Finding the minimum element in a Collection

The `min(Collection c)` method can be used to find the minimum element in a **Collection**. The elements present in the **Collection** must implement the **Comparable** interface. If the elements do not implement the Comparable interface, we can use another overloaded method, `min(Collection c, Comparator comp)`. This method takes a **Comparator** as an argument that is used to compare the elements. This method iterates over the entire collection; hence it requires time proportional to the size of the collection.

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

public class ArrayListDemo {

    public static void main(String args[]) {
        List<Integer> list = new ArrayList<>();
        list.add(34);
        list.add(12);
        list.add(9);
        list.add(76);
        list.add(29);
        list.add(75);

        System.out.println("The minimum element in the List is: " + Collections.min(list));
    }
}

```

Finding the maximum element in a Collection

The `max(Collection c)` method can be used to find the maximum element in a **Collection**. The elements present in the **Collection** must implement the **Comparable** interface. If the elements do not implement the **Comparable** interface, we can use another overloaded method `max(Collection c, Comparator comp)`. This method takes a **Comparator** as an argument that is used to compare the elements. This method iterates over the entire Collection; hence it requires time proportional to the size of the Collection.

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

public class ArrayListDemo {

    public static void main(String args[]) {
        List<Integer> list = new ArrayList<>();
        list.add(34);
        list.add(12);
        list.add(9);
        list.add(76);
        list.add(29);
        list.add(75);

        System.out.println("The maximum element in the List is: " + Collections.max(list));
    }
}

```

Finding the frequency of elements in a Collection

There is a `frequency(Collection c, object o)` method that can be used to find the frequency of a given element in the Collection. This method iterates the entire Collection so the time complexity is proportional to the size of the collection.

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

public class ArrayListDemo {

    public static void main(String args[]) {
        List<Integer> list = new ArrayList<>();
        list.add(9);
        list.add(12);
        list.add(9);
        list.add(76);
        list.add(9);
        list.add(75);

        System.out.println("Total number of times,9 is present in the List is: " + Collections.frequency(
    }
}

```

Run Save Reset