the subclass's constructor executes
D) Java will automatically call the superclass's default or no-arg constructor just before the code in the subclass's constructor executes

10) The `super` statement that calls the superclass constructor:
A) must be the first statement in the superclass's constructor
B) can appear in any method of the subclass
C) must be the first statement in the subclass's constructor
D) is deprecated and is no longer supported in newer versions of Java

11) Replacing inadequate superclass methods with more suitable subclass methods is known as what?
A) Method upgrading
B) Tactical inheritance
C) Method overriding
D) Method overloading

12) If two methods have the same name but different signatures, they are:
A) overridden
B) overloaded
C) superclass methods
D) subclass methods

13) Look at the following code. The method in line ________ will override the method in line ________.
```
Line 1 public class ClassA
Line 2 {
Line 3 public ClassA() {}
Line 4 public int method1(int a){}
Line 5 public double method2(int b){}
Line 6 }
Line 7 public ClassB extends ClassA
Line 8 {
Line 9 public ClassB(){}
Line 10 public int method1(int b, int c){}
Line 11 public double method2(double c){}
Line 12 }
```
A) 10, 4
B) 11, 5
C) Both A and B
D) None of the above

14) Look at the following code. Which line will cause a compiler error?
```
Line 1 public class ClassA
Line 2 {
```

```
Line 3 public ClassA() {}
Line 4 public final int method1(int a){}
Line 5 public double method2(int b){}
Line 6 }
Line 7 public ClassB extends ClassA
Line 8 {
Line 9 public ClassB(){}
Line 10 public int method1(int b){}
Line 11 public double method2(double c){}
Line 12 }
```

A) 4
B) 5
C) 10
D) 11

15) When a subclass overloads a superclass method:
A) Both methods may be called with a subclass object
B) Only the subclass method may be called with a subclass object
C) Only the superclass method may be called with a subclass object
D) Neither method may be called with a subclass object

16) A protected member of a class may be directly accessed by:
A) methods of the same class
B) methods of a subclass
C) methods in the same package
D) All of the above

17) When declaring class data members, it is best to declare them as:
A) private members
B) public members
C) protected members
D) restricted members

18) If you do not provide an access specifier for a class member, the class member is given ________ by default.
A) private
B) public
C) protected
D) package

19) When a method is declared with the ________ modifier, it cannot be overridden in a subclass.
A) extends
B) final
C) super
D) public

20) If `ClassC` extends `ClassB`, which extends `ClassA`, this would be an example of:
A) multiple inheritance
B) a chain of inheritance

C) a family tree
D) packaging

21) Look at the following code.
```
Line 1 public class ClassA
Line 2 {
Line 3 public ClassA() {}
Line 4 public void method1(){}
Line 5 }
Line 6 public class ClassB extends ClassA
Line 7 {
Line 8 public ClassB(){}
Line 9 public void method1(){}
Line 10 }
Line 11 public class ClassC extends ClassB
Line 12 {
Line 13 public ClassC(){}
Line 14 public void method1(){}
Line 15 }
```
Which `method1` will be executed as a result of the following statements?
```
ClassA item1 = new ClassC();
item1.method1();
```
A) Line 4
B) Line 9
[✓ C) Line 14]
D) This is an error and will cause the program to crash.

22) Look at the following code.
```
Line 1 public class ClassA
Line 2 {
Line 3 public ClassA() {}
Line 4 public void method1(int a){}
Line 5 }
Line 6 public class ClassB extends ClassA
Line 7 {
Line 8 public ClassB(){}
Line 9 public void method1(){}
Line 10 }
Line 11 public class ClassC extends ClassB
Line 12 {
Line 13 public ClassC(){}
Line 14 public void method1(){}
Line 15 }
```
Which method will be executed as a result of the following statements?
```
ClassB item1 = new ClassA();
item1.method1();
```
A) Line 4
B) Line 9
C) Line 14
[✓ D) This is an error and will cause the program to crash.]

23) If a class contains an abstract method:
A) you cannot create an instance of the class

B) the method will have only a header, but not a body, and end with a semicolon
C) the method must be overridden in subclasses
D) All of the above

24) Given the following code which of the following is TRUE?

```
public class ClassB implements ClassA{}
```

A) ClassA must override each method in ClassB.
B) ClassB must override each method in ClassA.
C) `ClassB` inherits from `ClassA`.
D) `ClassA` inherits from `ClassB`.

