The artifact for programming two is an example of object-oriented programming. It was a project where we kept track of the two different types of customers, the wholesale customers and the retail customers. What I learnt in this class was Lists, and how to use List<T>. List<T> Class Represents a strongly typed list of objects that can be accessed by an index or foreach loop and it stores values of specific types without the need to for the types to be casted to or from the object. It also provides methods to search, sort, and manipulate lists. Lists are flexible compared to arrays. They are dynamic which means that they can shrink and grow as needed. Lists are very convenient to use when we do not know how much storage to accumulate for it. There are important properties and methods using lists such as add, insert, count, and remove. We typically use a foreach loop to iterate the list and for the Customer Maintenance project, we used foreach loop for iteration and the reason behind choose List<T> was because of its dynamic nature since it is ideal for storing and retrieving large number of elements.

Dynamic link library, also known as DLL is another topic we learnt in this class. DLLs provides a mechanism for shared data and codes that helps in sharing important or necessary codes without requiring applications to be linked or recompiled. In this class, we also learnt how to link our databases with our programs. Apart from learning about List<T>, DLL and linking databases, we also learnt more about object-oriented programming. Object oriented programming consists of abstraction: which is taking a concept and treating it as an object, encapsulation: which is used to hide the values or the state of a structured data object in a class by only allowing access to authorized personell, polymorphism: which is used to subtype a class and inheritance: which allows a new (child, sub or derived)class to derive from an existing (parent, base or super) class.

Until programming 2, we only used existing exceptions. In this class, we were taught how to create and handle exceptions. Exceptions can be handled by using parse which is converting to a type of variable, call stack, which is using try, catch and finally statements, or exception propagation. Try-catch statements were taught in programming 1 but in this class, we were introduced to the finally statement. In a try block we write our normal program, catch is where we deal with the exception and finally is where the codes are executed regardless of exceptions. When users define their own exception classes it is created as Application Exception class also known as exception propagation. At the end, we also learnt XML, which is eXtensible Markup Language. XML is a markup language much like HTML and is designed to store and transport data. We used XMLReader Class to read XML as it lets us run through the XML content, one element at a time. Another class called XMLDocument class also exists that reads the entire CML content into memory and then lets us navigate as we want. However, XMLReader is commonly used as it is more efficient and faster.