**44-542 Object Oriented Programming Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Exam 02 (100 points) Part 2** *please print*

1. (8 pts) Suppose we have a **Course** class that implements **Comparable<Course>**, with the **compareTo** method comparing courses on the basis of **courseNumber**. In the space below, write the complete code, including the method header, for the **compareTo** method of the **Course** class.

You may assume **courseNumber** is of type **String**.

Your code should fit in the space below. Do not write on the back.

1. (8 pts) Consider the code for class City below.

**public class City**

**{**

**private String cityName;**

**private ArrayList<Integer> rainAmt;**

**public City (String cityName)**

**{**

**this.cityName = cityName;**

**rainAmt = new ArrayList<Integer>();**

**}**

**public void addAmt (int amt)**

**{**

**if(amt < 0 || amt > 100)**

**{**

**throw new IllegalArgumentException(**

**"Invalid precipitation amount");**

**}**

**rainAmt.add(amt);**

**}**

**}**

In method main, assume we have the following code (you can assume **myCity** has been declared and initialized as a **City** object):

**myCity.addAmt(-10);**

Rewrite this code. Include the statement inside a try-catch block and catch the **IllegalArgumentException** when it is thrown. The catch block should print the message **Rain amount must be between 0 and 100**, followed, on a separate line, by the Java-generated error message.

Your code should fit in the space provided. Do not write on the back.

1. (6 pts) Make the necessary changes to the class Courses below, by adding the code for the iterator method and adding any other necessary code so that the result will compile and run.

***You do not have to include package or import statements.***

***Do not add any methods or private instance variables.***

***Your code should fit in the space provided. Do not write on the back.***

**public class Courses**

**{**

**private ArrayList<Course> courses;**

**public Courses()**

**{**

**courses = new ArrayList<Course>();**

**}**

**public void addCourse(Course course)**

**{**

**courses.add(course);**

**}**

**@Override**

**public Iterator<Course> iterator()**

**{**

**}**

**}**

1. (6 pts) Assume method **mystery** is defined as shown here:

**public static String mystery(String str1, String str2)**

**{**

**if(!str1.contains(",") && (!str2.contains(",")))**

**{**

**return "done";**

**} else**

**{**

**if(str1.indexOf(",") > 0)**

**{**

**str1 = str1.substring(str1.indexOf(",") + 1);**

**}**

**if(str2.indexOf(",") > 0)**

**{**

**str2 = str2.substring(str2.indexOf(",") + 1);**

**}**

**return str1 + " " + str2 + " " + mystery(str1, str2);**

**}**

**}**

What is the output of the following statement?

**System.out.println(mystery("SNOW,IS,FALLING", "GO,BEARCATS"));**

**OUTPUT**

1. (12 pts) Assume we have defined the classes and interfaces shown in the UML diagram below. Note that **Student** and **Employee** are interfaces, **AbstractStudent** is an abstract class, and all other classes are concrete.



Tell which of the following statements are legal and which are illegal. Circle the correct answer for each one.

**Student stu1 = new GraduateAssistant();** Legal Illegal

**Student stu2 = new Undergraduate();** Legal Illegal

**Employee emp1 = new GraduateAssistant();** Legal Illegal

**Employee emp2 = new Graduate();** Legal Illegal

**AbstractStudent stu3 = new Undergraduate();** Legal Illegal

**Graduate GA2 = new GraduateAssistant();** Legal Illegal

1. (4 pts) Using the same UML diagram as for the previous problem, consider the following code segment.

**Student stu = new Graduate();**

**Undergraduate undergrad = new Undergraduate();**

**undergrad = (Undergraduate) stu;**

Which of the following statements is true about the code segment above? Circle the correct answer – be sure to circle only one choice; if you circle more than one choice, you answer will be considered incorrect.

The code segment will not compile.

The code segment will compile, but there will be a runtime error.

The code segment will compile and run.

1. (6 pts) Find the output of the following code:

**int num1 = 90;**

**int num2 = 80;**

Output

**int num3 = 85;**

**int num4 = 3;**

**while(num1 + num2 > 50)**

**{**

**num3 = num3 / 2;**

**if(num1 + num2 > 100)**

**{**

**num1 -= 3;**

**num2 -= 5;**

**}**

**for(int i = 1; i < num4; i++)**

**{**

**num1 -= 10;**

**num2 -= 10;**

**}**

**System.out.println(num1 + " " + num2 + " " + num3);**

**}**