25) All fields declared in an interface:
A) are final and static
B) have protected access
C) must be initialized in the class implementing the interface
D) have private access

26) Look at the following code. Which line has an error?

```
Line 1 public interface Interface1
Line 2 {
Line 3 int FIELDA = 55;
Line 4 public int methodA(double){}
Line 5 }
```

A) 1
B) 2
C) 3
D) 4

27) Look at the following code. Which line in `ClassA` has an error?

```
Line 1 public interface MyInterface
Line 2 {
Line 3 int FIELDA = 55;
Line 4 public int methodA(double);
Line 5 }
Line 6 public class ClassA implements MyInterface
Line 7 {
Line 8 FIELDA = 60;
Line 9 public int methodA(double) { }
Line 10 }
```

A) 6
B) 7
C) 8
D) 9

28) Look at the following code. What is missing from ClassA?

```
Line 1 public interface MyInterface
Line 2 {
Line 3 int FIELDA = 55;
Line 4 public int methodA(double);
Line 5 }
Line 6 public class ClassA implements MyInterface
Line 7 {
Line 8 FIELDA = 60;
Line 9 public int methodB(double) { }
Line 10 }
```

A) It does not override `methodA.`
B) It does not have a constructor.
C) It does not overload `methodA.`
D) Nothing is missing. It is a complete class.

29) When one object is a specialized version of another object, there is this type of relationship between them.
A) "has a"
B) "is a"
C) direct
D) "contains a"

30) A subclass can directly access:
A) all members of the superclass
B) only public and private members of the superclass
C) only protected and private members of the superclass
D) only public and protected members of the superclass

31) In the following statement, which is the superclass?
`public class ClassA extends ClassB implements ClassC`
A) ClassA
B) ClassB
C) ClassC
D) Cannot tell

32) A subclass may call an overridden superclass method by:
A) prefixing its name with the `super` key word and a dot `(.)`
B) prefixing its name with the name of the superclass in parentheses
C) using the `extends` keyword before the method is called
D) calling the superclass method first and then calling the subclass method

33) In the following statement, which is the subclass?
`public class ClassA extends ClassB implements ClassC`
A) ClassA
B) ClassB
C) ClassC
D) Cannot tell

34) In the following statement, which is the interface?
`public class ClassA extends ClassB implements ClassC`
A) ClassA
B) ClassB
C) ClassC
D) Cannot tell

35) What is wrong with the following code?

```
public class ClassB extends ClassA
{
public ClassB()
{
super(40);
System.out.println("This is the last statement " +
"in the constructor.");
}
}
```

[✓ A) Nothing is wrong with the code.]

B) The method `super` is not defined.

C) The call to the method `super` must be the first statement in the constructor.

D) No values may be passed to super.

36) In the following code, what will the call to super do?

```
public class ClassB extends ClassA
{
public ClassB()
{
super(40);
System.out.println("This is the last statement "+
"in the constructor.");
}
}
```

A) This cannot be determined from the code.

B) It will call the method `super` and pass the value 40 to it as an argument.

[✓ C) It will call the constructor of `ClassA` that receives an integer as an argument.]

D) The method `super` will have to be defined before we can say what will happen.

37) If a superclass does not have a default constructor or a no-arg constructor:

A) then a class that inherits from it, must initialize the superclass values.

[✓ B) then a class that inherits from it, must call one of the constructors that the superclass does have.]

C) then a class that inherits from it, does not inherit the data member fields from the superclass.

D) then a class that inherits from it, must contain the default constructor for the superclass.

38) Look at the following code. The method in line ________ will override the method in line ________.

```
Line 1 public class ClassA
Line 2 {
Line 3 public ClassA() {}
Line 4 public int method1(int a){}
Line 5 public int method2(int b){}
Line 6 }
Line 7 public ClassB extends ClassA
Line 8 {
Line 9 public ClassB(){}
Line 10 public int method1(int b){}
Line 11 public int method2(double c){}
Line 12 }
```

A) 4, 10

B) 5, 11

[✓ C) 10, 4]

D) 11, 5
E) Both A and B

39) Look at the following code. Which line will cause a compiler error?

```
Line 1 public class ClassA
Line 2 {
Line 3 public ClassA() {}
Line 4 public int method1(int a){}
Line 5 public final int method2(double b){}
Line 6 }
Line 7 public ClassB extends ClassA
Line 8 {
Line 9 public ClassB(){}
Line 10 public int method1(int b){}
Line 11 public int method2(double c){}
Line 12 }
```

A) 4
B) 5
C) 10
D) 11

40) Protected members are:
A) not quite private
B) not quite public
C) both A and B
D) neither A nor B

41) Protected class members are denoted in a UML diagram with the symbol
A) *
B) #
C) +
D) -

42) Which of the following is TRUE about protected access?
A) Protected members may be accessed by methods in the same package or in a subclass, even when the subclass is in a different package.
B) Protected members may be accessed by methods in the same package or in a subclass, but only if the subclass is in the same package.
C) Protected members cannot be accessed by methods in any other classes.
D) Protected members are actually named constants.

