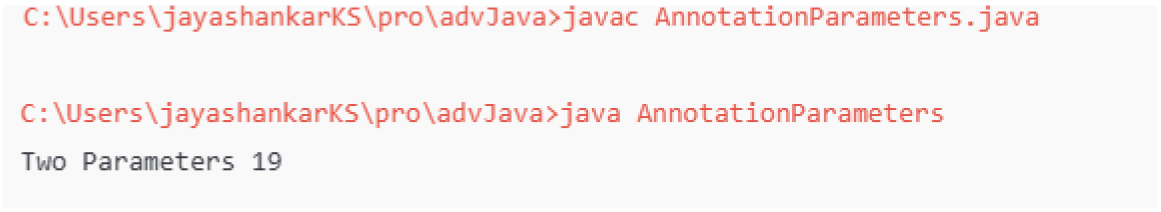
AAT 1

Program 1.

```
import java.lang.annotation.*;
import java.lang.reflect.*;
@Retention(RetentionPolicy.RUNTIME)
@interface MyAnno{
   String str();
   int val();
}
class AnnotationParameters{
   @MyAnno(str="Two Parameters",val=19)
   public static void myMeth(String str,int i)
   {
      AnnotationParameters ob = new AnnotationParameters();
      try{
         Class<?> c = ob.getClass();
         Method m = c.getMethod("myMeth",String.class,int.class);
         MyAnno anno = m.getAnnotation(MyAnno.class);
         System.out.println(anno.str()+" "+anno.val());
      }catch(NoSuchMethodException exc)
      {
         System.out.println("Method Not Found.");
      }
   }
   public static void main(String[] args) {
      myMeth("test",10);
```

```
    }
}
```

Output:

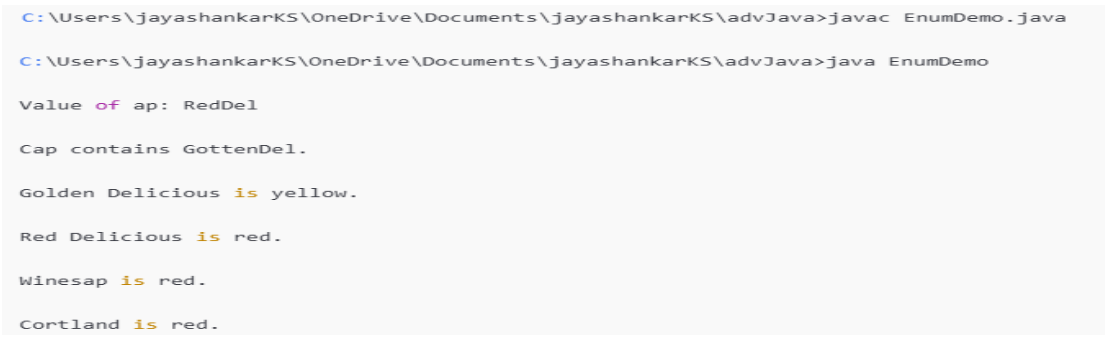


```
C:\Users\jayashankarKS\pro\advJava>javac AnnotationParameters.java

C:\Users\jayashankarKS\pro\advJava>java AnnotationParameters
Two Parameters 19
```

Program 2.

```
class EnumDemo {
    enum Apple{
        Jonathan, GoldenDel, RedDel, Winesap, Cortland
    }
    public static void main(String[] args) {
        Apple ap;
        ap = Apple.RedDel;
        System.out.println("Value of ap: "+ap);

        ap = Apple.GoldenDel;
        if(ap == Apple.GoldenDel)
            System.out.println("ap contains GoldenDel.");

        switch(ap)
        {
            case Jonathan -> System.out.println("Jonathan is red.");
            case GoldenDel -> System.out.println("Golden Delicious is yellow.");
            case RedDel -> System.out.println("Red Delicious is red.");
            case Winesap -> System.out.println("Winesap is red.");
            case Cortland -> System.out.println("Cortland is red.");
        }
    }
```

Output:



```
C:\Users\jayashankarKS\OneDrive\Documents\jayashankarKS\advJava>javac EnumDemo.java

C:\Users\jayashankarKS\OneDrive\Documents\jayashankarKS\advJava>java EnumDemo

Value of ap: RedDel

Cap contains GottenDel.

Golden Delicious is yellow.

Red Delicious is red.

Winesap is red.

Cortland is red.
```

