[**Closed book and notes. You may use one side of an 8.5×11 inch sheet of paper that you personally prepared.**] Show all of your work clearly in the space provided or on the additional page at the end of the exam. Be sure to **read each problem carefully**. Note that the exam is double sided. Due to time constraints, you are not required to document your source code.

1. Write code that creates a JPanel that uses the FlowLayout to place four JButtons in the panel. Describe how the buttons will be displayed.

2. List two **listener** classes provided by the Java standard library. For each listener class, give at least two examples of types of events that can be handled by the listener class.

3. During the development of the Media Player, the JLabel that displayed the current track playing was required to be an attribute of the MediaPlayerUI class. Why couldn't it have just been declared in the constructor of the class?

4. Define polymorphism and explain where it was used in the Media Player application.

5. Create a customized class, ExamException, that can be thrown by the following command:

```
throw new ExamException("Question␣is␣too␣hard");
```

6. Is the ExamException thrown in the previous question a check or uncheck exception? Why?

7. What is the purpose of the `finally` keyword? Give one specific example of where it makes sense to use it.

8. Write a GUI program that has one button and a text area. When the program starts, it should read in a bunch (number unknown) phrases from a file called "phrases.txt". Each time the button is pressed one of the phrases from the file (selected at random) should be displayed in the text area.

9. Draw the class diagram for all of the source files provided in Appendix A. Note: all `import`s have been removed to save space.

10. Draw the sequence diagram for the main method in Driver.java from Appendix A.

11. Using the code from Appendix A, classify each of the following statements into one of the following categories:

- **C** will not compile
- **E** will cause an exception at runtime
- **OK** prefectly fine instruction

Explain why.

(a)

```
Drawable draw = new Drawable();
```

(b)

```
Drawable draw = new Shape();
```

(c)

```
Drawable draw = new Triangle();
```

(d)

```
Shape draw = new Triangle();
```

(e)

```
Triangle draw = new Circle();
```

(f)

```
Drawable draw;
Triangle draw2 = draw;
```

(g)

```
Triangle triangle = new Triangle();
triangle.toString();
```

Appendix A

Driver.java

```
public class Driver {

  public static void main(String[] args) {

    Circle cir = new Circle(Math.random());
    cir.draw();
    cir.setRadius(1.0);
    cir.zoom();
    cir.draw();

    Shape shape = new Triangle();
    shape.draw();
  }
}
```

Drawable.java

```
public interface Drawable {

  public void draw();

}
```

Shape.java

```
public abstract class Shape {

  private Color color;
  private double xCoord;
  private double yCoord;

  public Shape(Color color) {
    this.color = color;
    xCoord = 0.0;
    yCoord = 0.0;
    System.out.println("Shape: constructor");
  }

  public abstract void draw();

  public void erase() {
    System.out.println("Shape: erase");
  }

  public void move() {
    xCoord += 5.0;
    yCoord += 5.0;
    System.out.println("Shape: move");
  }
  public void zoom() {
    System.out.println("Shape:␣zoom");
  }

  public Color getColor() {
    System.out.println("Shape:␣getColor");
    return color;
  }

  public void setColor(Color color) {
    System.out.println("Shape:␣setColor");
    this.color = color;
  }

  public String toString() {
    return "Color:␣" + color + "␣(" + xCoord + ",␣" + yCoord + ")";
  }
}
```

Circle.java

```
public class Circle extends Shape implements Drawable, Serializable {

  private double radius;

  public Circle() {
    super(Color.WHITE);
    radius = 0.0;
  }

  public Circle(double radius) {
    super(Color.WHITE);
    this.radius = radius;
    System.out.println("Circle:␣constructor");
  }

  public void draw() {
    System.out.println(toString());
  }

  public void setRadius(double radius) {
    System.out.println("Circle:␣setRadius");
    this.radius = radius;
  }

  public String toString() {
    return "Circle:␣draw␣with␣radius␣" + radius
    + super.toString();
  }

}
```

Triangle.java

```
public class Triangle extends Shape {

  public Triangle() {
    super(Color.WHITE);
    System.out.println("Triangle:␣default␣constructor");
  }

  public void draw() {
    System.out.println("Triangle:␣draw␣" + super.toString());
  }
}
```