43) Like a family tree, a ________ shows the inheritance relationship between classes.
A) flowchart
B) class map
C) class hierarchy
D) binary tree

44) In a class hierarchy:
A) the more general classes are toward the bottom of the tree and the more specialized are toward the top

B) the more general classes are toward the top of the tree and the more specialized are toward the bottom
C) the more general classes are toward the left of the tree and the more specialized are toward the right
D) the more general classes are toward the right of the tree and the more specialized are toward the left

45) Look at the following code:

```
Line 1 public class ClassA
Line 2 {
Line 3 public ClassA() {}
Line 4 public void method1(int a){}
Line 5 }
Line 6 public class ClassB extends ClassA
Line 7 {
Line 8 public ClassB(){}
Line 9 public void method1(){}
Line 10 }
Line 11 public class ClassC extends ClassB
Line 12 {
Line 13 public ClassC(){}
Line 14 public void method1(){}
Line 15 }
```

Which `method1` will be executed when the following statements are executed?

```
ClassA item1 = new ClassB();
item1.method1();
```

A) Line 4
B) Line 9
C) Line 14
D) This is an error and will cause the program to crash.

46) Look at the following code:

```
Line 1 public class ClassA
Line 2 {
Line 3 public ClassA() {}
Line 4 public void method1(int a){}
Line 5 }
Line 6 public class ClassB extends ClassA
Line 7 {
Line 8 public ClassB(){}
Line 9 public void method1(){}
Line 10 }
Line 11 public class ClassC extends ClassB
Line 12 {
Line 13 public ClassC(){}
Line 14 public void method1(){}
Line 15 }
```

Which method will be executed when the following statements are executed?

```
ClassC item1 = new ClassA();
item1.method1();
```

A) Line 4
B) Line 9
C) Line 14
D) This is an error and will cause the program to crash.

47) If a class contains an abstract method:

A) you must create an instance of the class
B) the method will have only a header, but not a body, and end with a semicolon
C) the method cannot be overridden in subclasses
D) All of the above.

48) In an interface all methods have:
A) private access
B) protected access
C) public access
D) packaged access

49) Which of the following statements correctly specifies three interfaces?
A) public class ClassA implements Interface1, Interface2, Interface3
B) public class ClassA implements [Interface1, Interface2, Interface3]
C) public class ClassA implements (Interface1, Interface2, Interface3)
D) public class ClassA implements Interface1 Interface2 Interface3

50) What is wrong with the following code?
```
IntCalculator square = new IntCalculator() {
      public int calculate(int number)
      {
            return number * number;
      }
}
```
A) The outer braces are not needed.
B) The inner braces are not needed.
C) The `new` key word is not needed.
D) The statement does not end with a semicolon.

51) This is a variable whose value is never changed, but it isn't declared with the `final` keyword.
A) virtually constant variable
B) effectively final variable
C) default variable
D) anonymous inner variable

52) Which of the following is an example of a lambda expression?
A) `int x = x * factor;`
B) `IntCalculator = new divider(x, 2);`
C) `IntCalculator multiplier = x -> x * factor;`
D) All of the above

True/False Questions

1) Inheritance involves a subclass, which is the general class, and a superclass, which is the specialized class.
F

2) Private members of the superclass cannot be accessed by the subclass.
T

3) It is not possible for a superclass to call a subclass's method.
T

4) When a subclass extends a superclass, the public members of the superclass become public members of the subclass.
T

5) If a method in a subclass has the same signature as a method in the superclass, the subclass method overloads the superclass method.
F

6) Every class is either directly or indirectly derived from the `Object` class.
T

7) An abstract class is not instantiated, but serves as a superclass for other classes.
T

8) In an inheritance relationship, the subclass constructor always executes before the superclass constructor.
F

9) If two methods in the same class have the same name but different signatures, the second overrides the first.
F

10) Every class has a `toString` method and an `equals` method inherited from the `Object` class.
T

11) All methods in an abstract class must also be declared `abstract`.
F

12) When an interface variable references an object, you can use the interface variable to call any and all of the methods in the class implementing the interface.
F

13) A compiler error will result if an anonymous inner class tries to use a variable that is not `final`, or not effectively `final`.
T

14) A functional interface is simply an interface that has one abstract method.

T

2.Labs

Lab 1:

```
package project1;

public class lab1 {

//*****************************************************
******
   // Name: Eyad Mohamed
   // Title: Lab1.java
   // Description: Prints out My first program and my
name.
   // Time spent: 20 minutes
   // Date: 7/11/2022
//*****************************************************
******
   public static void main (String[] args)
   {
   System.out.println("This is my first program!");
   }
}
```