Program 3.

```
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.reflect.Method;
@Retention(RetentionPolicy.RUNTIME)
@interface MyAnno{
    String str();
    int val();
}
class AnnotationExample{
    @MyAnno(str = "Annotation Example",val = 100)
    public static void myMeth() {
        AnnotationExample ob = new AnnotationExample();
        try{
            Class<?> c = ob.getClass();
            Method m = c.getMethod("myMeth");
            MyAnno anno = m.getAnnotation(MyAnno.class);
            System.out.println(anno.str()+" "+anno.val());
        }catch(NoSuchMethodException exc)
        {
            System.out.println("Method Not Found.");
        }
    }
    public static void main(String[] args) {
        myMeth();
    }
}
```

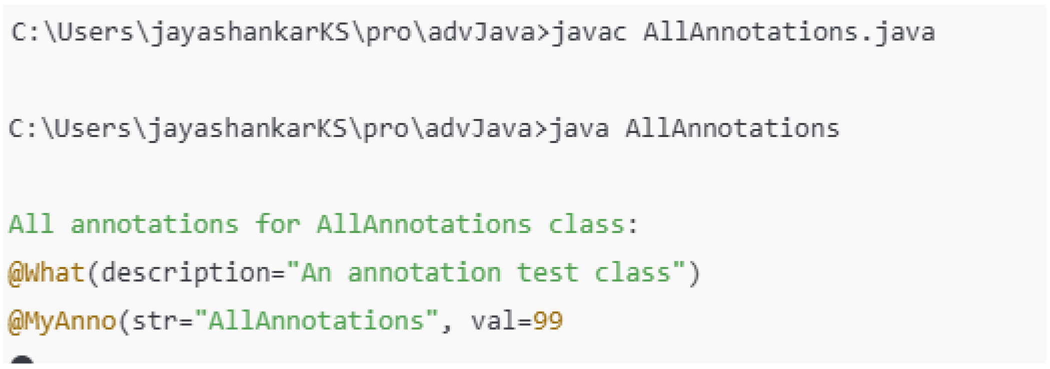
```
C:\Users\jayashankarKS\pro\advJava>javac AnnotationExample.java

C:\Users\jayashankarKS\pro\advJava>java AnnotationExample

Annotation Example 100
```

Program 4.

```
import java.lang.annotation.*;
import java.lang.reflect.*;
@Retention(RetentionPolicy.RUNTIME)
@interface MyAnno{
    String str();
    int val();
}
@Retention(RetentionPolicy.RUNTIME)
@interface What{
    String description();
```

```
}
@What(description="An annotation test class")
@MyAnno(str="AllAnnotations",val=99)
public class AllAnnotations {
   @What(description="An annotation test method")
   @MyAnno(str="Testing",val=100)
   public static void myMeth()
   {
      AllAnnotations ob = new AllAnnotations();
      try{
         Annotation annos[] = ob.getClass().getAnnotations();
         System.out.println("All annotations for AllAnnotations class:");
         for(Annotation a : annos)
            System.out.println(a);


         System.out.println();
         Method m = ob.getClass().getMethod("myMeth");
         annos = m.getAnnotations();
         System.out.println("All annotations for myMeth:");
         for(Annotation a : annos)
            System.out.println(a);
      }catch(NoSuchMethodException exc)
      {
         System.out.println("Method Not Found.");
      }
   }
   public static void main(String args[])
```

```
    {
        myMeth();
    }
}
```

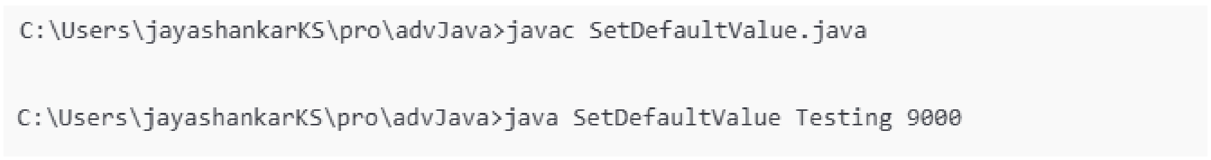


```
C:\Users\jayashankarKS\pro\advJava>javac AllAnnotations.java

C:\Users\jayashankarKS\pro\advJava>java AllAnnotations

All annotations for AllAnnotations class:
@What(description="An annotation test class")
@MyAnno(str="AllAnnotations", val=99
```

