Enumerated Types – 01/29/08

Team Name: _____Members Present:

This will be a graded assignment. Be sure that you work to get the answers right. If explanation is required, make sure the answer is complete, concise and correct. Assign a team leader for today, and designate one person to record answers. Feel free to ask questions of other teams or the instructor if you are confused about something.

Task 1:

1. In your group, discuss the lab from Monday. Specifically answer these questions:
    a. How is building a Scanner over a file different from building a Scanner over standard input (System.in)?
    b. How is using the Scanner once it is built the same or different from using the Scanner over standard input?
    c. What tools of the File class will help you to prevent the FileNotFoundException and other IOExceptions from occurring?
    d. If you build the logic to prevent the errors, do you still have to deal with the IOExceptions that may be thrown? Why? Specifically, what do we call exceptions like IOException?
    e. When you write to a file, how is this process the same as writing to standard output (System.out)? How is it different?
    f. Write detailed pseudo code that will prompt for a file name, check to see if that file exists on disk and either provides an error message to the user or opens the file for writing.

Task2:

1. In the Gaddis book, look at the example on page 564, EnumDemo.java:
    a. What is the name of the enum data type?
    b. What are the allowable values?
    c. Which value is the lowest possible value? Highest?
    d. What method lets us compare enumerated type values?
    e. Write a statement that declares choreDay as a Day type.
    f. Write a statement that assigns choreDay the value for Saturday.
    g. Write an expression that returns true if choreDay is greater than workday and false otherwise.
    h. Think of a way that you might use the ordinal method of the enum type.
2. Extending enums
    a. Look at the two examples of enumerated types in their own file, on page 565.

b. What do you think will be produced when these files compile successfully?

c. Notice that there is no visibility modifier on the enum types. What do you think the visibility is?

3. Referring to SportsCar.java, answer the following:

   a. If I create a new SportsCar using the statement,

      SportsCar myCar = new SportsCar(CarType.JAGUAR, CarColor.BLUE, 50000);

      what value is returned by myCar.getMake()?

   b. What value is returned by myCar.getColor()?

4. Looking at page 568, why can we use an enum type in a switch statement? Think about the kinds of values we can use in a switch and then compare how enum types are similar to those values.

5. Do the checkpoint exercises that you find on page 569/570.

6. What do we know about classes?

   a. Classes can have _____ and _____.

   b. If we say that enumerated types are a special form of a class, that implies that these classes (and their objects have _____ and _____.

   c. Give an example of a method in the enum type, CarType. (Look at Day for some examples).

   d. Enum types share several methods regardless of the type. They include ordinal(), name(), toString(), values(). Would it be useful to define our own methods for an enum type?

   e. Think about the months that we used in the PA1. What methods may have been useful if we wanted to define a series of named constants, one for each month? Or in other words, given the use that we put the months to, what would an enum class have to contain to help us?

7. Enum types can have some "intelligence" built into them. We can define data beyond just the name of the type. Look at the Planets example from the Java Tutorial which you will find on the last page.

   a. How many Planet objects are defined by this class? _____

   b. What are they? _____

   c. What data do these objects contain? _____

   d. Does this class contain a constructor (think about what defines a constructor).

   e. Can this constructor be called by outside classes to build their own Planet? Why / Why not?

   f. Which Planet has the lowest value? The highest?

   g. What methods does the Planet class offer to users of the class?

   h. In words, describe what the main method does?

i. Specifically, what do you think the role of the for loop is? Think about what the enhanced for loop does?

j. Write a comparable loop to print out the Days of the week (from the Days class).

```
public enum Planet
{
    MERCURY (3.303e+23, 2.4397e6),
    VENUS   (4.869e+24, 6.0518e6),
    EARTH   (5.976e+24, 6.37814e6),
    MARS    (6.421e+23, 3.3972e6),
    JUPITER (1.9e+27,   7.1492e7),
    SATURN  (5.688e+26, 6.0268e7),
    URANUS  (8.686e+25, 2.5559e7),
    NEPTUNE (1.024e+26, 2.4746e7);

    private final double mass;   // in kilograms
    private final double radius; // in meters

    Planet(double mass, double radius)
   {
        this.mass = mass;
        this.radius = radius;
    }
    private double mass()   { return mass; }
    private double radius() { return radius; }

    // universal gravitational constant  (m3 kg-1 s-2)
    public static final double G = 6.67300E-11;

    double surfaceGravity() {
        return G * mass / (radius * radius);
    }
    double surfaceWeight(double otherMass) {
        return otherMass * surfaceGravity();
    }
    public static void main(String[] args) {
        double earthWeight = Double.parseDouble(args[0]);
        double mass = earthWeight/EARTH.surfaceGravity();
        for (Planet p : Planet.values())
           System.out.printf("Your weight on %s is %f%n",
                             p, p.surfaceWeight(mass));
    }
}
```