MULTITHREADING:

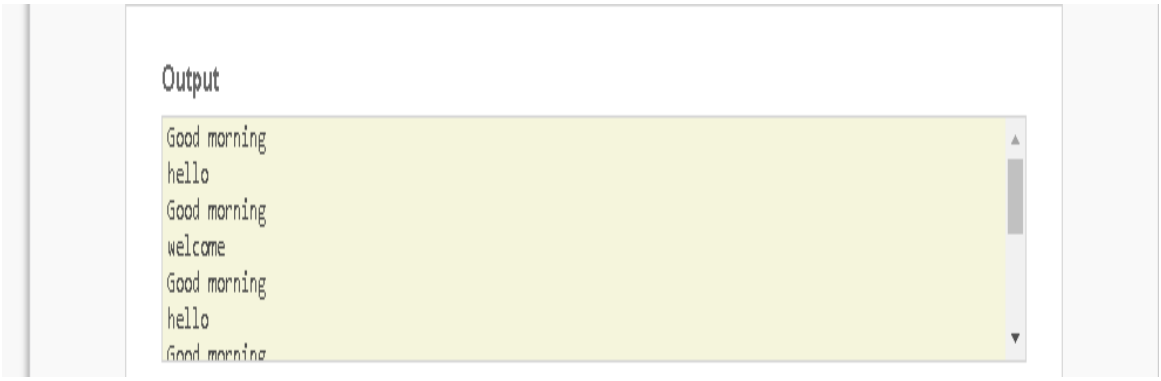
MULTITHREADING:

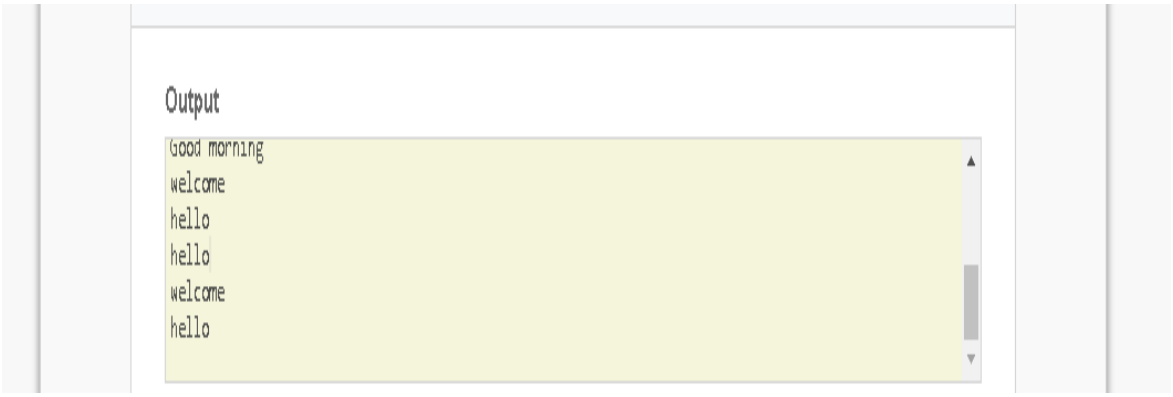
```
class GoodMorning extends Thread {
    synchronized public void run() {
        try {
            int i=0;
            while (i<5) {
                sleep(1000);
                System.out.println("Good morning ");
                i++;
            }
        } catch (Exception e) {
        }
    }
}

class Hello extends Thread {
    synchronized public void run() {
        try {
            int i=0;
            while (i<5) {
                sleep(2000);
                System.out.println("hello");
                i++;
            }
        } catch (Exception e) {
```

```
        }
    }
}

class Welcome extends Thread {
    synchronized public void run() {
        try {
            int i=0;
            while (i<5) {
                sleep(3000);
                System.out.println("welcome");
                i++;
            }
        } catch (Exception e) {
        }
    }
}

class threadclass {
public static void main(String args[]) {
        GoodMorning t1 = new GoodMorning();
        Hello t2 = new Hello();
        Welcome t3 = new Welcome();
        t1.start();
        t2.start();
        t3.start();
```

```
    }
}
```