Program 5.

```
import java.lang.annotation.*;
import java.lang.reflect.*;
@Retention(RetentionPolicy.RUNTIME)
@interface MyAnno {
    String str() default "Testing";
    int val() default 9000;
}
public class SetDefaultValue {
    @MyAnno() // Using annotation without parameters, defaults will be used
    public static void myMeth() {
        SetDefaultValue ob = new SetDefaultValue();
        try {
            Class<?> c = ob.getClass();
            Method m = c.getMethod("myMeth");

            MyAnno anno = m.getAnnotation(MyAnno.class);

            System.out.println(anno.str() + " " + anno.val());

        } catch (NoSuchMethodException exc) {

            System.out.println("Method Not Found.");

        }

    }

    public static void main(String[] args) {

        myMeth();

    }

}
```

Output:

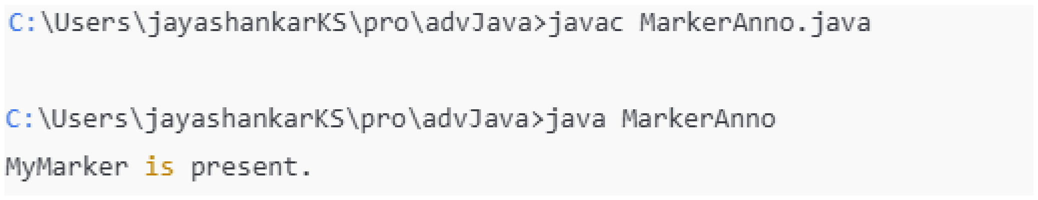


```
C:\Users\jayashankarKS\pro\advJava>javac SetDefaultValue.java

C:\Users\jayashankarKS\pro\advJava>java SetDefaultValue Testing 9000
```

Program 6.

```
import java.lang.annotation.*;

import java.lang.reflect.*;

@Retention(RetentionPolicy.RUNTIME)

@interface MyMarker{}

public class MarkerAnno {

    @MyMarker

    public static void myMeth()
    {
        MarkerAnno ob = new MarkerAnno();
        try{
            Method m = ob.getClass().getMethod("myMeth");
            if(m.isAnnotationPresent(MyMarker.class))
                System.out.println("MyMarker is present.");
        }catch(NoSuchMethodException exc)
        {
            System.out.println("Method Not Found.");
        }
    }
    public static void main(String[] args) {
        myMeth();
    }
}
```

Output:

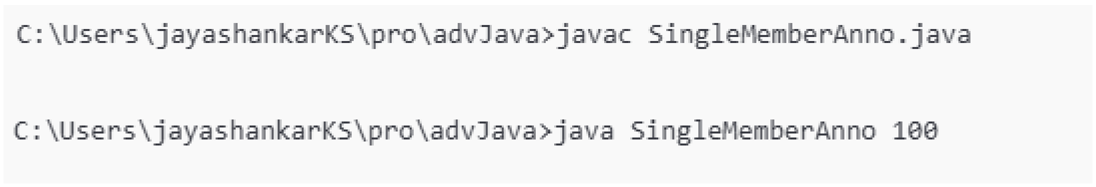


```
C:\Users\jayashankarKS\pro\advJava>javac MarkerAnno.java

C:\Users\jayashankarKS\pro\advJava>java MarkerAnno
MyMarker is present.
```

Program 7.

```
import java.lang.annotation.*;
import java.lang.reflect.*;
@Retention(RetentionPolicy.RUNTIME)
@interface MySingle{
    int value();
}
public class SingleMemberAnno {
   @MySingle(100)
   public static void myMeth(){
      SingleMemberAnno ob = new SingleMemberAnno();
      try{
         Method m = ob.getClass().getMethod("myMeth");
         MySingle anno = m.getAnnotation(MySingle.class);
         System.out.println(anno.value());
      }catch(NoSuchMethodException exc)
      {
         System.out.println("Method Not Found.");
      }
   }
   public static void main(String[] args) {
      myMeth();
   }
}
```

Output:



```
C:\Users\jayashankarKS\pro\advJava>javac SingleMemberAnno.java

C:\Users\jayashankarKS\pro\advJava>java SingleMemberAnno 100
```

Program 8.

```
public class SubStringCons {
   public static void main(String[] args) {
      byte ascii[] = {65, 66, 67, 68, 69, 70 };
      String s1 = new String(ascii);
```

```
        System.out.println(s1);

        String s2 = new String(ascii, 2, 3);

        System.out.println(s2);

    }

}
```

