```
cd /home/cg/root/63e714addda77

[Tabu, feeza, faiza]
[Tabu, feeza, sonu, faiza]
[Tabu, feeza, sufiya, faiza]
[Tabu, faiza, feeza, sufiya]
[sufiya, feeza, faiza, Tabu][sufiya, feeza, faiza
    , Tabu]
[feeza, faiza]Iterating synchronized ArrayList
    :sufiyafeeza
faiza
Tabu
[sufiya, Tabu, faiza, feeza]
ArrayList items: [5, 5, 5, 5, 5, 5, 5, 5, 5, 5]
[hello, world, GOOD][world, GOOD]
truetrue
[one, two]false
[one, two]
[one, two]
one
twonull
[task5, task4, task3, task1, task2]
```

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.Iterator;
import java.util.LinkedList;
import java.util.List;
import java.util.PriorityQueue;
import java.util.Queue;
import java.util.concurrent
    .ArrayBlockingQueue;
import java.util.concurrent.BlockingQueue;

public class ArrayListExample {

    public static void main(String[] args) {
        List<String> al = new ArrayList
            <String>();
        al.add("Tabu");
        al.add("feeza");
        al.add("faiza");
        System.out.println(al);
        al.add(2, "sonu");
        System.out.println(al);
        al.set(2, "sufiya");
```

Online Java Compiler

Project Login

Execute | Share Source Code Output

```
22        al.set(2, "sufiya");
23        System.out.println(al);
24        /*
25         * al.remove(0); System.out.println
              (al); al.clear(); System.out
              .println(al);
26         */
27        Collections.sort(al);
28        System.out.println(al);
29        Collections.sort(al, Collections
              .reverseOrder());
30        System.out.println(al);
31        ArrayList<String> al1 = new
              ArrayList<String>();
32        al1.addAll(al);
33        System.out.println(al1);
34        ArrayList<String> al2 = new
              ArrayList<String>(al.subList(1,
              3));
35        System.out.println(al2);
36
37        al = Collections.synchronizedList(al
              );
38
```

```
39          System.out.println("Iterating
                synchronized ArrayList:");
40          synchronized (al) {
41              Iterator<String> iterator = al
                    .iterator();
42              while (iterator.hasNext())
43                  System.out.println(iterator
                        .next());
44          }
45          Collections.swap(al, 1, 3);
46          System.out.println(al);
47          ArrayList<Integer> intlist = new
                ArrayList<Integer>(Collections
                .nCopies(10, 5));
48          System.out.println("ArrayList items:
                " + intlist);
49
50          Queue<String> q = new LinkedList
                <>();
51          String a[] = { "hello", "world",
                "GOOD" };
52          Collections.addAll(q, a);
53          System.out.println(q);
54
```

```
56
57          BlockingQueue<String> queue = new
                ArrayBlockingQueue<>(2);
58
59          System.out.println(queue.offer("one"
                ));
60          System.out.println(queue.offer("two"
                ));
61          System.out.println(queue);
62          System.out.println(queue.offer
                ("three"));
63          System.out.println(queue);
64
65          Queue<String> queue2 = new
                LinkedList<>();
66          queue2.offer("one");
67          queue2.offer("two");
68          System.out.println(queue2);
69          System.out.println(queue2.poll());
70          System.out.println(queue2.poll());
71          System.out.println(queue2.poll());
72
73          PriorityQueue<String> tasks = new
                PriorityQueue<String>(Comparator
                .reverseOrder());
74
```

```
64
65          Queue<String> queue2 = new
                LinkedList<>();
66          queue2.offer("one");
67          queue2.offer("two");
68          System.out.println(queue2);
69          System.out.println(queue2.poll());
70          System.out.println(queue2.poll());
71          System.out.println(queue2.poll());
72
73          PriorityQueue<String> tasks = new
                PriorityQueue<String>(Comparator
                .reverseOrder());
74          tasks.add("task1");
75          tasks.add("task4");
76          tasks.add("task3");
77          tasks.add("task2");
78          tasks.add("task5");
79          System.out.println(tasks);
80      }
81
82  }
```