Answers: Test 2: Functions, Classes, and Objects (202pts)

Name______________________________

Recognize Elements (40pts)

```
public class Fly{

  public Fly(){
    isAlive = true;
  }

  public void setLilyPad(LilyPad l){
    lilypad = l;
  }
  public void flyForward(){
    if(lilypad.getNext() != null){
      this.lilypad.setFly(null);
      this.lilypad = lilypad.getNext();
      this.lilypad.setFly(this);
    }
  }


  public void flyBackward(){
    if(lilypad.getPrevious() != null){
      this.lilypad.setFly(null);
      this.lilypad = lilypad.getPrevious();
      this.lilypad.setFly(this);
    }
  }

  public void die(){
    isAlive = false;
  }

  public boolean getIsAlive(){
    return isAlive;
  }

  public String toString(){
    return isAlive ? "Fly buzzes!" : "...";
  }

  private boolean isAlive;
  private LilyPad lilypad;
}
```

```
public class Frog{

  public Frog(){
    eatCount = 0;
    isAlive = true;
  }

  public void setLilyPad(LilyPad l){
    this.lilypad = l;
  }

  public void eat(Fly fly){
    if(fly != null){
      if(fly.getIsAlive()){
        fly.die();
        eatCount++;
      }
    }
  }


  public void jumpForward(){
    if(lilypad.getNext() != null){
      this.lilypad.setFrog(null);
      this.lilypad = lilypad.getNext();
      this.lilypad.setFrog(this);
    }
  }


  public void jumpBackward(){
    if(lilypad.getPrevious() != null){
      this.lilypad.setFrog(null);
      this.lilypad = lilypad.getPrevious();
      this.lilypad.setFrog(this);
    }
  }


  public int getEatCount(){
    return eatCount;
  }

  public void die(){
    isAlive = false;
  }

  public boolean getIsAlive(){
    return isAlive;
  }

  public LilyPad getLilyPad(){
    return lilypad;
  }

  public String toString(){
    return isAlive ? "Frog says, ribbit! It
has eaten " + eatCount + " flies." : "...";
  }

  private boolean isAlive;
  private int eatCount;
  private LilyPad lilypad;
}
```

```
public class LilyPad{

  public void setPrevious(LilyPad lilypad){
    previous = lilypad;
  }

  public void setNext(LilyPad lilypad){
    next = lilypad;
  }

  public LilyPad getNext(){
    return next;
  }

  public LilyPad getPrevious(){
    return previous;
  }

  public void setSnake(Snake snake){
    this.snake = snake;

    if(snake != null){
      this.snake.setLilyPad(this);
    }
  }

  public void setFrog(Frog frog){
    this.frog = frog;

    if(frog != null){
      this.frog.setLilyPad(this);
    }
  }

  public void setFly(Fly fly){
    this.fly = fly;

    if(fly != null){
      this.fly.setLilyPad(this);
    }
  }

  public Snake getSnake(){
    return snake;
  }

  public Fly getFly(){
    return fly;
  }

  public Frog getFrog(){
    return frog;
  }

  public String toString(){
   String result = "";

   result += "Lilypad: ";

   if(fly != null){
     result += fly;
   }

   if(frog != null){
     result += frog;
   }

   if(snake != null){
     result += snake;
   }

   result += "\n";
   return result;
  }

  private LilyPad previous;
  private LilyPad next;
  private Fly fly;
  private Frog frog;
  private Snake snake;

}
```

```
public class Snake{

  public Snake(){
    eatCount = 0;
  }

  public void eat(Frog frog){
    if(frog != null){
      if(frog.getIsAlive()){
        frog.die();
        eatCount++;
      }
    }
  }


  public void swimForward(){
    if(lilypad.getNext() != null){
      this.lilypad.setSnake(null);
      this.lilypad = lilypad.getNext();
      this.lilypad.setSnake(this);
    }
  }


  public void swimBackward(){
    if(lilypad.getPrevious() != null){
      this.lilypad.setSnake(null);
      this.lilypad = lilypad.getPrevious();
      this.lilypad.setSnake(this);
    }
  }


  public int getEatCount(){
    return eatCount;
  }

  public LilyPad getLilyPad(){
    return lilypad;
  }

  public void setLilyPad(LilyPad l){
    this.lilypad = l;
  }

  public String toString(){
    return "Snake hisses! It has eaten " +
eatCount + " frogs.";
  }

  private int eatCount;
  private LilyPad lilypad;
}
```

```
public class Lake{
  public static void main (String [] args){
     Lake lake = createLake();

     System.out.println(lake);
     lake.step();
     System.out.println(lake);
     lake.step();
     System.out.println(lake);
     lake.step();
     System.out.println(lake);
  }

  private static Lake createLake(){
    LilyPad l1 = new LilyPad();
    LilyPad l2 = new LilyPad();
    LilyPad l3 = new LilyPad();
    LilyPad l4 = new LilyPad();

    Fly f1 = new Fly();
    Fly f2 = new Fly();

    Snake s1 = new Snake();

    Frog frog = new Frog();

    l1.setNext(l2);
    l2.setPrevious(l1);
    l2.setNext(l3);
    l3.setPrevious(l2);
    l3.setNext(l4);
    l4.setPrevious(l3);

    l2.setFly(f1);
    l4.setFly(f2);

    l3.setSnake(s1);

    l1.setFrog(frog);
    return new Lake(l1, s1, f1, f2, frog);
  }


  public Lake(LilyPad lilypad, Snake snake,
Fly f1, Fly f2, Frog frog){
    this.lilypad = lilypad;
    this.snake = snake;
    this.f1 = f1;
    this.f2 = f2;
    this.frog = frog;
  }

  public void step(){
    snake.swimForward();
    frog.jumpForward();

    frog.eat(frog.getLilyPad().getFly());
    snake.eat(snake.getLilyPad().getFrog());
  }


  public int chainSize(){
    int i = 0;
    LilyPad l = this.lilypad;
    i++;

    while(l.getNext() != null){
     l = l.getNext();
     i++;
    }

    return i;
  }


  public LilyPad padAt(int index){
    LilyPad l = this.lilypad;

    for(int i = index; i > 0; i--){
      if(l.getNext() != null){
        l = l.getNext();
      }
    }

    return l;
  }


  public String toString(){
    String result = "Lake state: \n";

    for(int i = 0; i < this.chainSize(); i++){
      result += padAt(i);
    }

    result += "\n";
    return result;
  }

  private LilyPad lilypad;
  private Snake snake;
  private Fly f1;
  private Fly f2;
  private Frog frog;
}
```