Output:

```
C:\Users\jayashankarKS\pro\advJava>java SubStringCons
ABCDEF
CDE
```

Program 9.

```
public class MakeString {

    public static void main(String[] args) {

        char c[] = {'J', 'a', 'v', 'a'};

        String s1 = new String(c);

        String s2= new String(s1);

        System.out.println(s1);

        System.out.println(s2);

    }

}
```

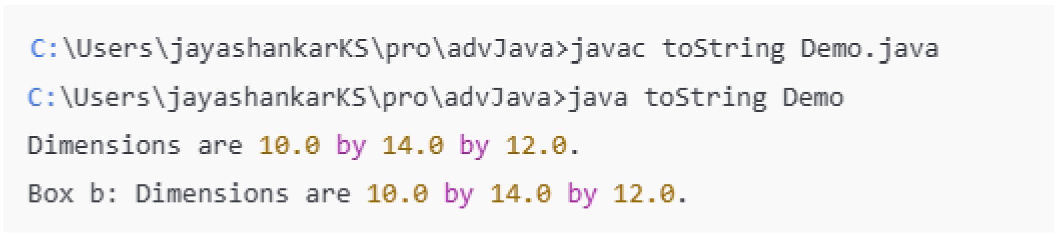
Output:

```
C:\Users\jayashankarKS\pro\advJava>javac MakeString.java
C:\Users\jayashankarKS\pro\advJava>java MakeString
Java
Java
```

Program 10.

```
class Box {

    double width; double height; double depth;

    Box(double w, double h, double d)
    {
        width = w;
        height = h;
        depth = d;
    }
   public String toString() {
        return "Dimensions are " + width + " by " + depth + " by " + height + ".";
    }
}
class toStringDemo {
    public static void main(String[] args) {
        Box b = new Box(10, 12, 14);
        String s = "Box b: " + b;
        System.out.println(b);
        System.out.println(s);
    }
}
```

Output:

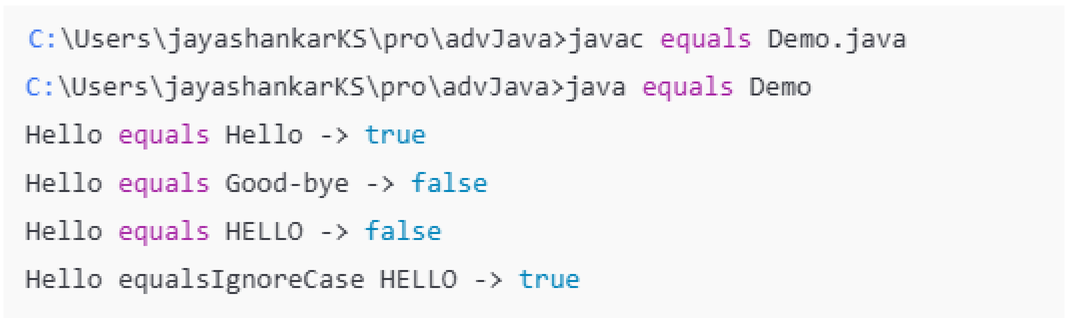


```
C:\Users\jayashankarKS\pro\advJava>javac toString Demo.java
C:\Users\jayashankarKS\pro\advJava>java toString Demo
Dimensions are 10.0 by 14.0 by 12.0.
Box b: Dimensions are 10.0 by 14.0 by 12.0.
```

Program 11.

```
class equalsDemo {
    public static void main(String args[]) {
        String s1 = "Hello";

        String s2 = "Hello";

        String s3 = "Good-bye";

        String s4 = "HELLO";

        System.out.println(s1 + " equals " + s2 + " -> " + s1.equals(s2));

        System.out.println(s1 + " equals " + s3 + " -> " + s1.equals(s3));

        System.out.println(s1 + " equals " + s4 + " -> " + s1.equals(s4));

        System.out.println(s1 + " equalsIgnoreCase " + s4 + " -> " +

        s1.equalsIgnoreCase(s4));

    }

}
```

Output:

