Graded Exercise 1

CSC144 Software Architecture

30 March 2018

1. The GraphicsContext class has methods that can be used to draw pictures. You used the methods of this class to draw your landscape.

   Suppose that you have an instance of the GraphicsContext class named gc.

   (a) Show a call to a method of the GraphicsContext class that will alter some drawing attribute (for example, the color or width of lines).

   (Your answer need not be perfect.)

   (b) Show a call to a method of the GraphicsContext class that draws something on the screen.

(a) gc.setStroke(Color.BLACK);

(b)
```
double[] xCoordinates = new double[3];
double[] yCoordinates = new double[3];

xCoordinates[0] = 0.0;
xCoordinates[1] = 100.0;
xCoordinates[2] = 100.0;

yCoordinates[0] = 0.0;
yCoordinates[1] = 0.0;
yCoordinates[2] = 100.0;

gc.strokePolygon(xCoordinates, yCoordinates, 3);
```

2. A Java class may extend another class or it may implement an interface.

   An interface, like a class, lists the names of methods, the types of values that those methods return to their callers, and the numbers and types of parameters that those methods require. Unlike a class, an interface does not define how a method will produce its result.

   Why did the designers of the Java programming language choose to make a distinction between classes and interfaces?

   If Java allowed a class to extend two classes, it might happen that the class would inherit methods from both classes that had the same name and same number and types of parameters. These methods might work in different ways and produce different results. The interpreter could not know which version to execute.

   If a class implements two interfaces, there is no conflict, because, although both interfaces might specify methods with the same name, neither defines the method—the relationship between class and interface requires the author of the class to give the method a single, unambiguous definition.

3. In each of the classes that we wrote we included a piece that looked like a method in most ways. It differed from the methods by the fact that it lacked a return type and a return statement. It also differed from the methods by the fact that its name matched the name of the class.

   What do we call this part of a class?

   It is a constructor.

4. Encapsulation is a key goal of object-oriented programming. Support for encapsulation is a key feature of object-oriented languages.

   In the Java programming language, this support takes the form of instance variables, accessor (getter) methods, mutator (setter) methods, and **private** and **public** qualifiers.

   Why is encapsulation a good idea?

Encapsulation hides the details of how a class organizes, combines, and changes data. These allows members of a software engineering team to each focus their efforts on their own parts of a project. They are relieved from the need to know all of the details of their teammates' contributions. Encapsulation prevents a person who is responsible for one part of a program from writing code that accidentally makes changes in another part. Encapsulation makes programming easier and safer.

5. What is a stub method? Why might you write a stub method?

A stub method defines the name of the method, the type of value it returns to its caller, and the number and types of its parameters. The body of stub method contains only a **return** statement. It returns an arbitrary value (for example, 0, 1, **true**, or **false**). This will most often be an incorrect result. However, with the stub method in place, the program will compile and run, and the programmer can work on other parts of the program until the programmer finds the knowledge needed to complete the definition of the method.

6. What is the value of regression testing?

Software engineers continue running tests even after the code passes the tests. They do this because they understand that defects, once repaired, can reappear. If one programmer misunderstood specifications, made an unjustified assumption about how clients will use the code, or got the logic or arithmetic wrong, another programmer might repeat the mistake at a later time. Code evolves, many people touch the code, and changes in one part of a program can affect other, distant parts of the program. Regression testing—repeated, continued testing—protects against the reintroduction of errors.

7. The availability of components makes the design and manufacture of many kinds of products easier. Engineers do not have to design and build every part of product when they have the option of using components that someone else has already designed, built, and tested.

   What is the form of the components that software engineers create and share? (That is, what is a word that describes a component of a software program?)

A class is the must fundamental component of an object-oriented program. It is a shareable and reusable component.

8. Inheritance is another key concept in the object-oriented discipline. How does the use of inheritance enable a programmer to solve a problem with less work than might otherwise be necessary?

