Abstract Data Type

- Doesn't dictate how the data is organized
- Dictate the operations you can perform
- Concrete data structure is usually a concrete class
- Abstract data type is usually an interface
- It's more about behavior instead of operations that we can do
- In Java they are normally interfaces

ArrayList

- **Good for random access if we know the index**
- **Good for iterating over items in the list**
- **Not good for inserting items in any position other the and**
- **Not good for deleting items from list**
- **Not good for retrieving items from list if don't have it's index**
- Resizable array implementation of the List interface
- Data is stored in Array which is called Backing Array
- Adding items to existing list can be slow if the size isn't large enough to accommodate the new items. The backing array is already full
- Removing items from ArrayList can be slow aswell because all of the elements will have to shift down
- We can specify the capacity
    - Capacity is the maximum number of items that Array can store before it's going be have to be resized
    - **Default Capacity when we initializing empty ArrayList is 10**

Operation	**Time complexity**
get()	**O(1)**
Set()	**O(1)**
add()	**O(1)**
add(int position)	**O(n)**
remove()	**O(n)**