CC3 – Object Oriented Programming

Laboratory Exercise #4
Classes, Objects and Methods

Name: _____ Date: _____

Code/Schedule: _____ Terminal #: _____

Topic(s) Covered: Methods, Class, Object

Estimated Completion Time: 1-2 meetings

Objectives:

1. To write a class that contains instance variables, accessor and mutator methods.
2. To generate random numbers using the Random class from the utilities package.
3. To implement user-defined methods.
4. To instantiate two objects that will utilize declared variables and use the methods declared in the class.

Activity:

1. Create a new project named RPG.

2. Complete the Dice.java template (see page 2 of this file) given to model rolling a regular six-sided die. You only need to complete the Dice() constructor and roll() method. To generate random numbers, use the imported Random class. Random has a method called nextInt(int), which takes an integer as its parameter, and returns a random integer between 0 (inclusive) and the given parameter (exclusive). For example:
r.nextInt(100);
returns a number from 0-99

3. Complete Character.java code template (see page 3 of this file), which models a character for a roleplaying game. You will need to add the 6 instance variables described in the documentation, as well as the code for all of the given methods. Some of the implementation (such as the attack() method) can be done in many different ways and is open to your interpretation.

4. Write a TestCharacter.java class, which creates two characters who alternate attacking each other, until one character's current life is less than or equal to zero. Print the results of each attack and the eventual winner.

Sample output for the class TestCharacter:
Round 1
Joe: 24
Bob: 23
Joe attacks Bob for 6
Bob attacks Joe for 11
Round 2
Joe: 13
Bob: 17
Joe attacks Bob for 5
Bob attacks Joe for 9
Round 3
Joe: 4
Bob: 12
Joe attacks Bob for 9
Bob attacks Joe for 8
Bob wins!

Dice. java

```
/**
* The Dice class models rolling a regular six-sided die
*/

import java.util.Random;

public class Dice {
      /** instance variables */
      private Random r;

      /**
       * Instantiate the object r
       */
      public Dice() {
      }

      /**
       * Returns a random integer between 1 and 6
       */
      public int roll() {
      }
}
```

Character. java

```
/**
* The Character class models a character for a roleplaying game.
* A character has a name and 3 stats: strength, dexterity, and
intelligence, as
* well as maximum allowed life points and the current value.
*/

public class Character {
      /** static variables */
      static Dice dice = new Dice();

      /** instance variables */

      /**
       * Constructor takes 4 parameters: n,s,d,i and assigns them to
       * instance variables name, strength, dexterity, intelligence.
       * Also uses the dice instance to assign a random
       * number to maxlife. currentlife is initialized to maxlife
       */
      public Character(String n, int s, int d, int i) {
      }

      /**
       * Returns a random die roll using the roll method in the
       * Dice.java, *modified by the character's strength
       */
      public int attack() {
      }
      /**
       * Decreases currentlife by the damage parameter
       */
      public void wound(int damage) {
      }
      /**
       * Increases currentlife by the heal parameter.
       * currentlife cannot be greater than maxlife
       */
      public void heal(int heal) {
      }
      /**
       * Returns name
       */
      public String getName() {
      }
      /**
       * Returns strength
       */
      public int getStrength() {
      }
      /**
       * Returns dexterity
       */
      public int getDexterity() {
      }
```

```
        /**
         * Returns intelligence
         */
        public int getIntelligence() {
        }
        /**
         * Returns currentlife
         */
        public int getCurrentLife() {
        }
        /**
         * Returns maxlife
         */
        public int getMaxLife() {
        }
}
```

Laboratory Exercise Score Sheet		
Criteria		Score
1 Conformance to documentation requirements	15	
2 Proper use of program structure	15	
3 Program produces desired output	15	
4 Program is submitted on schedule	10	