CSCI 2320
Spring 2018
Exam 2

Last Name: _____

First Name: _____

Unless specified otherwise, write your answers on the provided answer sheets. You may refer to printouts of any source code you have written.

1. (3 pts each) Briefly define each of the following terms in the context of this course:
   (a) overriding
   (b) polymorphism
   (c) indirect recursion

2. (4 pts) Explain how the terms "is-a" and "has-a" are helpful in determining how to define relationships between various objects in an object-oriented program.

3. Consider the following problem (adapted from codingbat.com):

   > Given a non-negative int n, return the count of the occurrences of 7 as a digit, so for example 717 yields 2. (no loops). Note that mod (%) by 10 yields the rightmost digit (126 % 10 is 6), while divide (/) by 10 removes the rightmost digit (126 / 10 is 12).

   (a) (2 pts) Using words provide a recursive description of this problem. You might start with something like: "The number of sevens in a number is ...". Be sure to identify a base condition as well.
   (b) (6 pts) Write a recursive function called `numSevens` that takes a single integer parameter (assumed to be non-negative) and returns the number of 7 digits that appears in the number.

4. (6 pts) Consider the following recursive function. What gets output by `fun(-831)`? REMEMBER: an integer divided by an integer results in an integer (e.g., $7/2 = 3$).

```
public static void fun(int n) {
   if (n < 0) {
      fun(-n);
   }
   else if (n < 10) {
      System.out.println(n);
   }
   else {
      fun(n/10);
      System.out.println(n%10+1);
   }
}
```

5. Suppose we continue from the previous exam with our concept of a representing a stocks using the following class:

```
class Stock
{
   protected String ticker;        // ticker symbol for the stock
   protected int sharesOwned;      // number of shares owned
   protected double pricePerShare; // price of 1 share of this stock
                                   // at time of purchase

   public Stock(String ticker, int shares, double price) {
      this.ticker= ticker;
      sharesOwned= shares;
      pricePerShare= price;
   }

   public String toString() {
      return ticker+" "+sharesOwned+" "+pricePerShare;
   }

   public double sellShares(int numSold) {
      if (numSold <= 0 || numSold > numShares)
         return 0.0;
      sharesOwned-= numSold;
      return numSold*pricePerShare;
   }
}
```

Imagine that this class has been used in a variety of programs to represent stocks and it is working well. One concern is that this class does not support the concept of stock dividends. Sometimes when a company does well they pass along some of their profits to shareholders in the form of dividends. For example, if a person owns 100 shares of a stock and the company pays dividends of $0.10 per share then that person would be paid $10.00. This is different than getting paid for selling shares because the shareholder still owns 100 shares of the stock. To represent the concept of dividends we will keep the original class untouched and instead will inherit from it to create a new class: `DividendStock`.

(a) (4 pts) Create a new class called `DividendStock` to represent a stock that pays annual dividends. This new class should inherit from `Stock` and should add an double attribute called `dividendPerShare` that identify the amount the company will pay in dividends per share.

(b) (4 pts) In the new `DividendStock` class create a constructor that accepts four parameters (name of stock, purchase price, number of shares, and dividend paid per share) that are used to initialize the attributes of the class. The constructor should, as one of its steps, invoke the constructor of its parent class.

(c) (6 pts) Draw a UML diagram depicting these two classes. When depicting attributes you should specify their type. When depicting methods you should specify their return type, name, and parameters. Do not include constructors or `toString()` in the diagram.

(d) Write a section of code that makes use of the `Stock` and `DividendStock` classes as follows:

i. (4 pts) Establish an array (called `cust`) of 10 references to stocks.

ii. (4 pts) The first element of the array should refer to a "regular" stock (named ACME) in which we own 20 shares that cost $54.00 per share.

iii. (2 pts) The second element of the array should refer to a stock named RR for which we own 50 shares at $32.00 per share. We'll assume that RR pays an annual dividend of $0.10 per share.

iv. (4 pts) Write (a) statement(s) that would accomplish selling of 5 shares of RR.

(e) (2 pts) In our current design would it make sense to declare the `Stock` class as abstract or not? Briefly explain.

6. Suppose we want to represent a 30×20 ASCII art image as a 2d array of characters. Further assume that our images utilize only four colors as follows: a space represent white, a period represents yellow, a capital 'B' represents brown, a capital 'X' represents black.

(a) (2 pts) Create a class called `ASCIIArt` that contains an attribute called `pad` that provides a 2-dimensional array representation of the art image described above.

(b) (4 pts) In the constructor reserve memory for the `pad` attribute.

(c) (4 pts) Also, in the constructor write code that will initialize the entire image to be "white".

(d) (6 pts) Write a method called `draw()` that accepts three parameters: a row number, a column number, and a color as parameters. If the row and column positions are outside the pad then the function should end without doing anything. Likewise, if the color we are asked to draw is not one of the four colors we allow then we don't want to do anything. If the parameters are okay then you should draw the specified color in the specified location on the pad.

(e) (6 pts) Write a method called `isDark()` that will return `true` if the image currently on the pad is considered to be dark. Otherwise return `false`. The colors brown and black are considered dark and the colors white an yellow are considered light. If there are more dark colors than light colors on the pad then the image is considered to be dark.

7. (4 pts) Write/show something you know from this course that did not get asked on this exam.