Introduction to Java
Exercises for Packet 4: Classes and Objects

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

        Duck d1 = new Duck();
        d1.quack();
        int e = d1.reduce();
        e = d1.reduce();
        e = d1.reduce();
        System.out.println( e );

        Duck d2 = new Duck( 5 );
        e = d2.reduce();
        System.out.println( e );
        d2.quack();
    }
}
```

```
public class Duck {
    private int feathers;

    public Duck( int n ) {
        feathers = n;
    }

    public Duck(){
        feathers = 14;
    }

    public void quack(){
        for ( int k = 1; k<=3; k++ )
            System.out.println("quack");
    }

    public int reduce() {
        feathers--;
        return feathers;
    }
}
```

Use the above classes to answer questions 1 to 8.

1. What is/are the instance variable(s) in the Duck class? **feathers**

2. What is/are the instance variable(s) in the Pond class? **none**

3. How many constructors does the Duck class have? **2**

4. How many methods does the Duck class have? **2, quack and reduce**

5. How many methods does the Pond class have? **1, the main method**

6. When this code is run, what is displayed? (ignore the line breaks)

quack quack quack 11 4 quack quack quack

7. Variable d1 is:
   a) a parameter
   b) an instance variable
   c) a local variable

C, a local variable

8. Write what is displayed in the space provided.

```
public class Runner {
        public static void main(String[] args) {
                Thing w = new Thing();
                w.set( 3 );
                int num = w.get();
___8___         System.out.println( num );

                Thing joe= new Thing( 12 );
                joe.set( -5 );
                num = joe.get();
___7___         System.out.println( num );
        }
}
```

```
public class Thing{
        private int x;

        public Thing() {
                x = 5;
        }

        public Thing( int n ) {
                x = n;
        }
        public int get() {
                return x;
        }

        public void set( int h ) {
                x = x + h;
        }
}
```

9. Write what is displayed in the space provided.

```
public class Bunny {
        public static void main(String[] args) {
                Rumpus roo = new Rumpus( 3, 7 );
                int n = roo.clang( 5 );
___15___        System.out.println( n );
                roo.roar();
                n = roo.get();
___7___         System.out.println( n );

        }
}
```

10. In the Rumpus class, name all the:

parameters: **n1, n2, k**

instance variables : **a, b**

local variables : **n, c**

```
public class Rumpus{
        private int a, b;

        public Rumpus ( int n1, int n2 ) {
                a = n1;
                b = n2;
        }

        public int clang( int k ) {
                int n = a + b + k;
                return n;
        }

        public void roar() {
                int c = a;
                a = b;
                b = c;
        }

        public int get(){
                return a;
        }
}
```

11. What is the correct way to create an object of the English class?

a) English e = English(65);
b) English e = new English(65);
c) English e = English();
d) English e = new English(); **D**

```
public class English{
        private double fun;

        public English() {
                fun = 11;
        }
}
```

12. Write what is displayed in the space provided.

```
public class Runner {
        public static void main(String[] args) {
                Wahoo deb = new Wahoo( 6 );
                deb.set( 2 );
                int num = deb.get();
 12             System.out.println( num );

                Wahoo moe= new Wahoo();
                moe.set( -5 );
                num = moe.get();
 -15            System.out.println( num );
        }
}
```

```
public class Wahoo{
        private int z;

        public Wahoo () {
                z = 3;
        }

        public Wahoo ( int n ) {
                z = n;
        }
        public int get() {
                return z;
        }

        public void set( int x ) {
                z = x * z;
        }
}
```

13. What is displayed?

```
public class Sink{
    public static void main( String [] args ){
        Soap s1 = new Soap();
        Soap s2 = new Soap( 23 );
        s2.use( 17 );
        s1.use( 4 );
        int a = s1.getSize();
        int b = s2.getSize();
        System.out.println( a + ", " + b );   16,  6
        for ( int i = 2; i <= 4; i++ ){
            s1.use( 2 );
            s2.use( i );
        }
        a = s1.getSize();
        b = s2.getSize();
        System.out.println( a + ", " + b );   10,  0
    }
}
```

```
public class Soap{
    private int size;

    public Soap( int x ){
        size = x;
    }

    public Soap(){
        size = 20;
    }

    public void use(int minutes){
        size -= minutes;
        if ( size < 0 )
            size = 0;
    }

    public int getSize(){
        return size;
    }
}
```

14. Assume this code compiles. The variable g must be: **A**

a) a parameter
b) a local variable
c) an instance variable

```
// a method within a class

public void method3( double g ){
        double h = g + k;
        System.out.println( h );
}
```

15. What is displayed?