OUTPUT:


Output
Good morning
hello
Good morning
welcome
Good morning
hello

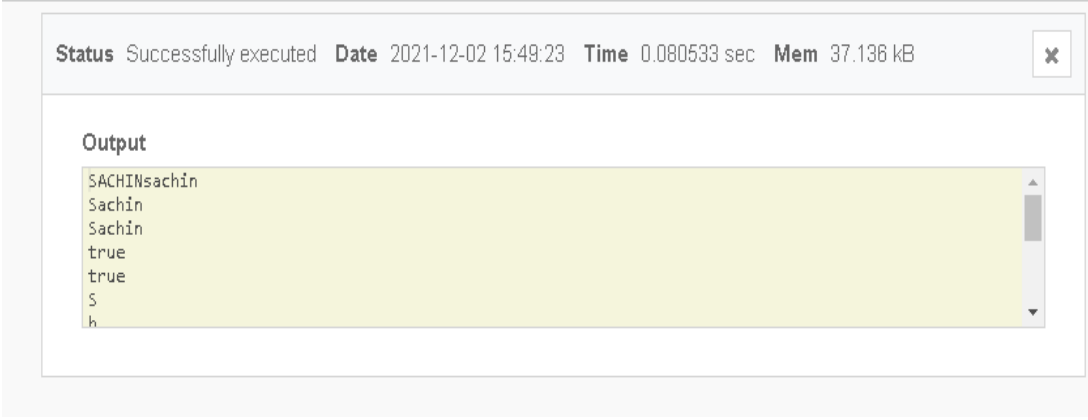

Output
Good morning
welcome
hello
hello
welcome
hello

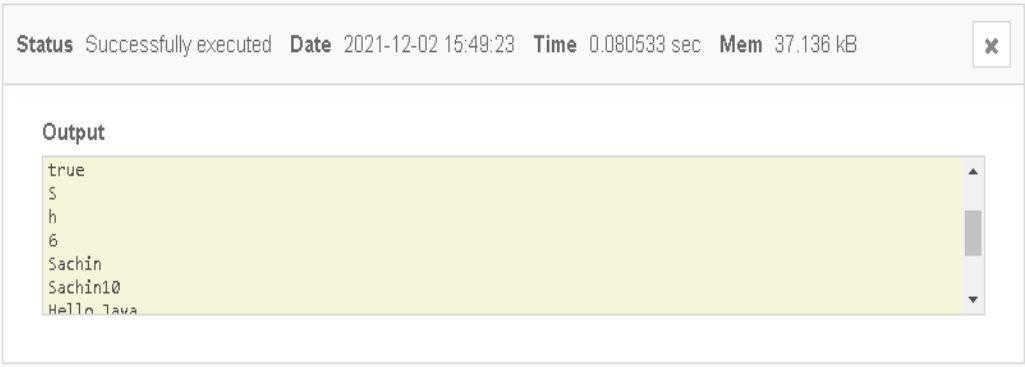
STRING OPERATIONS

```
import java.util.*;
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main" only if the class is public. */
 class StringExample
 {
    public static void main(String args[])
    {
    String s="Sachin";
    System.out.print(s.toUpperCase());
    System.out.println(s.toLowerCase());
    System.out.println(s);
    System.out.println(s.trim());
    System.out.println(s.startsWith("Sa"));//true
    System.out.println(s.endsWith("n"));
    System.out.println(s.charAt(0));//S
    System.out.println(s.charAt(3));
    System.out.println(s.length());
    String s4=s.intern();
    System.out.println(s4);
    String s6=String.valueOf(s);
    System.out.println(s6+10);
    StringBuilder sb=new StringBuilder("Hello ");
    sb.append("Java");//now original string is changed
```

```
    System.out.println(sb);
    sb.insert(1,"Java");//now original string is changed
    System.out.println(sb);
    System.out.println(sb.hashCode());
    sb.append("tpoint");
    String s5="Java is a programming language. Java is a platform. Java is an Island.";
    String s7=s5.replace("Java","Kava");//replaces all occurrences of "Java" to "Kava"
    System.out.println(s7);
    String s3=new String("Sachin");


  }
}
```

