[all multiple choice 3pts]

1— The name of the type of method that creates objects of a class is
   a. instantiator
   b. creator
   c. constructor
   d. declarator

2— If you wanted to store a student’s letter grade in a data type, which of these would be best?
   a. int
   b. float
   c. char
   d. enum

3— Which methods can access the private variables of a class?
   a. Only static methods of the same class
   b. Only methods defined in the same class
   c. Only methods defined in the same package
   d. Only methods within inner classes

4— Which of the following is the purpose of a try-catch statement?
   a. Determines if a method exists in a class
   b. Throws an exception so that the caller knows there was an error
   c. Handles exceptions that were thrown by methods
   d. To be used as an alternative to if-else statements

5— Which code would you use to instantiate a new ArrayList that could ONLY hold Apples?
   a. ... = new ArrayList<Apple>();
   b. ... = <Apple>ArrayList();
   c. ... = new ArrayList<Apple>;
   d. ... = ArrayList<Apple> a;

6— The following code compiles and runs, so which of the following must be true?
   apple.isItRipe( friday );
   a. friday must be an int or an enum
   b. apple must be the name of a class
   c. apple must be a fruit
   d. isItRipe must be a method
7— Consider the following code segment, which will be used to time a function. It should print out the number of *seconds* that have elapsed so far each time magicFunction finishes. And then return the total time for all of magicFunction’s runs. [8pts]

```java
long timerMethod( int times ) {
    long time1;
    long time2;

    time1 = System.currentTimeMillis();

    for( int i=0; i<times; i++ ) {
        magicFunction( i );
        System.currentTimeMillis()
        System.out.println( time2 = time2/1000 );
    }

    time1 = System.currentTimeMillis();
    return long = time2 - time1;
}
```

There are errors with this code. Can you fix them? (Don’t just say “Yes!” actually fix them)

8— Why do the following lines of code not compile? (Explain in one sentence each) [6pts]

```java
ArrayList<boolean> testAnswers;

try { System.out.println( “Hi” ); } 
catch new NullPointerException;
```
9— Write a function called `wordsWithoutList`, which given an `ArrayList` of strings, returns an `ArrayList` where all the strings of the given length are omitted. (Note these `ArrayList`s are *not* using generics) [12pts]

```java
wordsWithoutList("
    "a", 
    "bb", 
    "b", 
    "ccc"}, 1) → 
    
    wordsWithoutList({
    "a", 
    "bb", 
    "b", 
    "ccc"}, 3) → 
    
    wordsWithoutList({
    "a", 
    "bb", 
    "b", 
    "ccc"}, 4) → 

ArrayList wordsWithoutList(ArrayList words, int len) {
```
11 — Recursion #1 [10pts]
Write a method that given a base and an exponent (both 1 or more) returns the base to the exponent power. so powerN(3, 2) is $3^2 = 9$. Do not use loops. Do not use Math.pow. Write it recursively!

```java
int powerN(int base, int exp) {
```

12 — Traversing a tree. [9pts]
For the tree on the right write out the following traversals:

Breadth-first:

___ ___ ___ ___ ___ ___ ___ ___ ___

Depth-first, In-order:

___ ___ ___ ___ ___ ___ ___ ___ ___

Depth-first, Pre-order:

___ ___ ___ ___ ___ ___ ___ ___ ___
13— Operations on a tree. [14pts]
Given a binary tree composed of nodes defined as follows:

```java
class Node {
    public String data;
    public Node left = null;
    public Node right = null;
}
```

Write a **method** that given a node, returns a String containing an in-order concatenation of all its children (and itself). For example running this method on the root of the tree on the right would return the String “FINCH”
14— Generics [9pts]
Using a Basket class, explain what happens when we try to run each of the following code snippets.

```java
public class Basket<E> {
    private E element;

    public void setElement(E x) {
        element = x;
    }

    public E getElement() {
        return element;
    }
}

class Food { }
class Cheese extends Food { }
class Brie extends Cheese { }
class Wine extends Food { }

--
14— Run 1
Basket<Food> picnic = new Basket<Food>();   // 1
picnic.setElement(new Cheese());            // 2
Cheese cheese = picnic.getElement();        // 3
What happens?

14— Run 2
Basket<Food> picnic = new Basket<Food>();   // 1
picnic.setElement(new Cheese());            // 2
Wine wine = (Wine)picnic.getElement();      // 3
What happens?

14— Run 3
Basket<Food> picnic = new Basket<Food>();   // 1
picnic.setElement(new Cheese());            // 2
picnic.setElement(new Brie());              // 3
Brie brie = (Brie)picnic.getElement();      // 4
What happens?
```
Recursion #2 [10pts]

Given a string, compute recursively a new string where all the lowercase 'x' chars have been moved to the end of the string.

```
endX("rexx") → "rexx"
endX("xxre") → "rexx"
endX("xxhixx") → "hixxxx"
endX("xhixhix") → "hihixxx"
```

```
public String endX(String str) {
```