Lab 3

```
public class PizzaOrder {
   public static void main(String[] args) {
      // Create a Scanner object to read input.
      Scanner keyboard = new Scanner(System.in);

      String firstName; // User's first name
      boolean discount = false; // Flag for discount
      int inches; // Size of the pizza
      char crustType; // For type of crust
      String crust = "Hand-tossed"; // Name of crust
```

```
        double cost = 12.99; // Cost of the pizza
        final double TAX_RATE = .08; // Sales tax rate
        double tax; // Amount of tax
        char choice; // User's choice
        String input; // User input
        String toppings = "Cheese "; // List of toppings
        int numberOfToppings = 0; // Number of toppings

        // Prompt user and get first name.
        System.out.println("Welcome to Mike and " +
                "Diane's Pizza");

        System.out.print("Enter your first name: ");
        firstName = keyboard.nextLine();
        // Determine if user is eligible for discount by
        // having the same first name as one of the owners.
        // ADD LINES HERE FOR TASK #1
        if (firstName.equalsIgnoreCase("Mike") ||
firstName.equalsIgnoreCase("Diane")) {
            discount = true;
        }

        // Prompt user and get pizza size choice.
        System.out.println("Pizza Size (inches) Cost");
        System.out.println(" 10 $10.99");
        System.out.println(" 12 $12.99");
        System.out.println(" 14 $14.99");
        System.out.println(" 16 $16.99");
        System.out.println("What size pizza " +
                "would you like?");
        System.out.print("10, 12, 14, or 16 " +
                "(enter the number only): ");
        inches = keyboard.nextInt();
        // Set price and size of pizza ordered.
        // ADD LINES HERE FOR TASK #2
        while (inches != 10 && inches != 12 && inches != 14 && inches !=
16) {
           System.out.println("Please choose one of the given sizes");
           System.out.print("Choose again: ");
           inches = keyboard.nextInt();
        }
        if (inches == 10) {
           cost = 10.99;
        } else if (inches == 12) {
           cost = 12.99;
        } else if (inches == 14) {
           cost = 14.99;
        } else if (inches == 16) {
           cost = 16.99;
        }

        // Consume the remaining newline character.
        keyboard.nextLine();

        // Prompt user and get crust choice.
        System.out.println("What type of crust " +
              "do you want? ");
        System.out.print("(H)Hand-tossed, " +
              "(T) Thin-crust, or " +
              "(D) Deep-dish " +
              "(enter H, T, or D): ");
        input = keyboard.nextLine();
        crustType = input.charAt(0);
        // Set user's crust choice on pizza ordered.
        // ADD LINES FOR TASK #3
        while (crustType != 'H' && crustType != 'h'
        && crustType != 'T' && crustType != 't'
        && crustType != 'D' && crustType != 'd') {
      System.out.println("Please choose one of the given crust types");
      System.out.print("Choose again: ");
      input = keyboard.nextLine();
      crustType = input.charAt(0);
    }
    if (crustType == 'H' || crustType == 'h') {
      crust = "Hand-tossed";
    } else if (crustType == 'T' || crustType == 't') {
      crust = "Thin-crust";
    } else if (crustType == 'D' || crustType == 'd') {
      crust = "Deep-dish";
    }

    // Prompt user and get topping choices one at a time.
    System.out.println("All pizzas come with cheese.");
    System.out.println("Additional toppings are " +
        "$1.25 each, choose from:");
    System.out.println("Pepperoni, Sausage, " +
        "Onion, Mushroom");
    // If topping is desired,
    // add to topping list and number of toppings
    System.out.print("Do you want Pepperoni? (Y/N): ");
    input = keyboard.nextLine();
    choice = input.charAt(0);
    if (choice == 'Y' || choice == 'y') {
      numberOfToppings += 1;
      toppings = toppings + "Pepperoni ";
    }
    System.out.print("Do you want Sausage? (Y/N): ");
    input = keyboard.nextLine();
    choice = input.charAt(0);
```

```
if (choice == 'Y' || choice == 'y') {
   numberOfToppings += 1;
   toppings = toppings + "Sausage ";
}
System.out.print("Do you want Onion? (Y/N): ");
input = keyboard.nextLine();
choice = input.charAt(0);
if (choice == 'Y' || choice == 'y') {
   numberOfToppings += 1;
   toppings = toppings + "Onion ";
}
System.out.print("Do you want Mushroom? (Y/N): ");
input = keyboard.nextLine();
choice = input.charAt(0);
if (choice == 'Y' || choice == 'y') {
   numberOfToppings += 1;
   toppings = toppings + "Mushroom ";
}
// Add additional toppings cost to cost of pizza.
cost = cost + (1.25 * numberOfToppings);
// Display order confirmation.
System.out.println();
System.out.println("Your order is as follows: ");
System.out.println(inches + " inch pizza");
System.out.println(crust + " crust");
System.out.println(toppings);
// Apply discount if user is eligible.
// ADD LINES FOR TASK #4 HERE
if (discount) {
   System.out.println("You are eligible for a $2.00 discount");
   cost -= 2;
}
```

```
        // EDIT PROGRAM FOR TASK #5
        // SO ALL MONEY OUTPUT APPEARS WITH 2 DECIMAL PLACES
        System.out.printf("The cost of your order " +
              "is: $%,.2f\n", cost);
        // Calculate and display tax and total cost.
        tax = cost * TAX_RATE;
        System.out.printf("The tax is: $%,.2f\n", tax);
        System.out.printf("The total due is: $%,.2f\n",
              (tax + cost));
        System.out.println("Your order will be ready " +
              "for pickup in 30 minutes.");
    }
}
```

