CMSC 113: Computer Science I
Practice Exam #1

During the actual exam, you will have access to whatever printed resources you like, but no electronic resources of any sort.

Part I. Each of the following programs draws a rectangle. For each problem, say where will its upper-left corner be with respect to the app window's upper-left hand corner (i.e., global coordinates). You may assume all the programs are correct and indeed draw a rectangle, and that all the necessary import statements have been included (but are left out of the test questions).

1.
```
public class Problem1 extends GraphicsProgram
{
	@Override
	public void run()
	{
		GRect rect = new GRect(70, 80, 90, 100);
		add(rect);
	}
}
```

2.
```
public class Problem2 extends GraphicsProgram
{
	@Override
	public void run()
	{
		GRect rect = new GRect(90, 60, 10, 10);
		rect.move(20, 20);
		add(rect);
	}
}
```

3.
```
public class Problem3 extends GraphicsProgram
{
	@Override
	public void run()
	{
		GRect rect = new GRect(30, 50, 10, 10);
		rect.setLocation(70, 100);
		add(rect);
	}
}
```

4.
```
public class Problem4 extends GraphicsProgram
{
	@Override
	public void run()
	{
		GRect rect = new GRect(20, 40, 10, 10);
		GOval oval = new GOval(80, 100, 10, 10);
		rect.setLocation(oval.getX(), rect.getY());
		add(rect);
	}
}
```

5.
```
public class Problem5 extends GraphicsProgram
{
	@Override
	public void run()
	{
		Problem5Object obj = new Problem5Object();
		obj.setLocation(30, 60);
		add(obj);
	}
}

public class Problem5Object extends GCompound
{
	public Problem5Object()
	{
		GRect rect = new GRect(0, 0, 10, 10);
		add(rect);
	}
}
```

6.
```
public class Problem6 extends GraphicsProgram
{
     @Override
     public void run()
     {
          Problem6Object obj = new Problem6Object();
          obj.setLocation(80, 120);
          add(obj);
     }
}

public class Problem6Object extends GCompound
{
     public Problem6Object()
     {
          GRect rect = new GRect(-10, -10, 20, 20);
          add(rect);
     }
}
```

7.
```
public class Problem7 extends GraphicsProgram
{
     @Override
     public void run()
     {
          Problem7Object obj = new Problem7Object();
          obj.setLocation(60, 80);
          obj.move(20, 20);
          add(obj);
     }
}

public class Problem7Object extends GCompound
{
     public Problem7Object()
     {
          GRect rect = new GRect(50, 50, 20, 20);
          add(rect);
     }
}
```

8.
```
public class Problem8 extends GraphicsProgram
{
	@Override
	public void run()
	{
		Problem8Object obj = new Problem8Object();
		obj.setLocation(40, 30);
		obj.frob();
		add(obj);
	}
}

public class Problem8Object extends GCompound
{
	private GRect rect;

	public Problem8Object()
	{
		rect = new GRect(10, 10, 10, 10);
		add(rect);
	}

	public void frob()
	{
		rect.setLocation(80, 90);
	}
}
```

9.
```
public class Problem9 extends GraphicsProgram
{
	@Override
	public void run()
	{
		Problem9Object obj = new Problem9Object();
		obj.setLocation(50, 90);
		obj.frob(10, 10);
		add(obj);
	}
}

public class Problem9Object extends GCompound
{
	private GRect rect;

	public Problem9Object()
	{
		rect = new GRect(20, 20, 10, 10);
		add(rect);
	}

	public void frob(int x, int y)
	{
		rect.move(x, y);
	}
}
```

10.
```
public class Problem10 extends GraphicsProgram
{
	@Override
	public void run()
	{
		Problem10Object obj = new Problem10Object();
		obj.setLocation(120, 110);
		obj.frob(20);
		add(obj);
	}
}

public class Problem10Object extends GCompound
{
	private GRect rect;

	public Problem10Object()
	{
		rect = new GRect(30, 50, 10, 10);
		add(rect);
	}

	public void frob(int z)
	{
		rect.move(z, 50 - z);
	}
}
```

Part II. Writing programs.

You will have to write several programs, and you will have a choice of which programs to write. Here are a few possible programs:

11. Write a program simulating a car crash. Draw the car as a rectangle, moving into the applet from the left side of the screen. Draw a wall as a line going down the center of the applet. When the front end of the car hits the wall (when the right side of the rectangle touches the line), the car immediately comes to a stop and does not move again.

12. Write a program to countdown to takeoff. The program starts with the text "Count: 5" displayed in the center. After every click, the count goes down. When the count reaches 0, the words "Blast off!" appear below the countdown. Further clicks do not affect the applet in any way.

13. Write an applet with a 100x100 box in the center. If the user clicks in the left half of this box, it grows bigger. If the user clicks in the right half of this box, it shrinks smaller. There is no limit to how big or small the box can get. The box must stay centered as it changes size.

14. Write a program showing a rocket taking off. Represent the rocket as an upward-pointing triangle (made of three lines – don't worry about polygons!) starting at the bottom of the applet. When you click once, the rocket starts moving up. It should *not* get faster every time you click.