```
public class WagonTrain{
    public static void main( String [] args ){
        Wagon a = new Wagon( 14 );
        Wagon b = new Wagon( 8 );
        Wagon c = new Wagon( 16 );
        int x = b.add( 11 );
        System.out.println( x );  ___________3___________
        x = c.add( 9 );
        System.out.println( x );  ___________0___________
        x = a.add( 13 );
        System.out.println( x );  ___________0___________

        x = a.add( 5 );
        System.out.println( x );  ___________4___________
        x = b.add( 7 );
        System.out.println( x );  ___________7___________
        x = c.add( 13 );
        System.out.println( x );  ___________6___________
    }
}
```

```
public class Wagon{
    private int items;
    private int max;

    public Wagon( int m){
        items = 0;
        max = m;
    }

    public int add( int i ){
        items += i;
        if ( items > max ){
            int num = items - max;
            items = max;
            return num;
        }
        return 0;
    }
}
```

16. What is displayed?

```
public class Runner{
  public static void main( String [] args ){
    Bank b1 = new Bank( 20000 );
    Bank b2 = new Bank( 10000 );
    b2.deposit( 3000 );
    b2.deposit( -4000 );
    b1.deposit( 2000 );
    System.out.println( b1.balance() );  ____22000.0____
    System.out.println( b2.balance() ); ____13000.0____

    System.out.println( b2.withdraw( 14000 ) );  ______0.0__
    System.out.println( b1.withdraw( 21000 ) ); __21000.0__

    b1.addInterest();
    b2.addInterest();
    System.out.println( b1.balance() ); ____1100.0____
    System.out.println( b2.balance() ); ____14300.0___
  }
}
```

```
public class Bank{
   private double money;

   public Bank( double m ){
      money = m;
   }

   public void addInterest(){
      money +=  0.1*money;
   }

   public void deposit(double m){
      if ( m <= 0 )
         return;

      money += m;
   }

   public double balance(){
      return money;
   }

   public double withdraw(double m){
      if ( m > money )
         return 0;
      else {
         money = money - m;
         return m;
      }
   }
}
```

17. This method will not compile. Explain.
It returns an 8 when the return type is void.
	// a method within a class
	public void method1() {
	    return 8;
	}

Assuming the code to the right compiles,

18. The first line of the Arg class constructor is:
a) public Arg()
b) public void Arg(int a, int b);
c) public Arg(int a, int b)

C

19. The first line of the avast method is:
a) public avast()
b) public void avast()
c) public void avast(int x)

B

20. The first line of the parrot method is:
a) public void parrot (double x)
b) public double parrot ()
c) public void parrot (int x)
d) public int parrot ( )

A

21. The first line of the ayeMatey method is:
a) public int ayeMatey(double x)
b) public int ayeMatey()
c) public int ayeMatey(int x)
d) public void ayeMatey()

B

```
public class PirateRunner {
        public static void main(String[] args) {
                Arg g = new Arg( 5, 6 );
                g.avast();
                g.parrot( 4.78 );
                int x = g.ayeMatey();
                System.out.println( x );
        }
}
```

22. This method will not compile. Why?

Missing return statement.

```
// a method within a class
public int method2( int x ) {
        if ( x < 0 )
                return -1;
        else
                x = x + 5;
}
```

23. What is displayed?

```
public class Runner {
    public static void main(String[] args) {
        Example2 e = new Example2();
        System.out.println( e.met( 6.7 ) );   7
        System.out.println( e.met( 43.4) );   43
        System.out.println( e.met( -8.8 ) );  -9
    }
}
```

```
public class Example2{
    public Example2() {
    }

    public int met( double n ){
        if ( n >= 0 )
            return (int)( n + 0.5 );
        else
            return (int)( n - 0.5 );
    }
}
```

****
Some of the problems after this point deal with the boolean data type.
****

24. This compiles and runs. What is displayed?