Lab 5

```
/*-----------------------------------------------------------------------
// AUTHOR: Eyad Mohamed AbdelMohsen Ghanem
// FILENAME: Geometry.java
// SPECIFICATION: This program demonstrates static methods
// FOR: CSE 110- Lab #5
// TIME SPENT: 50 minutes
//----------------------------------------------------------*/

import java.util.Scanner;

public class Geometry {
   public static void main(String[] args) {
      int choice; // The user's choice
      double value = 0; // The method's return value
      char letter; // The user's Y or N decision
      double radius; // The radius of the circle
      double length; // The length of the rectangle
      double width; // The width of the rectangle
      double height; // The height of the triangle
      double base; // The base of the triangle
      double side1; // The first side of the triangle
      double side2; // The second side of the triangle
      double side3; // The third side of the triangle

      // Create a scanner object to read from the keyboard
      Scanner keyboard = new Scanner(System.in);

      // The do loop allows the menu to be displayed first
      do {
         // TASK #1 Call the printMenu method
         printMenu();

         choice = keyboard.nextInt();
         switch (choice) {
            case 1:
               System.out.print("Enter the radius of " +
                     "the circle: ");
               radius = keyboard.nextDouble();
               // TASK #3 Call the circleArea method and
               // store the result in the value variable
               value = circleArea(radius);
               System.out.println("The area of the " +
                     "circle is " + value);
               break;
            case 2:
               System.out.print("Enter the length of " +
                     "the rectangle: ");
               length = keyboard.nextDouble();
               System.out.print("Enter the width of " +
                     "the rectangle: ");
               width = keyboard.nextDouble();
               // TASK #3 Call the rectangleArea method and
               // store the result in the value variable
               value = rectangleArea(length, width);
               System.out.println("The area of the " +
                     "rectangle is " + value);
               break;
```

```
case 3:
   System.out.print("Enter the height of " +
         "the triangle: ");
   height = keyboard.nextDouble();
   System.out.print("Enter the base of " +
         "the triangle: ");
   base = keyboard.nextDouble();
   // TASK #3 Call the triangleArea method and
   // store the result in the value variable
   value = triangleArea(base, height);
   System.out.println("The area of the " +
         "triangle is " + value);
   break;
case 4:
   System.out.print("Enter the radius of " +
         "the circle: ");
   radius = keyboard.nextDouble();
   // TASK #3 Call the circumference method and
   // store the result in the value variable
   value = circleCircumference(radius);
   System.out.println("The circumference " +
         "of the circle is " +
         value);
   break;
case 5:
   System.out.print("Enter the length of " +
         "the rectangle: ");
   length = keyboard.nextDouble();
   System.out.print("Enter the width of " +
         "the rectangle: ");
   width = keyboard.nextDouble();
   // TASK #3 Call the perimeter method and
   // store the result in the value variable
   value = rectanglePerimeter(length, width);
   System.out.println("The perimeter of " +
         "the rectangle is " +
         value);
```

```
          break;
        case 6:
          System.out.print("Enter the length of " +
                "side 1 of the " +
                "triangle: ");
          side1 = keyboard.nextDouble();
          System.out.print("Enter the length of " +
                "side 2 of the " +
                "triangle: ");
          side2 = keyboard.nextDouble();
          System.out.print("Enter the length of " +
                "side 3 of the " +
                "triangle: ");
          side3 = keyboard.nextDouble();
          // TASK #3 Call the perimeter method and
          // store the result in the value variable
          value = trianglePerimeter(side1, side2, side3);
          System.out.println("The perimeter of " +
                "the triangle is " +
                value);
          break;
        default:
          System.out.println("You did not enter " +
                "a valid choice.");
      }
      keyboard.nextLine(); // Consume the new line
      System.out.println("Do you want to exit " +
            "the program (Y/N)?: ");
      String answer = keyboard.nextLine();
      letter = answer.charAt(0);
    } while (letter != 'Y' && letter != 'y');
  }

  // TASK #1 Create the printMenu method here
  public static void printMenu() {
    System.out.println("This is a geometry calculator");
    System.out.println("Choose what you would like to calculate");
```

```
      System.out.println("1.\tFind the area of a circle");
      System.out.println("2.\tFind the area of a rectangle");
      System.out.println("3.\tFind the area of a triangle");
      System.out.println("4.\tFind the circumference of a circle");
      System.out.println("5.\tFind the perimeter of a rectangle");
      System.out.println("6.\tFind the perimeter of a triangle");
      System.out.print("Enter the number of your choice: ");
   }
   // TASK #2 Create the value-returning methods here
   // TASK #4 Write javadoc comments for each method

   /**
    * this method returns the area of a circle
    *
    * @param radius
    * @return area of the circle
    */
   public static double circleArea(double radius) {
      return Math.PI * Math.pow(radius, 2);
   }

   /**
    * this method returns the area of a rectangle
    *
    * @param length
    * @param width
    * @return area of the rectangle
    */
   public static double rectangleArea(double length, double width) {
      return length * width;
   }

   /**
    * this method returns the area of a triangle
    *
    * @param base
    * @param height
```

```
     * @return area of the triangle
     */
    public static double triangleArea(double base, double height) {
        return 0.5 * base * height;
    }

    /**
     * this method returns the circumference of a circle
     *
     * @param radius
     * @return circumference of the circle
     */
    public static double circleCircumference(double radius) {
        return 2 * Math.PI * radius;
    }

    /**
     * this method returns the perimeter of a rectangle
     *
     * @param length
     * @param width
     * @return perimeter of the rectangle
     */
    public static double rectanglePerimeter(double length, double width) {
        return 2 * (length + width);
    }

    /**
     * this method returns the perimeter of a triangle
     *
     * @param side1
     * @param side2
     * @param side3
     * @return perimeter of the triangle
     */
    public static double trianglePerimeter(double side1, double side2, double side3)
{
        return side1 + side2 + side3;
    }
}
```

