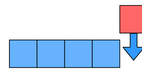
Inserting into an ArrayList

There are four ways to add elements in an **ArrayList**:

Inserting a single element at the end

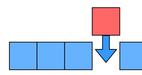
To add a single element at the end of the List, the `add(E e)` method can be used, where `E` refers to any type of object. This method will check if there is sufficient capacity in the **ArrayList**. If the **ArrayList** is full, then it will resize it and insert the element at the end.



Element inserted at the end of a list

Inserting a single element at a given index

We can also insert an element at a particular index using the `add(int index, E element)` method. This method will insert the element at the given index and will shift the element currently at that position (if any) and any subsequent elements to the right. This method will throw `IndexOutOfBoundsException` if the provided index is less than zero or greater than the size of **ArrayList**.

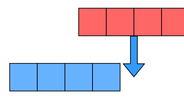


Element inserted at a particular index of a list

Inserting multiple elements from another Collection

If we have a **Collection** and we need to add all its elements to another **ArrayList**, then the `addAll(Collection c)` method can be used. This method will add all the elements at the end of the **ArrayList**.

```
list.addAll(anotherList)
```

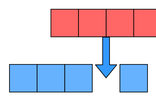


A list inserted at the end of another list

Inserting multiple elements from another Collection at a particular index

If we have a Collection and need to add all its elements to another **ArrayList** at a particular index, then the `addAll(int index, Collection c)` method can be used. This method inserts all of the specified collection elements into this list, starting at the specified position. It also shifts the element currently at that position (if any) and any subsequent elements to the right.

```
list.addAll(3, anotherList)
```



A list inserted at the a particular index of another list

```
import java.util.ArrayList;
import java.util.List;

public class ArrayListDemo {

    public static void main(String args[]) {
        List list = new ArrayList();
        list.add(1);
        list.add(2);
        list.add(3);
        System.out.println(list);

        list.add(4);  // This will add 4 at the end of the List.
        System.out.println(list);

        list.add(1, 50); // This will add 50 at index 1. All the other elements will be shifted to right.
        System.out.println(list);

        List newList = new ArrayList(); // Creating a new List which will be added to original list.
        newList.add(150);
        newList.add(160);

        list.addAll(newList); // This will add all the elements present in newList to list.
        System.out.println(list);
    }
}

```

Run Save Reset

In the above example, you must have encountered a warning message stating, "`ArrayListDemo.java uses unchecked or unsafe operations`". The reason for this is that our **ArrayList** is of **raw type**, meaning that while creating the **ArrayList**, we did not define what type of elements this **ArrayList** can hold. If we had defined the type of elements it can hold when we created the **ArrayList**, then it is called a **parameterized type**. It can be done as shown below.

```
List<String> list = new ArrayList<>();
```

So, we need to provide the type of object within `<>` while creating the list.

Creating a parameterized Collection is very important. Without it, there can be serious errors, which we will see in the next lesson when we discuss **ArrayList** iteration.

Fetching elements from an ArrayList

To fetch an element from **ArrayList**, we can use the `get(int index)` method. This method takes an `index` as input and returns the element at that `index`. The `index` provided should be greater than zero and should be less than **ArrayList** size.

We can fetch the size of the ArrayList using the `size()` method.

```
import java.util.ArrayList;
import java.util.List;

public class ArrayListDemo {

    public static void main(String args[]) {
        List<Integer> list = new ArrayList<>();
        list.add(1);
        list.add(2);
        list.add(3);
        System.out.println(list);

        System.out.println("The element at index two is " + list.get(1));

        System.out.println("The size of the List is  " + list.size());

    }
}

```

Run Save Reset