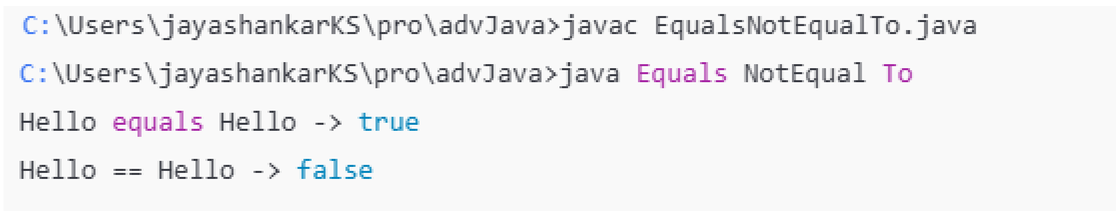


```
C:\Users\jayashankarKS\pro\advJava>javac equals Demo.java
C:\Users\jayashankarKS\pro\advJava>java equals Demo
Hello equals Hello -> true
Hello equals Good-bye -> false
Hello equals HELLO -> false
Hello equalsIgnoreCase HELLO -> true
```

Program 12.

```
class EqualsNotEqualTo {

    public static void main(String args[]) {

        String s1 = "Hello";

        String s2 = new String(s1);

        System.out.println(s1 + " equals " + s2 + " -> " + s1.equals(s2));

        System.out.println(s1 + " == " + s2 + " -> " + (s1 == s2));

    }}
```

Output:

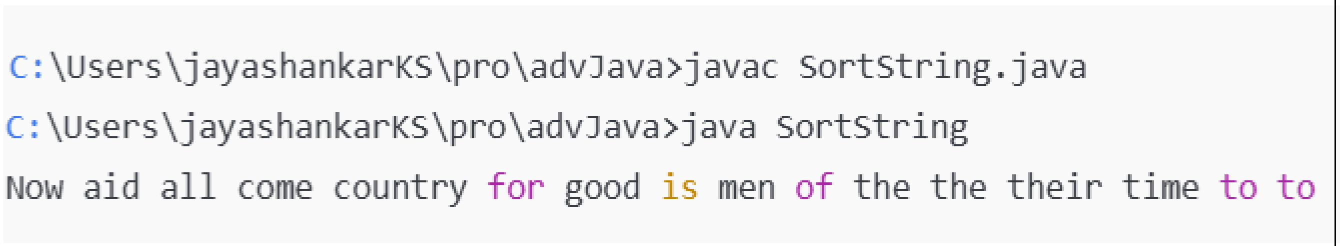


```
C:\Users\jayashankarKS\pro\advJava>javac EqualsNotEqualTo.java
C:\Users\jayashankarKS\pro\advJava>java Equals NotEqual To
Hello equals Hello -> true
Hello == Hello -> false
```

Program 13.

```
class SortString {
   static String arr[] = { "Now", "is", "the", "time", "for", "all", "good", "men", "to", "come",
"to", "the", "aid", "of", "their", "country" };
   public static void main(String args[]) {
      for(int j = 0; j < arr.length; j++) {
         for(int i = j + 1; i < arr.length; i++) {
            if(arr[i].compareTo(arr[j]) < 0)
            {
               String t = arr[j];
               arr[j] = arr[i];
               arr[i] = t;
            }
         }
         System.out.print(arr[j]+" ");
      }
   }
}
```

Output:



```
C:\Users\jayashankarKS\pro\advJava>javac SortString.java
C:\Users\jayashankarKS\pro\advJava>java SortString
Now aid all come country for good is men of the the their time to to
```

Program 14.

```
class indexOfDemo {
   public static void main(String args[]) {
      String s = "Now is the time for all good men " + "to come to the aid of their country.";
       System.out.println(s);
      System.out.println("indexOf(t) = " + s.indexOf('t'));
      System.out.println("lastIndexOf(t) = " + s.lastIndexOf('t'));
      System.out.println("indexOf(the) = " + s.indexOf("the"));
      System.out.println("lastIndexOf(the) = " + s.lastIndexOf("the"));
      System.out.println("indexOf(t, 10) = " + s.indexOf('t', 10));
      System.out.println("lastIndexOf(t, 60) = " + s.lastIndexOf('t', 60));
      System.out.println("indexOf(the, 10) = " + s.indexOf("the", 10));
      System.out.println("lastIndexOf(the, 60) = " + s.lastIndexOf("the",60));
   }
}
```