```
public class Runner {
        public static void main(String[] args) {
                boolean p = true;
                boolean q = false;
                System.out.println( 3 < 5 );              true
                System.out.println( p && q );             false
                System.out.println( p || q );             true
        }
}
```

25. What does the following code display?

```
public class RunAlarm{
    public static void main( String [] args ){
        AlarmClock pavan = new AlarmClock(605);
        AlarmClock nick = new AlarmClock();
        nick.check( 605 );
        pavan.check( 605 );

        if ( nick.morning( 915 ) )
            System.out.println( "AM" );
        else
            System.out.println( "AM NOT" );

        if ( pavan.morning( 915 ) )
            System.out.println( "AM" );
        else
        System.out.println( "AM NOT" );

        if ( nick.morning( 1943 ) )
            System.out.println( "AM" );
        else
            System.out.println( "AM NOT" );
    }
}
```

zzz

!!!

AM

AM

AM NOT

```
public class AlarmClock{
    private int hour, min;
    private boolean on;

    public AlarmClock( int time){
        hour = time / 100;
        min = time % 100;
        on = true;
    }

    public AlarmClock( ){
        hour = 0;
        min = 0;
        on = false;
    }

    public void check( int time ){
        if ( on == false ) {
            System.out.println ( "zzz" );
            return;
        }

        int h = time / 100;
        int m = time % 100;
        if ( h == hour && m == min  )
            System.out.println ( "!!!" );
        else
            System.out.println ( "zzz" );
    }

    public boolean morning( int time ){
        int h = time / 100;
        return h < 12;
    }
}
```

26. *Create a robot. Then have it move 3 feet, turn to the right and move 2 more feet. Assume there are no obstacles.*

```
public class FieldTest1 {
        public static void main(String[] args) {
        Robot r = new Robot();
        r.move( 3 );
        r.turn( 90 );
        r.move( 2 );
        }
}
```

27. *Create a robot. If the space in front is open, tell it to move 100 feet. If the space in front is not open, tell it to turn 180 degrees and move 100 feet. Afterwards determine how far it actually travelled and print that number.*

```
public class FieldTest2 {
        public static void main(String[] args) {
                Robot r = new Robot();
                if ( r.front() == false )
                        r.turn( 100 );

                r.move(100);
                int n = r.howFar();
                System.out.println( n );


        }
}
```

28. *Create a robot. Then write a loop where it will keep moving in one foot increments as long as the space in front of it is open. This can be done in the space provided.*

```
public class FieldTest3 {
        public static void main(String[] args) {
                Robot r = new Robot();
                while ( r.front() ){
                        r.move(1);
                }
        }
}
```

```
public class Robot{
    instance variables

    public Robot(){
        code
        initial direction is 0 deg.
    }

    public void move( int ft ){
        try to move ft feet forward.
If there is an obstacle in its way, it
will move as far as it can
    }

    public boolean front(){
        return true if the one foot
space in front of the robot is open;
otherwise it returns false
    }

    public void turn( int degrees ){
        turn, + is to the right, - is
to the left
    }

    public int howFar(){
        returns how many feet the
robot has traveled
    }
}
```

29. Write the first line of the Pen constructor and its two methods. Use the code on the left to determine what the two methods are.

```
public class Writer {
     public static void main(String[] args) {
          Pen p = new Pen( 20 );
          int count = 0;
          while ( p.hasInk() ){
               p.use();
               count++;
          }

          System.out.print( "You used the pen " );
          System.out.print( count );
          System.out.print( " times before it ");
          System.out.println( " ran out of ink." );
     }
}
```

```
public class Pen{
     // instance variables

Write out the first line of the constructor
     public Pen( int x ) {
          code
     }

Write out the first line of a method
     public boolean hasInk() {
          code
     }

Write out the first line of another method
     public void use() {
          code
     }
}
```

30. Write the first line of the Door constructor and its two methods. Use the code on the left to determine what the two methods are.

```
public class Building {
     public static void main( String [] args ){
          Door d = new Door( 3.1, 7.1 );

          if ( d.isOpen() == false )
               d.openIt( true );
     }
}
```

```
public class Door {
     // instance variables

Write out the first line of the constructor
     public Door( double a, double b ) {
          code
     }

Write out the first line of a method
     public boolean isOpen() {
          code
     }

Write out the first line of another method
     public void openIt( boolean b ) {
          code
     }
}
```