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.

2. Write the signature of the Lake class constructor.

3. List the names of the methods in the Snake class.

4. List the member variables of the LilyPad class.

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

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);
```

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);
```

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.

Questions (32pts)

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

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

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

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?

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.