Lab 8

```
/*------------------------------------------------------------
--------------------
// AUTHOR: Eyad Mohamed AbdelMohsen Ghanem
// FILENAME: Lab8.java
// SPECIFICATION: This program tests knowledge of both
objects and arrays, using a class with different
methods helping doing calculations with arrays
// FOR: CSE 110- Lab #8
// TIME SPENT: 20 minutes
//-----------------------------------------------------------
------*/
public class Lab8 {
    public static void main(String[] args) {

        // Create an Arrays object using the first constructor
        Arrays arr1 = new Arrays(5);
        // Print the contents of the array in arr1
        System.out.println(arr1);
        // Call findMin, findMax, and calcAverage on arr1 and
print their values
        System.out.println("Min: " + arr1.findMin());
        System.out.println("Max: " + arr1.findMax());
        System.out.println("Average: " + arr1.calcAverage());
        System.out.println();

        int[] arr = {4, 5, 6};
        Arrays arr2 = new Arrays(arr);
        System.out.println(arr2);
        System.out.println("Min: " + arr2.findMin());
        System.out.println("Max: " + arr2.findMax());
        System.out.println("Average: " + arr2.calcAverage());
        System.out.println();
    }
}
/*------------------------------------------------------------
------------------------------
// AUTHOR: Eyad Mohamed AbdelMohsen Ghanem
// FILENAME: Arrays.java
// SPECIFICATION: This program tests knowledge
of both objects and arrays, using a class with
different methods helping doing calculations
with arrays
// FOR: CSE 110- Lab #8
// TIME SPENT: 20 minutes
//----------------------------------------------
--------------*/

import java.util.Random;

public class Arrays {
    // Instance Variables
    private int[] array;
    private int count;

    // Constructors
    public Arrays(int size) {
        array = new int[size];
        count = size;
        Random rand = new Random();
        for (int i = 0; i < count; i++) {
            array[i] = rand.nextInt(10);
        }
    }

    public Arrays(int[] arr) {
        array = arr;
        count = arr.length;
    }

    // findMin
    public int findMin() {
        int min = array[0];
        for (int i = 1; i < count; i++) {
// reassign min if there is a smaller element
            if (array[i] < min) {
                min = array[i];
            }
        }
        return min;
    }
    // findMax
    public int findMax() {
        int max = array[0];
        for (int i = 1; i < count; i++) {
// reassign max if there is a bigger element
            if (array[i] > max) {
                max = array[i];
            }
        }
        return max;
    }

    // calcSum
    private int calcSum() {
        int sum = 0;
        for (int i = 0; i < count; i++) {
            sum += array[i];
        }
        return sum;
    }

    // calcAverage
    public double calcAverage() {
        return calcSum() / (double) count;
    }

    // toString
    public String toString() {
        StringBuilder output = new
StringBuilder("[ ");
        for (int i = 0; i < count; i++) {
            output.append(array[i]);
            if (i != count - 1) {
                output.append(", ");
            }
        }
        return output + " ]";
    }
}
```

