COLLECTIONS

WHAT ARE COLLECTIONS?

- Similar to an array, a collection is used to hold other objects.
- They are also more flexible and efficient than arrays.

COLLECTION TYPES

ARRAYLIST

An array list is a collection that's similar to an array, but it can change its capacity as elements are added or removed.

ARRAYLIST

EXAMPLE

```
// The ArrayList will store elements as objects
ArrayList numbers = new ArrayList();
numbers.add(5);
            foreach (int i in numbers)
{
            Console.WriteLine(i);
}
```

ARRAYLIST METHODS

- add(object): Adds an object to the end of the list.
- Count: Returns the number of elements in the list
- Insert(index, object): Adds an object at a specific location.

HASHTABLE

- A Hashtable where elements are organized based on Key-Value pairs.
- Keys are used to access elements or values.

HASHTABLE

EXAMPLE

```
Hashtable ht = new Hashtable();

ht.Add("001", "John");
ht.Add("002", "Paul");
Console.WriteLine(ht["001"]);
```

OTHER COLLECTIONS

- A stack is a LIFO (Last in First Out) data structure.
- A Queue is a FIFO (First in First Out) data structure.

GENERIC COLLECTIONS

- Generics allows us to create typed collections, which can hold objects of any type.
- To declare a variable that refers to a typed collection, we need to list the type in angle brackets (<>) following the name of the collection class.
- To include, use "System.Collections.Generic";

GENERIC COLLECTIONS

EXAMPLE

```
List < int > numbers = new List < int >();
numbers.add(5);
              foreach (int i in numbers)
{
              Console.WriteLine(i);
}
```

RECAP

WHAT YOU SHOULD KNOW AT THIS POINT:

- Know what are collections.
- Know classes and namespaces that define collections.
- Generics and their role in collections
- Define and use Arraylists and Lists.
- Define and use Hashtables. Know when to use Hashtables.