```java

import java.util.ArrayList;

public class JavaSkillsForInterview {

    public static void main(String[] args) {
        // String
        String s = "abc";
        s.charAt(0);
        s.length();
        s.substring(0, 1);
        s.substring(1);
        s.equals("b");
        s = s.trim();
        s.indexOf("a");
        s.indexOf("a", 1);
        s.lastIndexOf("a");
        s.indexOf("a", 1);
        s.toCharArray();
        Integer.valueOf(s); // returns an Integer object
        Integer.parseInt(s); // returns an int primitive
        String.valueOf(s); // integer to string
        // StringBuilder
```

```
35          StringBuilder sb = new StringBuilder();
36          sb.append("a");
37          sb.insert(0, "a");
38          sb.deleteCharAt(sb.length() - 1);
39          sb.reverse();
40          sb.toString();
41          // Array
42          int[] a = new int[10];
43          char[] b = { 'a', 'b' };
44          int[][] c = new int[10][10];
45          int m = a.length;
46          int n = c[0].length;
47          int max = Integer.MAX_VALUE;
48          int min = Integer.MIN_VALUE;
49          Arrays.sort(a);
50          for (int i = 0; i < c.length; i++) {
51              System.out.println(c[i]);
52          }
53          // List
54          List<Integer> list = new ArrayList<Integer>();
55          ArrayList<Integer> list1 = new ArrayList<Integer>();
56          List<List<Integer>> list2 = new ArrayList<List<Integer>>();
57          list.add(0);
58          list.add(0, 1);
```

```
59          list.get(0);
60          list.size();
61          list.remove(list.size() - 1);
62          Collections.sort(list);
63          Collections.sort(list, Collections.reverseOrder());
64          Collections.sort(list, new Comparator<Integer>() {
65              @Override
66              public int compare(Integer o1, Integer o2) {
67                  return o1 - o2;// 0->1
68                  // return o2-o1; 1->0
69              }
70          });
71          // Stack
72          Stack<Integer> stack = new Stack<Integer>();
73          stack.push(0);
74          stack.pop();
75          stack.peek();
76          stack.isEmpty();
77          stack.size();
78          // Queue add ------> remove, peek
79          Queue<Integer> q = new LinkedList<Integer>();
80          q.add(0);
81          q.remove();
82          q.peek();
```

```
83          q.isEmpty();
84          q.size();
85          // HashMap
86          HashMap<Character, Integer> map = new HashMap<Character, Integer>();
87          map.put('c', 1);
88          map.get('c');
89          if (map.containsKey('c')) {
90          }
91          if (map.containsValue(1)) {
92          }
93          for (Character d : map.keySet()) {
94          }
95          for (Integer i : map.values()) {
96          }
97          map.isEmpty();
98          map.size();
99          // HashSet
100         HashSet<Integer> set = new HashSet<Integer>();
101         set.add(0);
102         set.remove(0);
103         if (set.contains(0)) {
104         }
105         set.isEmpty();
106         set.size();
```

```
107         // mini heap
108         PriorityQueue<Integer> pq = new PriorityQueue<Integer>();
109         pq.add(0);
110         pq.remove();
111         pq.peek();
112         pq.isEmpty();
113         pq.size();
114         while (!pq.isEmpty()) {
115         }
116     }
117
118 }
119
```