C# Programming Lesson Plan 17 Chapter 14

Answer the discussion questions in writing.

1. What is the difference between a managed resource and an unmanaged resource?

Managed resources are those that are pure .NET code and managed by the runtime and are under its direct control.

Unmanaged resources are those that are not. File handles, pinned memory, COM objects, database connections etc.

2. When is memory for an object (reference type) allocated? When is the memory deallocated?

The memory for an object is allocated when the new keyword is used to create a new instance of that object. The Memory is deallocated when all references to that object have disappeared.

3. What is a destructor?

A destructor is used in conjunction with the automatic Garbage Collector to free/release resources and free up memory from the heap.

4. What is the difference in syntax between a constructor and a destructor?

The destructor does not specify an access modifier, does not take any parameters.

5. Give some examples of scarce resources. Why would you want to manage scarce resources?

Examples of scarce resources are: Memory, file handlers and database connections. You would want to manage the scarce resources because they are too valuable to wait on random garbage collection. They need immediate attention.

6. What is exception-safe disposal?

Exception-safe disposal is to call the disposal in a finally block to ensure that a disposal method is always called.

7. How do you think that the using statement works for resource management? Give an informal, English language, explanation of how it works.

The using statement introduces its own block and the object reference goes out of scope at the end of the using block, making it available to release its resources. You can apply the using statement in conjunction with scarce resources that need to be managed in a time expedient manner.

8. What ill effects could result from attempting to dispose of a resource more than once?

If you attempt to dispose of a resource more than once it can cause logical inconsistencies in you application.

9. We will look at threads later in the term. For now, what is your understanding of how threads interact with resource management? A good guess is a sufficient answer to this question.

Threads interact with resource management through the use of a lock statement.

10. Why does the book recommend not attempting to force the garbage collector? Are there any exceptions to this recommendation?

The book recommends not attempting to force the garbage collector but offers the GC.Collect method that can be called if you need to invoke the collector; however it is not recommended because it runs asynchronously the G.C. Collect does not wait for the garbage collector to complete to be complete before it returns, so you don’t know for sure that your objects were destroyed.