Output:

```
C:\Users\jayashankarKS\pro\advJava>javac indexOfDemo.java
C:\Users\jayashankarKS\pro\advJava>java indexOfDemo
Now is the time for all good men to come to the aid of their country.
indexOf(t) = 7
lastIndexOf(t) = 65
indexOf(the) = 7
lastIndexOf(the) = 55
indexOf(t, 10) = 11
lastIndexOf(t, 60) = 55
indexOf(the, 10) = 44
lastIndexOf(the, 60) = 55
```

Program 15.

```
import java.util.*;

public class ArrayListDemo {

    public static void main(String args[]) {

        ArrayList<String> al = new ArrayList<String>();

        System.out.println("Initial size of al: " + al.size());

        al.add("C");

        al.add("A");

        al.add("E");

        al.add("B");

        al.add("D");

        al.add("F");

        al.add(1, "A2");

        System.out.println("Size of al after additions: " + al.size());

        System.out.println("Contents of al: " + al);

        al.remove("F");
        al.remove(2);
        System.out.println("Size of al after deletions: " + al.size());
    }
}
```

Output:

```
C:\Users\jayashankarKS\pro\advJava>javac ArrayListDemo.java
C:\Users\jayashankarKS\pro\advJava>java ArrayListDemo
Initial size of al: 0
Size of al after additions: 7
Contents of al: [C, A2, A, E, B, D, F]
Size of al after deletions: 5
```

Program 16.

```
import java.util.*;
class arrayListToArray {
    public static void main(String args[]) {
        ArrayList<Integer> al = new ArrayList<Integer>();
        al.add(1);
        al.add(2);
        al.add(3);
        al.add(4);
        System.out.println("Contents of al: " + al);
        Integer ia[] = new Integer[al.size()];
        ia = al.toArray(ia);
        int sum = 0;
        for(int i : ia)
            sum += i;
        System.out.println("Sum is: " + sum);
    }
}
```

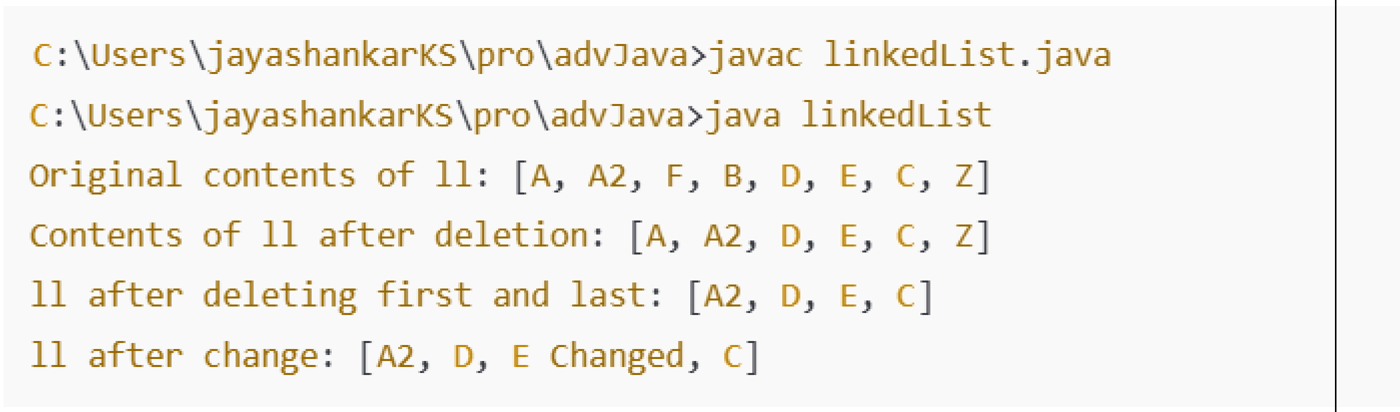
Output:

```
C:\Users\jayashankarKS\pro\advJava>javac arrayListToArray.java
C:\Users\jayashankarKS\pro\advJava>java arrayListToArray
Contents of al: [1, 2, 3, 4]
Sum is: 10
```

Program 17.

```
import java.util.*;
class linkedList {
    public static void main(String args[]) {
        LinkedList<String> ll = new LinkedList<String>();
        ll.add("F");
        ll.add("B");
        ll.add("D");
        ll.add("E");
        ll.add("C");
        ll.addLast("Z");
        ll.addFirst("A");
        ll.add(1, "A2");
        System.out.println("Original contents of ll: " + ll);
        ll.remove("F");
        ll.remove(2);
        System.out.println("Contents of ll after deletion: "+ ll);
        ll.removeFirst();
        ll.removeLast();
        System.out.println("ll after deleting first and last: "+ ll);
        String val = ll.get(2);
        ll.set(2, val + " Changed");
        System.out.println("ll after change: " + ll);
    }
}
```

