

1. True/False [15 pts]

Answer the following questions True or False. Be sure to write out True and False, NOT just a T or F.

- 3** (a) The protected visibility modifier allows a superclass access to the instance variables of a child class.
- 3** (b) Classes in java can be derived from more than one class.
- 3** (c) A list of command-line arguments is available to the java programmer via a parameter to main that is an array of Strings.
- 3** (d) All classes in java inherit directly or indirectly from the Object class.
- 3** (e) An abstract class cannot include non-abstract methods.

2. Arrays and loops [10 pts]

Write a void method called `findMax` that takes an array of integers as a parameter and it finds and prints both the index and the value of the largest element in the array. Your method should print out a String like the following example:

The largest value is 3000 found at array index position 5.

3. Objects and parameter passing [10 pts]

What is printed by the following code?

```
public class MethodCallingDemo {
    private int anInt = 1;
    private boolean someFlag = true;

    private static void changeArgs(int i, boolean b, MethodCallingDemo obj) {
        obj.anInt = 4;
        obj.someFlag = false;
        i = 3;
        b = true;
        obj = new MethodCallingDemo();
    }

    public static void main(String[] args) {
        MethodCallingDemo someObj = new MethodCallingDemo();
        changeArgs(someObj.anInt, someObj.someFlag, someObj);
        System.out.println(someObj.anInt + " " + someObj.someFlag);
    }
}
```

Output:

4. Interfaces [10 pts]

Write a class called MyClass that implements the two interfaces below. Note: Your class must compile cleanly. (Just do reasonable things for your methods.)

```
public interface MathType {
    public double area(double radius);
}

public interface Calculator {
    public double summer(double p1, double p2);
    public double product(double p1, double p2);
}
```

5. Tracing with inheritance [15 pts]

What would be printed out after running the main method below? Assume that the classes are in three separate files as appropriate.

```
public class Animal{
    public Animal(){
        System.out.println("I am an Animal");
    }

    public Animal(String s){
        System.out.println(s);
    }

    public void eat(){
        System.out.println("ALL ANIMALS EAT");
    }
}
```

```
public class Dog extends Animal {
    public Dog(){
        this("I am a Dog");
    }

    public Dog (String s){
        super();
        System.out.println(s);
    }

    public void eat(){
        System.out.println("Dogs EAT TOO!");
    }
}
```

```
public class AnimalTest{
    public static void main(String [] args){
        Dog snoopy = new Dog();
        snoopy.eat();
        Animal dog = new Animal("I am animal named Snoopy!!!");
        dog.eat();
    }
}
```

Output:

6. GUIs [13 pts]

Consider the code below.

```
import java.awt.*;
import javax.swing.*;

public class GUI2 {
    public static void main(String[] args) {
        JFrame jf = new JFrame("My GUI");
        jf.setSize(200,200);
        JPanel jp = new JPanel();
        jp.setLayout(new BorderLayout());
        jf.getContentPane().add(jp);
        JButton jb1 = new JButton("W");
        JButton jb2 = new JButton("X");
        JButton jb3 = new JButton("Y");
        JButton jb4 = new JButton("Z");
        jp.add(jb1,BorderLayout.WEST);
        jp.add(jb2,BorderLayout.NORTH);
        jp.add(jb3,BorderLayout.SOUTH);
        jp.add(jb4,BorderLayout.EAST);
        JPanel jp2 = new JPanel();
        jp2.setLayout(new GridLayout(2,2));
        jp.add(jp2,BorderLayout.CENTER);

        for (int i=0; i<4; i++) {
            jp2.add(new JLabel("thing "+i));
        }
        jf.setVisible(true);
    } //main
} //class GUI2
```

What is displayed on the screen when that program is run? (Draw a picture.)

7. Classes with inheritance [15 pts]

Consider the code below:

```
public class C1 {
    public int a;
    private int b;
    protected int c;
    ...
}

public class C2 extends C1 {
    protected int x;
    private int y;
    ...
}

public class C3 extends C2 {
    protected int m;
    public C3 { super(); }
    ...
}
```

Circle the one best answer for each question below.

- 3** (a) The relationship between a child class and a parent class is referred to as
1. overriding
 2. instance-of
 3. has-a
 4. is-a
 5. was-a
- 3** (b) The instruction `super()` does which of the following?
1. Calls the method `super` as defined in the current class
 2. Calls the method `super` as defined in the current class's parent class.
 3. Calls the method `super` that is defined in `java.lang`
 4. Calls the constructor as defined in the current class's parent class
 5. Calls the constructor as defined in the current class
- 3** (c) Which of the following lists of instance data are the only ones accessible in class C2?
1. x, y
 2. a, x, y
 3. a, c, x, y
 4. a, b, c, x, y
 5. a, b, c, x, y, m

- 3 (d) Which of the following lists of instance data are the only ones accessible in class C3?
1. a, b, c, x, y, m
 2. x, y, m
 3. x, m
 4. a, c, x, m
 5. a, x, m
- 3 (e) Which of the following is true with respect to C1, C2, and C3?
1. C1 and C2 are both subclasses of C3
 2. C2 and C3 are both subclasses of C1
 3. C1, C2, and C3 are all subclasses of the class C
 4. C1 is a subclass of C2, and C2 is a subclass of C3
 5. C3 is a subclass of C2, and C2 is a subclass of C1

8. Two-dimensional arrays [10 pts]

Write a method that takes in as parameters a two-dimensional array of doubles and a double value `boundary`. The method should return the number of values in the array that are less than `boundary`. Note that the 2D array may not be square.

```
public int checker(double[] [] arr, double boundary)
{
```

```
}
```

