Birthday: Month: ________ Day: ________

Summer 2010 CS61BL Midterm 1

You have 110 minutes to finish this test. Your exam should contain 6 problems (numbered 0-5). This is an open-book test. You may consult any books, notes, or other paper-based inanimate objects available to you. Read the problems carefully. If you find it hard to understand a problem, please ask a question. Please write your answers in the spaces provided in the test; if you need to use the back of a page make sure to clearly tell us so on the front of the page.

Good Luck!

0		out of 1 point
1		out of 2 points
2		out of 6 points
3		out of 3 points
4		out of 3 points
5		out of 9 points
Total		out of 24 points

Write your name on the last page

Birthday: Month: ________ Day: ________

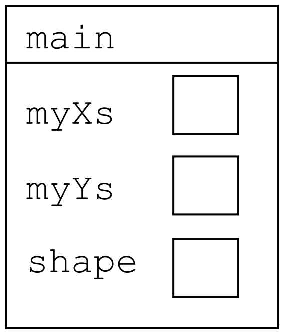
Question 1 (2 points)

Below is a class that creates a geometric shape. Draw a picture of what memory looks like at the spot indicated in the `main` method. Note: The `Point` class has instance variables `x` and `y` of type `int`.

```
import java.awt.Point;

public class GeometricShape {
      public Point[] vertices;
      int sides;
      public GeometricShape(int[] xValues, int [] yValues) {
            sides = xValues.length;
            vertices = new Point[6];
            for (int i = 0; i < sides; i ++){
                  vertices[i] = new Point(xValues[i], yValues[i]);
            }
      }
      public static void main(String [] args){
            int[] myXs = {0, 2, 2, 0};
            int[] myYs = {0, 0, 5, 5};
            GeometricShape shape = new GeometricShape (myXs, myYs);

            // DRAW A PICTURE OF WHAT MEMORY LOOKS LIKE HERE!!!
      }
}
```



Birthday: Month: ________ Day: ________

Question 2 (6 points)

The next three problems deal with the class `Set` from lab and shown below.

Taking advantage of inheritance, define a class `ExpandableSet` that behaves just like the `Set` except that, when `insert` is called with a value to be inserted larger than the `Set` can currently hold, the `Set` doubles in size until the value can be added. `ExpandableSet` should have a no argument constructor that makes the initial size of the `Set` be 1.

```
public class Set {
     // Represent a set of nonnegative ints from 0 to maxElement-1
     // for some initially specified maxElement.
     // contains[k] is true if k is in this set, false if it isn't
     protected boolean[] contains;

     // Initialize a set of ints from 0 to maxElement-1.
     public Set(int maxElement) {
          contains = new boolean[maxElement];
     }
     public void insert(int k) {
          contains[k] = true;
     }
     public void remove(int k) {
          contains[k] = false;
     }
     public boolean member(int k) {
          return contains[k];
     }
}

/*
 * JUnit has the following relevant methods (for Questions 3 and 4)
 */

static void assertTrue(boolean b)

static void assertFalse(boolean b)

static void assertEquals(String message, Object expected, Object actual)

static void fail()
```

Birthday: Month: ________ Day: ________

Please write your solution to Question 2 here:

Birthday: Month: ________ Day: ________

Question 3 (3 points)

Now we are going to test `ExpandableSet` using `JUnit`. When we change the `length` of `ExpandableSet`, we want it to become exactly the `length` we expect. For example, if the `ExpandableSet` should have doubled in length we want it to be exactly double the length, no more, no less.

a) However in the JUnit file we can't just check the `length` of the array. Explain why.

b) Complete the helper method below to appear in your `JUnit` file `ExpandableSetTest`. This method will return `true` if the length of the `ExpandableSet es` is equal to the `int expectedSize`. You may not write any additional methods. Hint: It will be helpful to use the concept of Exceptions.

```
public class ExpandableSetTest extends TestCase {

     private boolean sizeCheckHelper(ExpandableSet es, int expectedSize){











}
```

Birthday: Month: ________ Day: ________

Question 4 (3 points)

Fill in the table below with a convincing set of JUnit test methods for the `insert` method in `ExpandableSet`. You should test 1 important thing about `insert` in each test method. You may use your method `sizeCheckHelper`.