Output:



```
C:\Users\jayashankarKS\pro\advJava>javac linkedList.java
C:\Users\jayashankarKS\pro\advJava>java linkedList
Original contents of ll: [A, A2, F, B, D, E, C, Z]
Contents of ll after deletion: [A, A2, D, E, C, Z]
ll after deleting first and last: [A2, D, E, C]
ll after change: [A2, D, E Changed, C]
```

Program 18.

```
import java.util.*;
class hashSet {
    public static void main(String args[]) {
        HashSet<String> hs = new HashSet<String>();
        hs.add("B");
        hs.add("A");
        hs.add("D");
        hs.add("E");
        hs.add("C");
        hs.add("F");
        System.out.println(hs);
    }
}
```

Output:

```
C:\Users\jayashankarKS\pro\advJava>javac hashSet.java
C:\Users\jayashankarKS\pro\advJava>java hashSet
[A, B, C, D, E, F]
```

Program 19.

```
import java.util.*;
class treeSet {
    public static void main(String args[]) {
        TreeSet<String> ts = new TreeSet<String>();
        ts.add("C");
        ts.add("A");
        ts.add("B");
        ts.add("E");
        ts.add("F");
        ts.add("D");
        System.out.println(ts);
    }
}
```

```
C:\Users\ankit\OneDrive\Documents\ankita\advJava>javac treeSet.java

C:\Users\ankit\OneDrive\Documents\ankita\advJava>java treeSet
[A, B, C, D, E, F]
```

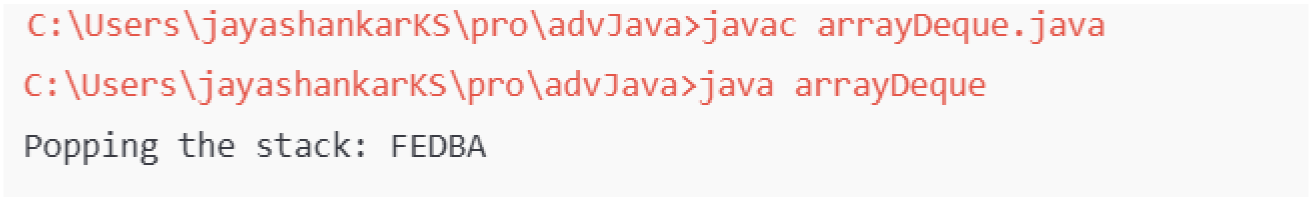
```
C:\Users\jayashankarKS\pro\advJava>javac treeSet.java
C:\Users\jayashankarKS\pro\advJava>java treeSet
[A, B, C, D, E, F]
```

Output:

Program 20.

```
import java.util.*;
public class arrayDeque {
    public static void main(String args[]) {
        ArrayDeque<String> adq = new ArrayDeque<String>();
        adq.push("A");
        adq.push("B");
        adq.push("D");
        adq.push("E");
        adq.push("F");
        System.out.print("Popping the stack: ");
        while(adq.peek() != null)
        System.out.print(adq.pop() + " ");
        System.out.println();
    }
}
```

Output:



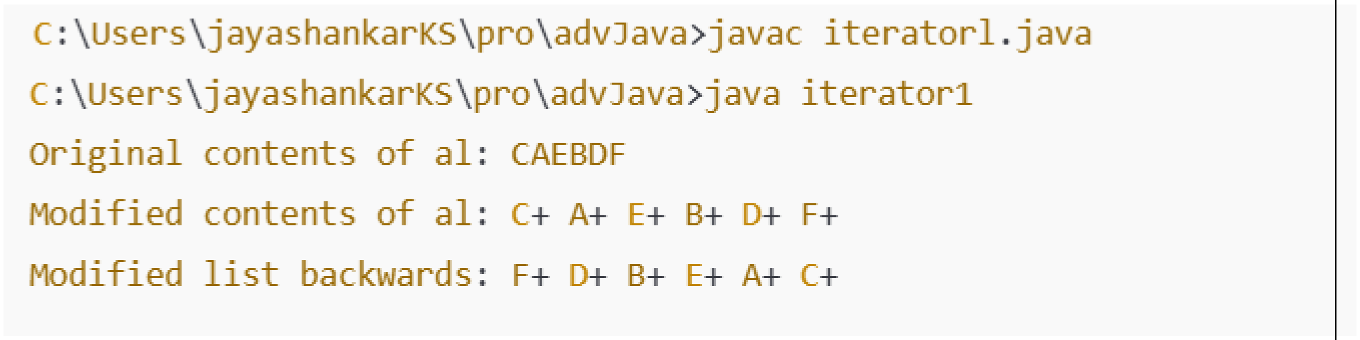
```
C:\Users\jayashankarKS\pro\advJava>javac arrayDeque.java
C:\Users\jayashankarKS\pro\advJava>java arrayDeque
Popping the stack: FEDBA
```