Lab 11

1. Class Month

```
package Labs.Lab11._5_1;

import java.util.ArrayList;
import java.util.Scanner;

public class Month {
    private int monthNumber;

    private final String JAN = "Jan";
    private final String FEB = "Feb";
    private final String MAR = "Mar";
    private final String APR = "Apr";
    private final String MAY = "May";
    private final String JUN = "Jun";
    private final String JUL = "Jul";
    private final String AUG = "Aug";
    private final String SEP = "Sep";
    private final String OCT = "Oct";
    private final String NOV = "Nov";
    private final String DEC = "Dec";

    private final String JANUARY = "January";
    private final String FEBRUARY = "February";
    private final String MARCH = "March";
    private final String APRIL = "April";
    // May is the same short and long
    private final String JUNE = "June";
    private final String JULY = "July";
    private final String AUGUST = "August";
    private final String SEPTEMBER = "September";
    private final String OCTOBER = "October";
    private final String NOVEMBER = "November";
    private final String DECEMBER = "December";

    private final ArrayList<String> shortMonthNames = new ArrayList<>() {
        {
            add(JAN);
            add(FEB);
            add(MAR);
            add(APR);
            add(MAY);
            add(JUN);
            add(JUL);
            add(AUG);
            add(SEP);
            add(OCT);
            add(NOV);
            add(DEC);
        }
    };

    private final ArrayList<String> longMonthNames = new ArrayList<>() {
        {
            add(JANUARY);
            add(FEBRUARY);
            add(MARCH);
            add(APRIL);
            add(MAY);
            add(JUNE);
            add(JULY);
            add(AUGUST);
            add(SEPTEMBER);
            add(OCTOBER);
            add(NOVEMBER);
            add(DECEMBER);
        }
    };

    public Month() {
        monthNumber = 1;
    }

    public Month(int monthNumber) {
        if (monthNumber < 0 || monthNumber > 12) {
            this.monthNumber = 1;
            return;
        }
        this.monthNumber = monthNumber;
    }

    public Month(String monthName) {
        Scanner scanner = new Scanner(System.in);

        while (!(shortMonthNames.contains(monthName) ||
longMonthNames.contains(monthName))) {
            System.out.print("Month name is invalid, re-enter: ");
            monthName = scanner.next();
        }

        switch (monthName) {
            case JAN:
            case JANUARY:
                monthNumber = 1;
                break;
            case FEB:
            case FEBRUARY:
                monthNumber = 2;
                break;
            case MAR:
            case MARCH:
                monthNumber = 3;
                break;
            case APR:
            case APRIL:
                monthNumber = 4;
                break;
            case MAY:
                monthNumber = 5;
                break;
            case JUN:
```

```
            case JUNE:
                monthNumber = 6;
                break;
            case JUL:
            case JULY:
                monthNumber = 7;
                break;
            case AUG:
            case AUGUST:
                monthNumber = 8;
                break;
            case SEP:
            case SEPTEMBER:
                monthNumber = 9;
                break;
            case OCT:
            case OCTOBER:
                monthNumber = 10;
                break;
            case NOV:
            case NOVEMBER:
                monthNumber = 11;
                break;
            case DEC:
            case DECEMBER:
                monthNumber = 12;
                break;
        }
    }

    public void setMonthNumber(int monthNumber) {
        if (monthNumber < 0 || monthNumber > 12) {
            this.monthNumber = 1;
            return;
        }
        this.monthNumber = monthNumber;
    }

    public int getMonthNumber() {
        return monthNumber;
    }

    public String getMonthName() {
        switch (monthNumber) {
            case 1:
                return JANUARY;
            case 2:
                return FEBRUARY;
            case 3:
                return MARCH;
            case 4:
                return APRIL;
            case 5:
                return MAY;
            case 6:
                return JUNE;
            case 7:
                return JULY;
            case 8:
                return AUGUST;
            case 9:
                return SEPTEMBER;
            case 10:
                return OCTOBER;
            case 11:
                return NOVEMBER;
            case 12:
                return DECEMBER;
            default:
                return null;
        }
    }

    public String toString() {
        return getMonthName();
    }

    public boolean equals(Month month) {
        return month.getMonthNumber() == this.getMonthNumber() ||
month.getMonthName().equals(this.getMonthName());
    }
}
```

2. Class CashRegister

```
public class CashRegister {
    private final double TAX_RATE = 0.06;
    private RetailItem retailItem;
    private int quantity;

    public CashRegister(RetailItem retailItem, int quantity) {
        this.retailItem = retailItem;
        this.quantity = quantity;
    }

    public double getSubtotal() {
        return retailItem.getPrice() * quantity;
    }

    public double getTax() {
        return retailItem.getPrice() * TAX_RATE;
    }

    public double getTotal() {
        return getSubtotal() + getTax();
    }
}
```

3.Class RetailItem

```
package Labs.Lab11._6;

public class RetailItem {
    private String description;
    private int units;
    private double price;

    public RetailItem(String description, int units, double price) {
        this.description = description;
        this.units = units;
        this.price = price;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    public void setUnits(int units) {
        this.units = units;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    public String getDescription() {
        return description;
    }

    public int getUnits() {
        return units;
    }

    public double getPrice() {
        return price;
    }

    public String toString() {
        return "Item description: " + description + "\tNumber of units: "
+ units + "\tPrice: " + price;
    }

    public void decrementInventory(int units) {
        if (units > 0)
            units -= 1;
        else
            units = 0;
        this.units = units;
    }
}
```