OUTPUT:


Status Successfully executed Date 2021-12-02 15:49:23 Time 0.080533 sec Mem 37.136 kB
Output
SACHINsachin
Sachin
Sachin
true
true
S
h


Status Successfully executed Date 2021-12-02 15:49:23 Time 0.080533 sec Mem 37.136 kB
Output
true
S
h
6
Sachin
Sachin10
Hello Java

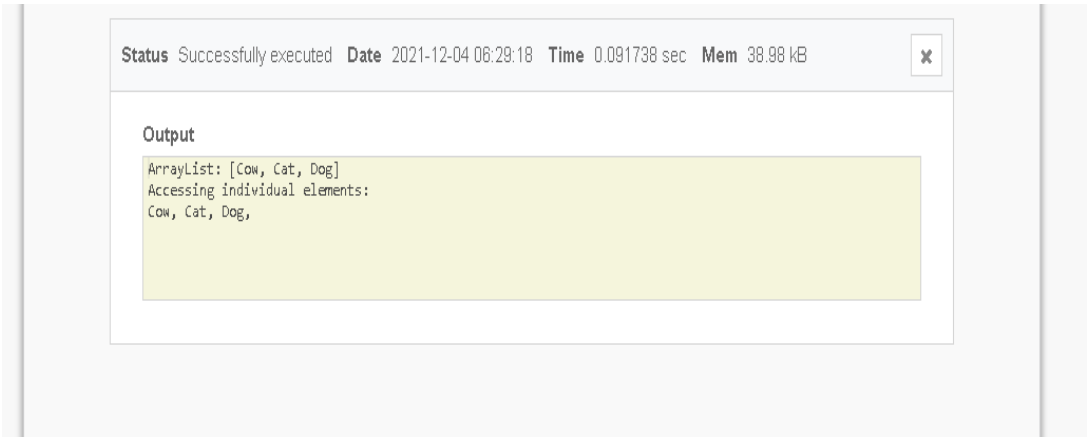

Status Successfully executed Date 2021-12-02 15:49:23 Time 0.080533 sec Mem 37.136 kB
Output
Sachin
Sachin10
Hello Java
HJavaello Java
356573597
Kava is a programming language. Kava is a platform. Kava is an Island.

COLLECTION FRAMEWORK:

ARRALIST:

```
import java.util.*;
import java.lang.*;
import java.io.*;
import java.util.ArrayList;

class Main {
  public static void main(String[] args) {

    // creating an array list
    ArrayList<String> animals = new ArrayList<>();
    animals.add("Cow");
    animals.add("Cat");
    animals.add("Dog");
    System.out.println("ArrayList: " + animals);

    // iterate using for-each loop
    System.out.println("Accessing individual elements:  ");

    for (String language : animals) {
      System.out.print(language);
      System.out.print(", ");
    }
  }
}
```

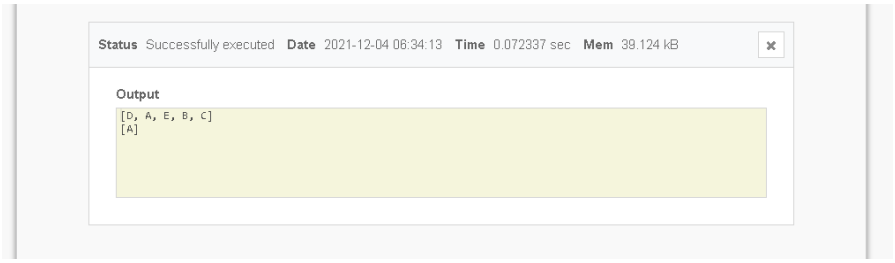
OUTPUT:


