**C# Notes**

*Abstract classes vs Interfaces:*

Abstract classes can have implementations for some of its members, but interfaces CAN’T have implementations for any of their members.

Interfaces CAN’T have fields, but abstract classes can.

An interface can inherit from another interface, but CAN’T inherit from an abstract class. An abstract class can inherit from another abstract class (or an interface).

Any class can inherit from multiple interfaces at the same time, but a class cannot inherit from multiple classes at the same time.

Abstract class members can have access modifiers whereas interface members CAN’T.

*Delegates:*

Must have the same return type and parameters as the function it is pointing to (hence they are “type safe”).

A delegate is similar to a class. You can create an instance of it and then pass the function name in as a parameter, which the delegate will then point to.

When creating a multicast delegate, the methods will be invoked in the SAME ORDER they were added.

*Exceptions:*

Attempt to catch specific (and predicted) exceptions higher up the chain, and defer lower exceptions downwards.

Close resources/IO streams in the finally block so that an exception does not prevent resources being freed up in your program/necessary tasks aren’t skipped.

Modified