CSC 143

Exam 1

Write also your name in the appropriate box of the scantron
Multiple Choice Questions (30 points)

Answer all of the following questions. READ EACH QUESTION CAREFULLY. Fill the correct bubble on your scantron sheet. Each correct answer is worth 1 point. Each question has EXACTLY one correct answer.

1. Consider
   ```java
   public class A {
       public void foo() {/* code */}
   }
   
   public class B extends A {
       public void foo() { /* code */}
   }
   
   When the following lines of code are executed
   ```java
   A a = new B();
   a.foo();
   ```
   what is the static type of this within the foo method called by the statement a.foo()?

   A. A
   B. B
   C. Object
   D. Can’t tell without having the code of foo.
   E. The above code is illegal.
2. Consider
   public class A {
       public void foo() {/* code */}
   }

   public class B extends A {
       public void foo() { /* code */}
   }

   When the following lines of code are executed

   A a = new B();
   a.foo();

   what is the dynamic type of this within the foo method called by the
   statement a.foo()?

   A. A
   B. B
   C. Object
   D. Can’t tell without having the code of foo.
   E. The above code is illegal.
3. Consider the following class definitions

```java
public class Vehicle {
    public Vehicle() {}
}

public class Car extends Vehicle implements Rentable {
    public Rentable getRentable() {
        return /* ??? */
    }
}
```

What could be written in place of /* ??? */ so that the above code compiles correctly?

A. `this;`
B. `new Vehicle();`
C. `new Rentable();`
D. Either A or B
E. Either A or B or C
4. Consider the following inheritance hierarchy

```java
public class OopsException extends Exception{}
public class MoreOopsException extends OopsException {}

public class A {
    public void foo() throws OopsException {}
}

public class B extends A {}
```

What are possible signatures of an implementation of an override of `foo` in the `B` class?

A. `public void foo() throws OopsException`  
B. `public void foo() throws MoreOopsException`  
C. `public void foo() throws MoreOopsException, RuntimeException`  
D. A and B  
E. A, B and C

5. Which statements are true?
   I. A try-catch block can only be used for checked exceptions  
   II. You can have multiple catch blocks for one try block  
   III. return statements and throw statements are legal in a catch block

A. I only  
B. II only  
C. I, II, & III  
D. I & III  
E. II, & III
6. Consider the following two class definitions written within the same file

```java
package disco;

public class Boogie{
    public void setColor (Color c) { …}
    void display(){ …}
    protected Color color;
}

class Woogie extends Boogie{
    public void setColor(Color c) {
        color = c;   // A
    }
    public void show() {
        display();   // B
        System.out.println(text);
    }
    private String text;
}
```

Are statements A and/or B legal? (Will they compile?)

A. Both statements are legal  
B. Neither statement is legal  
C. B is legal, A is not  
D. A is legal, B is not  
E. Can't tell. There is not enough information. I would need to read through the implementation of display.

7. Using these classes one more time... which is true? Assume that any possibly illegal statement (A and/or B) has been removed.  
A. Any client can instantiate disco.Woogie, but not disco.Boogie  
B. Any client can instantiate disco.Boogie, but not disco.Woogie  
C. Any client can instantiate objects of type disco.Boogie and disco.Woogie  
D. As long as the client imports disco, then it can instantiate objects of type disco.Boogie and disco.Woogie  
E. None of the above (A, B, C and D are all false).
8. To create a top level container (e.g. a graphics window), which `javax.swing` class(es) could you instantiate?
   A. JPanel
   B. JFrame
   C. JTextArea
   D. A and B
   E. A, B and C

9. In Java, which of the following is (are) true?
   (I) An event listener can listen to more than one component.
   (II) A component can send an event to more than one event listener
   A. I
   B. II
   C. I and II
   D. None are true

10. When using an MVC framework to program a game (e.g. a chess game), which component would know about the rules of the game (e.g. the motion of the pieces)?
    A. Model
    B. View
    C. Controller
    D. Rules are programmed within the View and Controller with no clear separation
    E. Rules are programmed within the View, Controller and Model with no clear separation
Short answers (50 points)

1) [10 points] Fill out the following table with Y for Yes and N for No.

<table>
<thead>
<tr>
<th></th>
<th>Enables support for multiple inheritance</th>
<th>Supports abstract methods</th>
<th>Implementation allowed (i.e. can you write code within the methods?)</th>
<th>Create an instance of</th>
<th>Partial implementation allowed (i.e. can you write code within some of the methods?)</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>abstract class</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>concrete class</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2) [10 points] Consider the following code:

