Answer the discussion questions in writing.

1. What is a class? According to the book, what does a class “arrange?”

A class systematically arranges information and behavior into a meaningful entity. For example, all cars share common behaviors (stop, accelerate) and common attributes (wheels, engine). People use the “class” car to mean an object that shares these common behaviors and attributes.

1. What are the two purposes of encapsulation?

1) To combine methods and data within a class; to support classification.  
2) To control the accessibility of the methods and data; to control the use of the class.

1. How do you instantiate an instance of a class? How do you access that instance?

To instantiate an instance of a class, you first create a new variable using the class like you would use a type to declare a new variable. You then initialize the variable with the ‘new’ keyword, which tells the runtime to construct that object by using the definition of the class. You can then access that instance by using the variable name followed by a dot and then the field or method you want to access. For example.

Circle c; // Create a Circle variable  
c = new Circle(); // Initialize it  
double area = c.Area() // Access the Area method

1. What is the default access of the fields and methods of a class? How do you change the default?

Private is the default. You change the default by writing the keyword “public” before the declaration.

1. What is the syntax for writing a constructor ?

The constructor must match the class name, and can have no parameters (default) or multiple parameters. For example:

Class Circle  
public Circle() // default constructor

1. What is the difference between class fields and methods, and instance fields ad methods? How do you create class fields and methods?

The class fields and methods typically contain computation instructions and variables that are accessed by instance methods to pass the results back to the instance fields and methods. You can create private classes and fields that are only accessed within the class, or you can create public classes and fields which can be accessed from outside the class. For example:

private int radius; // creates a private field  
public double Area() // creates a public method

1. How do you bring a static class in scope? Why would you want to bring a static class in scope?

You bring a static class into scope with the ‘using’ statement. You might want to do this so that you don’t need to fully qualify the classes with their namespaces. For example, instead of calling “Math.Sqrt” every time you need it, you can include “using static System.Math” to bring the Math class into scope, and then you only need to call “Sqrt.”

1. Can you think of a good reason to create an anonymous class? What is it?

Using query expressions is a good reason to create an anonymous class. It is a class that does not have a name. For example:

var myAnonObject = new { Name = “John” , Age = 47 };

1. What is polymorphism as this term is used in computer science? This is not in the book.

In object-oriented programming, polymorphism refers to a programming language's ability to process objects differently depending on their data type or class. More specifically, it is the ability to redefine methods for derived classes. For example, given a base class shape, polymorphism enables the programmer to define different area methods for any number of derived classes, such as circles, rectangles and triangles. No matter what shape an object is, applying the area method to it will return the correct results. Polymorphism is considered to be a requirement of any true object-oriented programming language (OOPL).

1. What is message passing as this term is used in computer science? This is not in the book.

Message passing, in computer terms, refers to the sending of a message to a process which can be an object, parallel process, subroutine, function or thread. This message can be used to invoke another process, directly or indirectly. Message passing is especially useful in object-oriented programming and parallel programming when a single message (in the form of a signal, data packet or function) is sent to a recipient.

1. What was the first object-oriented programming language?

Simula (1967) is considered the first OOP language.

1. Consider this quote by Alexander Stepanov:

I find OOP technically unsound. It attempts to decompose the world in terms of interfaces  
that vary on a single type. To deal with the real problems you need multisorted algebras  
— families of interfaces that span multiple types. I find OOP philosophically unsound. It  
claims that everything is an object. Even if it is true it is not very interesting — saying that  
everything is an object is saying nothing at all.

Who is Alexander Stephanov? What do you think about this quote?

Alexander Stephanov is the primary designer of C++ Standard Template Library. He is an advocate of generic programming over OOP. The advantage of generic programming is that you don’t need to create multiple classes with overloaded parameters. Instead, you create one class and the type of parameter determines which instance to create. The compiler infers the types of the fields from the types of the data you specify to initialize them. In C# this is the function of anonymous classes.

The quote is a little extreme. I think that generic programming is more flexible and functions more like how the brain works. For example, if I have a class called ‘car,’ then according to OOP to access this car I can only use specified parameters like BMW or Tesla to instantiate the object. However, if I want to buy a toy car for my child, I am not inputting BMW into my mental parameter. Instead, I am thinking (inputting) Tyco toy cars which would require a completely separate class in OOP but could be accessed in generic programming to instantiate an item like a car that is made of different types of materials from BMW.