1. Name the 5 classes in the program above. Which class has the main function.
Fly, Frog, LilyPad, Snake, Lake

2. Write the signature of the Lake class constructor.
Lake(LilyPad, Snake, Fly, Fly, Frog)

3. List the names of the methods in the Snake class.
Eat, swimForward, swimBackward, getEatCount, getLilyPad, setLilyad, toString

4. List the member variables of the LilyPad class.
previous, next, fly, frog, snake

5. In main, how many instances of the LilyPad class are created?
4

Trace Programs (40pts)

Use the code from above to determine the output of the following snippet:

1.

```
LilyPad a = new LilyPad();
a.setFly(new Fly());

LilyPad b = a;
b.setFrog(new Frog());

System.out.println(a);
```

Lilypad: Fly buzzes!Frog says, ribbit! It has eaten 0 flies.

2.

```
LilyPad lily1 = new LilyPad();
LilyPad lily2 = new LilyPad();

lily1.setNext(lily2);
lily2.setPrevious(lily1);

Fly f1 = new Fly();
Fly f2  = new Fly();
Snake s1 = new Snake();
Frog frog = new Frog();

lily1.setFly(f1);
lily2.setFly(f2);
lily1.setFrog(frog);
lily1.setSnake(s1);

frog.eat(lily1.getFly());

Lake lake = new Lake(lily1, s1, f1, f2, frog);

System.out.println(lake);
```

Lake state:
Lilypad: ...Frog says, ribbit! It has eaten 1 flies.Snake hisses! It has eaten 0 frogs.
Lilypad: Fly buzzes!

Use an Existing API (40pts)

```
public class Die{

  public Die(int numberOfSides, String name){
    this.numberOfSides = numberOfSides;
    this.name = name;
  }


  public Die(int numberOfSides){
    this.numberOfSides = numberOfSides;
  }


  public int roll(){
    this.currentSide = (int) Math.ceil((Math.random() * numberOfSides));
    return this.currentSide;
  }


  public int getCurrentSide(){
    return currentSide;
  }


  public int getNumberOfSides(){
    return numberOfSides;
  }


  public String getName(){
    return name;
  }


  public String toString(){
    return name + " " + currentSide;
  }


  public int getMaxRoll(){
    return numberOfSides;
  }


  private int currentSide;
  private int numberOfSides;
  private String name;
}
```

1. Using the code on the previous page, write a program that creates two dice. The first die should have 12 sides. The second die should have 7. Roll each die three times. Look to see what the current side of each die is on the third roll. Get the sum of the value of both die together on the third roll and compare that value to the maximum possible roll. If the roll of the sum of both die is less than the sum of the maximum value of each die, print out how far away the third roll was from the maximum. If the roll was the maximum possible roll, congratulate the player.

```
Die a = new Die(12, "a");
Die b = new Die(7, "b");

for(int i = 0; i < 3; i++){
  a.roll();
  b.roll();
}

int aThirdRollValue = a.getCurrentSide();
int bThirdRollValue = b.getCurrentSide();
int sumOfThirdRoll = aThirdRollValue + bThirdRollValue;

System.out.println(a);
System.out.println(b);

int maximumPossibleRoll = a.getNumberOfSides() + b.getNumberOfSides();

if(sumOfThirdRoll < maximumPossibleRoll){
  System.out.println("Distance from maximum roll: " + (maximumPossibleRoll - sumOfThirdRoll));
} else{
  System.out.println("Congrats! You rolled the highest possible roll!");
}
```

Questions (32pts)

1. In object-oriented programming, what does 'this' refer to?

The current instantiated object that is calling the method in which this occurs.