Program 21.

```
import java.util.*;
class iterator1 {
    public static void main(String args[]) {
        ArrayList<String> al = new ArrayList<String>();
        al.add("C");
        al.add("A");
        al.add("E");
        al.add("B");
        al.add("D");
        al.add("F");
        System.out.print("Original contents of al: ");
        Iterator<String> itr = al.iterator();
        while(itr.hasNext()) {
           String element = itr.next();
           System.out.print(element + " ");
        }
        System.out.println();
        ListIterator<String> litr = al.listIterator();
        while(litr.hasNext()) {
           String element = litr.next();
           litr.set(element + "+");
        }
        System.out.print("Modified contents of al: ");
        itr = al.iterator();
        while(itr.hasNext()) {
           String element = itr.next();
           System.out.print(element + " ");
        }
        System.out.println();
        System.out.print("Modified list backwards: ");
        while(litr.hasPrevious()) {
           String element = litr.previous();
           System.out.print(element + " ");
        }
```

```
        System.out.println();
    }
}
```

Output:



```
C:\Users\jayashankarKS\pro\advJava>javac iteratorl.java
C:\Users\jayashankarKS\pro\advJava>java iterator1
Original contents of al: CAEBDF
Modified contents of al: C+ A+ E+ B+ D+ F+
Modified list backwards: F+ D+ B+ E+ A+ C+
```

Program 22.

```
class changeCase {
    public static void main(String args[]) {
        String s = "This is a test.";
        System.out.println("Original: " + s);
        String upper = s.toUpperCase();
        String lower = s.toLowerCase();
        System.out.println("Uppercase: " + upper);
        System.out.println("Lowercase: " + lower);
    }
}
```

Output:

```
C:\Users\jayashankarKS\pro\advJava>java changeCase
Original:
This is a test.
Uppercase:
THIS IS A TEST.
Lowercase:
this is a test.
```

Program 23.

```
public class stringBuffer {
   public static void main(String args[]) {
      StringBuffer sb = new StringBuffer("This is a test.");
      sb.replace(5, 7, "was");
      System.out.println("After replace: " + sb);
   }
}
```

Output:

```
C:\Users\jayashankarKS\pro\advJava>javac stringBuffer.java
C:\Users\jayashankarKS\pro\advJava>java stringBuffer
After replace: This was a test.
```

Program 24.

```
public class deleteChar {
   public static void main(String args[]) {
      StringBuffer sb = new StringBuffer("This is a test.");
      sb.delete(4, 7);
      System.out.println("After delete: " + sb);
      sb.deleteCharAt(0);
```

```
        System.out.println("After deleteCharAt: " + sb);
    }
}
```

Output:

```
C:\Users\jayashankarKS\pro\advJava>javac deleteChar.java
C:\Users\jayashankarKS\pro\advJava>java deleteChar
After delete: This a test.
After deleteCharAt: his a test.
```

Program 25.

```
public class characterAt {
    public static void main(String args[]) {
        StringBuffer sb = new StringBuffer("Hello");
        System.out.println("buffer before = " + sb);
        System.out.println("charAt(1) before = " + sb.charAt(1));
        sb.setCharAt(1, 'i');
        sb.setLength(2);
        System.out.println("buffer after = " + sb);
        System.out.println("charAt(1) after = " + sb.charAt(1));
    }
}
```

Output:

```
C:\Users\jayashankarKS\pro\advJava>javac characterAt.java
C:\Users\jayashankarKS\pro\advJava>java characterAt
buffer before Hello
charAt(1) before = e
buffer after = Hi
charAt(1) after = i
```