When one class extends another class, it inherits methods and variables from the parent class. The author of the class that inherits does not have to repeat the definitions of these elements.

Inheritance is an "is a" (or "is a kind of") relationship. The class that inherits models a special kind of the thing that the parent class models. The author of the subclass has only to define that features that distinguish the subclass from the parent class.

9. Here is a function that finds the smallest integer in an array of integers.

```
int minimum( int [] data ) {
    int bestGuessSoFar = data[0];
    for( int i = 1; i < data.length; i++ ) {
        if( data[i] < bestGuessSoFar ) {
            bestGuessSoFar = data[i];
        } // if
    } // for
    return bestGuessSoFar;
} // minimum( int [] )
```

A function that finds the position of the smallest integer in an array of integers is very similar. It differs in only three places.

```
int positionOfMinimum( int [] data ) {
    int bestGuessSoFar = ????;
    for( int i = 1; i < data.length; i++ ) {
        if( data[i] < ???? ) {
            bestGuessSoFar = ????;
        } // if
    } // for
    return bestGuessSoFar;
} // positionOfMinimum( int [] )
```

(a) This function will return an index. It begins by guessing that index is equal to what?

(This replaces the first set of question marks.)

(b) This function goes on to compare the values at every position in the array with the value at the position where it has guessed that the smallest value is.

In the **if** statement, it compares data[i] with what?

(This replaces the second set of question marks.)

(c) When the function finds the position of a smaller integer it updates its guess of the position of the smallest element of the array.

What belongs on the right-hand side of the assignment statement within the **if** statement?

(This replaces the third set of question marks.)

(a) **int** bestGuessSoFar = 0;

(b) **if**(data[i] < data[bestGuessSoFar]) {

(c) bestGuessSoFar = i;

10. A logarithm appears in the expressions that describe how much work must be done in a binary search or in sorting a list with the quicksort or merge sort algorithms.

    In your high school algebra class, you learned that a logarithm is the power to which we have to raise one number to get a given value. For example, the logarithm base 2 of 16 is 4 because $2^4 = 16$.

    A different but equivalent definition of logarithm explains why logarithms show up in the analysis of divide-and-conquer algorithms like binary search, quicksort, and merge sort.

Here is an application of that different definition.

$$512/2 = 256 \quad (1)$$
$$256/2 = 128 \quad (2)$$
$$128/2 = 64 \quad (3)$$
$$64/2 = 32 \quad (4)$$
$$32/2 = 16 \quad (5)$$
$$16/2 = 8 \quad (6)$$
$$8/2 = 4 \quad (7)$$
$$4/2 = 2 \quad (8)$$
$$2/2 = 1 \quad (9)$$

$$\log_2 512 = 9$$

Put this definition into words: "The logarithm base 2 of a positive power of 2 is..."

The logarithm base 2 of a positive power of 2 is the number of times we have to divide the number by 2 before reaching 1.

A divide by conquer algorithm cuts a problem into half again and again, until it gets down to a trivial problem. It expresses the solution of big problems as the solution of a smaller problems.

11. The selection sort algorithm is $O(N^2)$. How much longer would you expect to wait for a program that uses this algorithm to put two million values into order, as compared to the time required to sort one million objects?

Two million is twice one million. A $O(N^2)$ algorithm takes four times as long to produce an answer when the size of the input is doubled.

12. Here are the first few steps of a selection sort. What is the next line in this table?

16	78	82	61	21	96	55	42	05	40	49	44
05	78	82	61	21	96	55	42	16	40	49	44
05	**16**	82	61	21	96	55	42	78	40	49	44
05	**16**	**21**	61	82	96	55	42	78	40	49	44
05	**16**	**21**	**40**	82	96	55	42	78	61	49	44
05	**16**	**21**	**40**	**42**	96	55	82	78	61	49	44

05	**16**	**21**	**40**	**42**	**44**	55	82	78	61	49	96