[0940] Generic Collections

Several years ago, C# introduced a new set of Collection classes and interfaces that support type-safety via Generics. These new classes are in the System.Collections.Generic namespace. Many of them have the same name as their non-generic counterpart, however Microsoft did change the name of "ArrayList" to just "List". #nocluewhy

So, assuming that we want to make sure our ArrayList only contains ints, we can change it to use the generic List collection instead. Here's the new line of code:

```
            myJunk[0] = new List<int>() { 3, 1, "4", 1, 5 };
```

You read "List<int>" as "a List of ints". And now, with that change, our program will no longer compile. #yay
Instead, it will complain that we are trying to add the string "4" to our List of ints - which is actually what we want in this case.

Generic Collection Classes

Again, C#'s Generic Collection Classes have superseded the older, non-generic System.Collection classes. The new versions are in the System.Collections.Generic namespace. Here are the important things in that namespace:

List<T>	ICollection<T>, IEnumerable<T>, IList<T>
Dictionary<TKey, TValue>	ICollection<T>, IEnumerable<T>, IDictionary<TKey, TValue>
Stack<T>	ICollection*, IEnumerable<T>
Queue<T>	ICollection*, IEnumerable<T>
LinkedList<T>	ICollection<T>, IEnumerable<T>

Notes:

- Whenever you see the "<T>" suffix, it means that the class (or interface) can use any type you choose to specify as "T". For example, "List<Car>" is a List collection of Car objects. In other words, "T" is a placeholder for any type you want to use.
- Stack<T> and Queue<T> implement the older, non-generic ICollection interface. #grrr #dumb
- For Java Programmers: Confusingly, in Java, "List" is an interface that is implemented by either the "ArrayList" or "LinkedList" classes. In C#, List is a concrete class that is very similar to ArrayList and LinkedList is a concrete class that is actually a Dictionary, not a list. #nocluewhy

More Info: **https://programmingwithmosh.com/csharp/csharp-collections/ (https://programmingwithmosh.com/csharp/csharp-collections/)**