`public class ExpandableSetTest extends TestCase {`	**Description of what this tests**

`public void test1() {`
 	
 `}`	

`public void test2() {`
 	
 `}`	

`public void test3() {`
 	
 `}`	

Birthday: Month: ________ Day: ________

Question 5 (9 points)

This question is based upon Project 1. Relevant details of the `Picture class` and `Pixel class` are provided on the next few pages. The goal is to iterate over all `Pixels` in a `Picture` that are within a Circle. The order in which `Pixels` are returned doesn't matter, but all `Pixels` within the circle should be returned. For example, we want to be able to run the following method in the `Picture` class:

```
     // Colors the picture with a black circle of radius 10,
     // centered at (50, 50)
     public void blackCircle(){
          this.initIterator(50, 50, 10);
          while (this.hasNext()){
               this.next().setColor(new Color(0, 0, 0, 255));
          }
     }
```

Based upon the comments below, implement the following methods for the `Picture class.` You may only add code within the methods defined below. Any other modifications will result in a score of 0 on this question.

```
public class Picture extends SimplePicture {
     // instance variables - You may NOT define any new instance variables

     int midX;              // the X coordinate of the center of the circle
     int midY;              // the Y coordinate of the center of the circle
     int radius;            // the radius of the circle
     Pixel nextPixel;       // Invariant: always references the next Pixel to
                            //            be returned by next() or null if no
                            //             pixels remain.

// Returns true if there are more pixels to return in the circle
// and false otherwise
public boolean hasNext() {
     // if nextPixel is null, there are no more pixels to return
     return nextPixel != null;
}

// HELPER METHOD
// Returns true if the instance variable nextPixel is within the
// the circle and false otherwise.
private boolean isNextPixelWithinRadius() {
     if (nextPixel == null) {
          return false;
     }
     int x = nextPixel.getX();
     int y = nextPixel.getY();
     return (Pixel.distance(x, y, midX, midY) < radius);
}
```

Birthday: Month: ________ Day: ________

```
// This method should restore the invariant described for nextPixel.
// You must call this method at least once in your code on the next page.
// This method may call the optional helper method getUnexploredPixel
private void restoreInvariant() {









}
// OPTIONAL HELPER METHOD
// If you chose to implement this method you must implement exactly what
// the comment specifies.
//
// This method should return the next unexplored Pixel based upon the
// argument lastPixel. Unexplored Pixels should be returned regardless of
// whether they are within the circle. This method does not modify
// nextPixel. The method returns the next Pixel to be explored or null
// if no unexplored Pixels remain.
public Pixel getUnexploredPixel(Pixel lastPixel)
{









}
```

```
// The circle is defined by a radius circleRadius and is centered at
// coordinates (centerX, centerY).
// This method will be called before the first call to hasNext() or next()
public void initIterator(int centerX, int centerY, int circleRadius) {















}
// return successive Pixels within the circle specified in initIterator
public Pixel next() {















}
```

Birthday: Month: ________ Day: ________

```
//*
 * The Picture class has the following relevant methods
 */

// returns the number of pixels that the image is wide.
public int getWidth()

// returns the number of pixels that the image is tall.
public int getHeight()

// returns the pixel at the location (x, y)
public Pixel getPixel(int x, int y)

/*
 * The Pixel class has the following relevant methods
 */

// returns the X coordinate of the current Pixel
public int getX()

// returns the Y coordinate of the current Pixel
public int getY()

// sets the color of the current Pixel to be newColor
public void setColor(Color newColor)

// calculates the distance between two points (x1, y1) and (x2, y2).
public static int distance(int x1, int y1, int x2, int y2)
```

Page intentionally left blank.

You may use it for extra space. If you expect it to be graded you must indicate so clearly on the relevant problem.

Question #0
Write the month and day of your birthday on every page! (worth 1 pt)

Last Name:	
First Name:	
Login:	cs61bl-
TA (circle):	☐ 101 - Karan 8-11 ☐ 102 - Courtney 11-2 ☐ 103 - Stephanie 2-5 ☐ 104 - Eric 2-5
Birthday:	Month: _____ Day: _____