**Oriette Ramos**

**Lesson Plan 29**

**ISTA-220**

1. What is a delegate? Explain delegates in terms of pointers and reference types.  
   **A delegate is an object that refers to a method. It is basically creating a function that will run a specified method or methods.**
2. How do you declare a delegate? Include a discussion of types, return values, names, and parameters.  
   **To declare a delegate you use the delegate keyword and specify the return type, a name for the delegate, and any parameters. It doesn't have a body, so it ends with a semicolon after the parenthesis. It basically looks like a normal method declaration, but with the access modifier replaced with the delegate keyword. The name of the delegate is a new type.**
3. How do you create instances of delegates and assign values to the instance? What are the values?  
   **You create an instance of a delegate the same way you create an instance of other types: you optionally specify an access modifier, you then specify the type (the delegate name), that is followed by a variable name, and you can either stop with a semicolon and assign values later, or include value assignment. The values are methods without parenthesis or parameters. You also have the option to use the new keyword to initialize a delegate explicitly with a single specific method.**
4. How do you invoke a method that has been added to a delegate?  
   **You use the same syntax to invoke a delegate as you use to call a method. If the method that the delegate refers to takes any parameters, you should specify them at this time between the parentheses.**
5. What is an event? Why do we have events?  
   **Events define and trap significant actions. An event source is usually a class that monitors its environment and raises an event when something significant happens. We have events to run code automatically when something significant happens.**
6. How do you declare events?  
   **You declare an event similarly to how you declare a field. However, because events are intended to be used with delegates, the type of an event must be a delegate, and you must prefix the declaration with the event keyword.**
7. How do delegates recognize event subscriptions? How do you unsubscribe an event from a delegate?  
   **You subscribe to an event by using the += operator. Calling the –= operator removes the method from the event’s internal delegate collection.**
8. How do you raise an event? How do you declare code that raises an event?  
   **You can raise an event by calling it like a method. When you raise an event, all the attached delegates are called in sequence. When declaring the code that raises an event, it is common to check for a null event. If you try to raise a null event, you will get a NullReferenceException exception.**
9. Explain with specificity what happens when an event fires and that event has been attached to a delegate.  
   **When an event fires, the methods that are subscribed are called in sequence. It functions exactly like calling a delegate, but with the protection that it cannot be called from outside the class and the subscribed methods must be unsubscribed individually (they can't all be overwritten by one method using the assignment operator like a delegate).**