4. Class Person

```
package Labs.Lab11._7;

public class Customer extends Person {
    private int customerNumber;
    private boolean subscribedToMailingList;

    public Customer() {
    }

    public Customer(int customerNumber, boolean subscribedToMailingList) {
        this.customerNumber = customerNumber;
        this.subscribedToMailingList = subscribedToMailingList;
    }

    public Customer(String name, String address, int telephoneNumber, int
customerNumber, boolean subscribedToMailingList) {
        super(name, address, telephoneNumber);
        this.customerNumber = customerNumber;
        this.subscribedToMailingList = subscribedToMailingList;
    }

    public int getCustomerNumber() {
        return customerNumber;
    }

    public void setCustomerNumber(int customerNumber) {
        this.customerNumber = customerNumber;
    }

    public boolean isSubscribedToMailingList() {
        return subscribedToMailingList;
    }

    public void setSubscribedToMailingList(boolean
subscribedToMailingList) {
        this.subscribedToMailingList = subscribedToMailingList;
    }
}
```

5. Class Customer

```
public class Customer extends Person {
    private int customerNumber;
    private boolean subscribedToMailingList;

    public Customer() {
    }
    public Customer(int customerNumber, boolean subscribedToMailingList) {
        this.customerNumber = customerNumber;
        this.subscribedToMailingList = subscribedToMailingList;
    }

    public Customer(String name, String address, int telephoneNumber, int
customerNumber, boolean subscribedToMailingList) {
        super(name, address, telephoneNumber);
        this.customerNumber = customerNumber;
        this.subscribedToMailingList = subscribedToMailingList;
    }

    public int getCustomerNumber() {
        return customerNumber;
    }

    public void setCustomerNumber(int customerNumber) {
        this.customerNumber = customerNumber;
    }

    public boolean isSubscribedToMailingList() {
        return subscribedToMailingList;
    }

    public void setSubscribedToMailingList(boolean
subscribedToMailingList) {
        this.subscribedToMailingList = subscribedToMailingList;
    }
}
```

6. Class Geometry

```
package Labs.Lab11._8;

public class Geometry {
    public static double areaCircle(double radius) {
        if (radius < 0) {
            System.out.println("invalid value");
            return 0;
        }
        return Math.PI * Math.pow(radius, 2);
    }

    public static double areaRectangle(double length, double width) {
        if (length < 0 || width < 0) {
            System.out.println("invalid values");
            return 0;
        }
        return length * width;
    }

    public static double areaTriangle(double base, double height) {
        if (base < 0 || height < 0) {
            System.out.println("invalid values");
            return 0;
        }
        return 0.5 * base * height;
    }
}
```

7. Class GeometryDemo

```
package Labs.Lab11._8;

import java.util.Scanner;

public class GeometryDemo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        char repeat;

        do {
            int input;
            boolean invalidInput;

            System.out.println("Geometry Calculator");
            System.out.println("1. Calculate the Area of a Circle");
            System.out.println("2. Calculate the Area of a Rectangle");
            System.out.println("3. Calculate the Area of a Triangle");
            System.out.println("4. Quit");
            System.out.println();
            System.out.print("Enter your choice (1-4): ");

            input = scanner.nextInt();
            invalidInput = input != 1 && input != 2 && input != 3 && input
!= 4;

            while (invalidInput) {
                System.out.print("Invalid input, please re-enter your
choice: ");
                input = scanner.nextInt();
                invalidInput = input != 1 && input != 2 && input != 3 &&
input != 4;
            }

            switch (input) {
                case 1:
                    System.out.print("Enter circle's radius: ");
                    double radius = scanner.nextDouble();
                    System.out.printf("Circle's area: %,.2f\n",
Geometry.areaCircle(radius));
                    break;
                case 2:
                    System.out.print("Enter rectangle's length: ");
                    double length = scanner.nextDouble();
                    System.out.print("Enter rectangle's width: ");
                    double width = scanner.nextDouble();
                    System.out.printf("Rectangle's area: %,.2f\n", 
Geometry.areaRectangle(length, width));
                    break;
                case 3:
                    System.out.print("Enter triangle's base: ");
                    double base = scanner.nextDouble();
                    System.out.print("Enter rectangle's height: ");
                    double height = scanner.nextDouble();
                    System.out.printf("Triangle's area: %,.2f\n", 
Geometry.areaTriangle(base, height));
                    break;
                default:
                    break;
            }

            System.out.println();
            System.out.println("Do you want to calculate another shape's 
area? y/n");
            repeat = scanner.next().charAt(0);
        } while (repeat == 'y' || repeat == 'Y');
    }
}
```