```java
public class Student {
    private double gpa; // must be between 0.0 and 4.0

    /**
     * Updates the gpa of this student and returns the new gpa.
     * @param courseGrade the grade received for a course
     * (between 0.0 and 4.0).
     * @param credits the number of credits (greater than 0).
     * @return new gpa value.
     */
    public double updateGPA(double courseGrade, int credits) {
        /* code */
    }
}
```

a) Give a class invariant

    _ How would you use the assert keyword to check that invariant?

b) Give one precondition of the setGPA method (which is not a class invariant):

    _ Write a statement with the throw keyword to check that precondition.

c) Give a postcondition of the setGPA method (other than any condition already mentioned).
3) [5 points] In homework 2, you created an abstract class MovingThing that had the two methods

```java
public void moveCenterBy (int dx, int dy) {
    // use moveCenterTo
    // x and y are the coordinates of the center
    // of this MovingThing (declared as protected instance
    // fields)
    int newX = x + dx;
    int newY = y + dy;
    moveCenterTo(newX, newY);
}

public abstract void moveCenterTo (int x, int y);
```

MoveCenterTo was implemented in the derived classes (e.g. Spider). However this approach had the potential danger of generating a StackOverflowError exception. Write an implementation of MoveCenterTo that illustrates this problem:

```java
public void moveCenterTo (int x, int y) {
```

4) [5 points] What is the difference between an adapter and a listener (for example MouseAdapter and MouseListener)? Explain the difference for a developer using one or the other.
5) [5 points]
The ActionListener interface is defined as follows (… actually this is not true, but it is not relevant for this question).

```java
public interface ActionListener {
    public void actionPerformed(ActionEvent e);
}
```

A JButton can send its click event to an action listener using the method with the following signature:

```java
public void addActionListener(ActionListener listener)
```

Complete the code below that creates a listener for a JButton myButton. The listener should print “Button has been clicked” whenever the button is clicked. Use an anonymous inner class.

```java
JButton myButton = new JButton("Click me!");

// Add your code below
```
6) Consider the following classes [15 points]

```java
public abstract class Vehicle {
    public boolean startEngine() {/* code */ }
    public abstract int getMaxSpeed();
    public abstract double getVehicleValue();
}
```

```java
public interface Rentable {
    public double getWeeklyRate();
}
```

```java
public interface UsesPremiumGas {
    public void fillTankWithPremiumGas(double gallons);
}
```

```java
public class Car extends Vehicle implements Rentable { /*code*/ }
```

```java
public class FancyCar extends Car implements UsesPremiumGas {
    /* code */
}
```

```java
public class Bus extends Vehicle { /* code */ }
```

a) Answer the following questions (must means so that the code compiles).
Which methods must be implemented in the Car class?

Which methods must be implemented in the FancyCar class?

Which methods must be implemented in the Bus class?
b) A company has a fleet of vehicles. Complete the method below that prints the cheapest weekly rate for the vehicles of the fleet that can be rented? In no vehicle can be rented, the method should state so.

Use an iterator in your implementation. You can assume that v is not null.

```java
/**
 * Prints the cheapest weekly rate available among a fleet of vehicles, or prints that no vehicle can be rented.
 * @param v the list of vehicles
 */
public void printCheapestWeeklyRate(List v) {
```
7) **Programming question [20 points]**

Consider a network of streets laid out in a rectangular grid; for example,

In a *northeast path* from one point in the grid to another, one may walk only to the north (up) and to the east (right). For example, there are four northeast paths from 1 to 2 in the preceding grid:

![Diagram of northeast paths](image)

Write a recursive method to count the number of northeast paths from one point \((x_1, y_1)\) to another point \((x_2, y_2)\) in a rectangular grid. Take that the number of northeast paths of a point to itself is 0.

```java
public int numberOfNEPaths(int x1, int y1, int x2, int y2) {
```