2. What is the difference between pass by value and pass by reference?

A parameter that is passed by value sends a copy of the data to the function. Changes to the copy are not reflected in the original variable. A parameter that is passed by reference, passes a pointer to the location in memory where the variable is stored. Changes to the data in the variable are reflected in the variable once the function has finished running. This is because inside the function, the actual variables data was changed not a copy of the data. In Java, objects are passed by reference and primitive variable types are passe by value. This may be handled differently in other programming languages.

3. What is the difference between a function and a method?

A method is any function that elongs to a class or instance of that class. A function is a stored set of instructions that takes parameters as input and produces an output or return value based on the inputs. Functions do not need to belong to objects, though they may. All methods are functions. Not all functions are methods. In Java, most functions (and all we have seen this semster) are methods. The story is different in other languages.

4. If we make all of our member variables public, we would no longer have to write extra methods to get and set their values (as they would be publicly available-- object.value). Why do we go through the trouble of writing these extra methods and making our variables private?

We do this to ensure encapsulation. It is good to tightly control how object data is manipulated via a well-defined API of public methods. This ensures our objects are used and behave as the programmer intended.

Design an Object-Oriented Program (50pts)

Make a class named SolarSystem that contains a main function. The SolarSystem will contain one Star object and 3 Planets. Each planet will have two moons. The planets belong to the star, and the moons belong to the planets. In addition to the SolarSystem class, your program should have a Star, Planet, and Moon class. The SolarSystem should have a Star member variable. The Star should have Planet member variables. And the planets should have Moon member variables. Each SolarSystem, Star, Planet, and Moon should have a name and a color. The name and color should be set in the constructor. These properties should be set to private and should be made accessible through an appropriate getter method. Each class should have a toString() method. There should also be getters for member variables of type Star, Planet, and Moon.

```
public class Moon {
  public Moon(String name, String color){
    this.name = name;
    this.color = color;
  }

  public String getName(){
    return name;
  }

  public String getColor(){
    return color;
  }

  public String toString(){
    return "Moon: " + color + " " + name;
  }

  private String name;
  private String color;
}
```

```
public class Planet{
  public Planet(String name, String color){
    this.name = name;
    this.color = color;
  }

  public String getName(){
    return name;
  }

  public String getColor(){
    return color;
  }

  public void setMoon1(Moon m){
    this.moon1 = m;
  }

  public void setMoon2(Moon m){
    this.moon2 = m;
  }

  public Moon getMoon1(){
    return moon1;
  }

  public Moon getMoon2(){
    return moon2;
  }

  public String toString(){
    String result = "Planet: " + color + " " + name + " ";
    result = result + moon1 + " " + moon2;

    return result;
  }

  private String name;
  private String color;
  private Moon moon1;
  private Moon moon2;

}
```

```
public class SolarSystem{
  public static void main(String [] args){
    Star sun = new Star("sun", "yellow");

    Planet mercury = new Planet("mercury", "gray");
    Planet venus = new Planet("venus", "red");
    Planet earth = new Planet("earth", "blue");

    Moon m1 = new Moon("m1", "black");
    Moon m2 = new Moon("m2", "black");

    Moon v1 = new Moon("v1", "green");
    Moon v2 = new Moon("v2", "green");

    Moon e1 = new Moon("e1", "gray");
    Moon e2 = new Moon("e2", "gray");

    sun.setPlanet1(mercury);
    sun.setPlanet2(venus);
    sun.setPlanet3(earth);

    mercury.setMoon1(m1);
    mercury.setMoon2(m2);

    venus.setMoon1(v1);
    venus.setMoon2(v2);

    earth.setMoon1(e1);
    earth.setMoon2(e2);

    System.out.println(sun);
  }
}
```

```
public class Star {
  public Star(String name, String color){
    this.name = name;
    this.color = color;
  }

  public void setPlanet1(Planet p1){
    this.planet1 = p1;
  }

  public void setPlanet2(Planet p2){
    this.planet2 = p2;
  }

  public void setPlanet3(Planet p3){
    this.planet3 = p3;
  }

  public Planet getPlanet1(){
    return planet1;
  }

  public Planet getPlanet2(){
    return planet2;
  }

  public Planet getPlanet3(){
    return planet3;
  }

  public String getName(){
    return name;
  }

  public String getColor(){
    return color;
  }

  public String toString(){
    String result = "Star: " + color + " " + name;
    result = result + "\n" + planet1 + "\n" + planet2 + "\
n" + planet3;

    return result;
  }

  private String name;
  private String color;
  private Planet planet1;
  private Planet planet2;
  private Planet planet3;
}
```