CE/CS/SE 3354.004, Software Engineering, Fall 2017
Homework Assignment #2

Due: 11:59pm, November 3
Submission: *.doc (or *.pdf) via elearning
Maximum points: 100
The submitted file should contain only solutions, and should be named by the assignment, your id, and your first-last name, e.g., hw02-xxx123456-jon-bell.doc

I Comments and JavaDoc

Comment the method "division" in the follow code snippet. In the comments, explain what the method does and also use @param, @return, and @exception tags in the comments. Auto-generate JavaDoc for the class and paste the screenshot of Method Detail (the comments of "division") of the generated doc in your answer. (30 points)
Instruction: provide a screenshot similar to the bottom half of the picture in p34 of slide "08-coding-style".

```
public class Division {
	/**
	 *
	 *
	 */
	public static double division(double x, double y) {
		if(y == 0) { throw new IllegalArgumentException("y is 0"); }
		return x/y;
	}
}
```

II Refactoring: Extract Method, Pull Method, and Template Method

Refactor the following code snippets. Use "extract method" and "pull method" when appropriate. Template method pattern should be applied after the refactoring. Write the complete refactored code in your answer. (30 points)
Hint: start by extracting the steps of "printNameAndDetails" to two methods.

```
public abstract class Party {
}

public class Person extends Party {
	private String name;
	private Date dob;
	private String nation;

	public void printNameAndDetails(){
		System.out.println("Name: " + name);
		System.out.println ("Details: DOB-" + dob.toString() + ", Nation-" + nation);
	}
}

public class Company extends Party {
	private String name;
	private Date incorporated;

	public void printNameAndDetails(){
		System.out.println ("Name: " + name);
		System.out.println ("Details: Incorporated-" + incorporated.toString());
	}
}
```

III Refactoring: Motivation and Techniques

Connect (draw a line between) a motivation to a corresponding refactoring technique below. (25 points)

Motivation	Technique
Several methods perform similar actions that are different only in their internal values	Extract method
The methods of subclasses perform the same tasks	Replace magic number
Code fragment in a method can be grouped together	Extract class
You have repeated checks for a null value	Introduce null object
A method returns a special value to indicate an error	Pull up method
You have a class doing the work that should be done by two classes	Replace error code with exception
Numbers with special values are not obvious	Parameterize method

IV Testing vs. Formal Verification

What are the pros and cons of formal verification? Why is testing more widely used than verification? (15 points)