Status Successfully executed Date 2021-12-04 06:29:18 Time 0.091738 sec Mem 38.98 kB
Output
ArrayList: [Cow, Cat, Dog]
Accessing individual elements:
Cow, Cat, Dog,

LINKED LIST

```
import java.util.*;

class  Linkedlist{
        public static void main(String args[])
        {
                LinkedList<String> ll = new LinkedList<String>();
                ll.add("A");
                ll.add("B");
                ll.addLast("C");
                ll.addFirst("D");
                ll.add(2, "E");

                System.out.println(ll);

                ll.remove("B");
                ll.remove(3);
                ll.removeFirst();
                ll.removeLast();
               System.out.println(ll);
        }
}
```

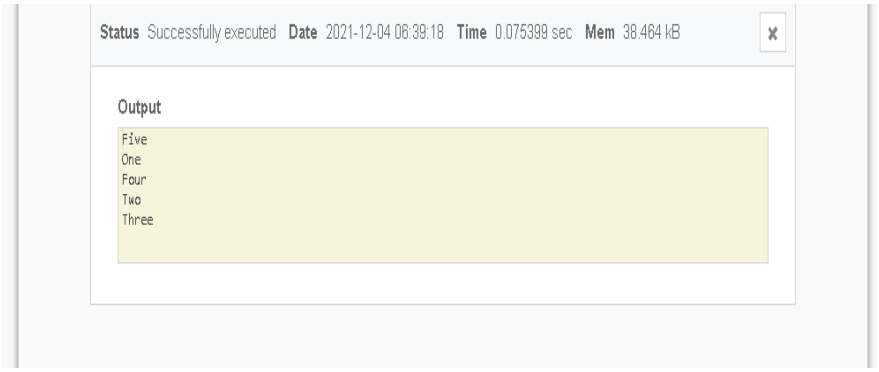
OUTPUT:


Status Successfully executed Date 2021-12-04 06:34:13 Time 0.072337 sec Mem 39.124 kB
Output
[D, A, E, B, C]
[A]

HASHSET:

```
import java.util.*;
class HashSet1{
 public static void main(String args[]){
  //Creating HashSet and adding elements
    HashSet<String> set=new HashSet();
          set.add("One");
          set.add("Two");
          set.add("Three");
          set.add("Four");
          set.add("Five");
          Iterator<String> i=set.iterator();
          while(i.hasNext())
          {
          System.out.println(i.next());
          }
 }
}
```

OUTPUT:



LINKEDHASHSET:

```
import java.util.LinkedHashSet;

public class LinkedHashSetExample
{

      // Main Method
    public static void main(String[] args)
    {
        LinkedHashSet<String> linkedset =
                           new LinkedHashSet<String>();

        // Adding element to LinkedHashSet
        linkedset.add("A");
        linkedset.add("B");
        linkedset.add("C");
        linkedset.add("D");

        // This will not add new element as A already exists
        linkedset.add("A");
        linkedset.add("E");

        System.out.println("Size of LinkedHashSet = " +
                                    linkedset.size());
        System.out.println("Original LinkedHashSet:" + linkedset);
        System.out.println("Removing D from LinkedHashSet: " +
                           linkedset.remove("D"));
        System.out.println("Trying to Remove Z which is not "+
                           "present: " + linkedset.remove("Z"));
        System.out.println("Checking if A is present=" +
                           linkedset.contains("A"));
        System.out.println("Updated LinkedHashSet: " + linkedset);
    }
}
```

OUTPUT:

Status Successfully executed **Date** 2021-12-04 06:43:12 **Time** 0.111694 sec **Mem** 37.48 kB

Output

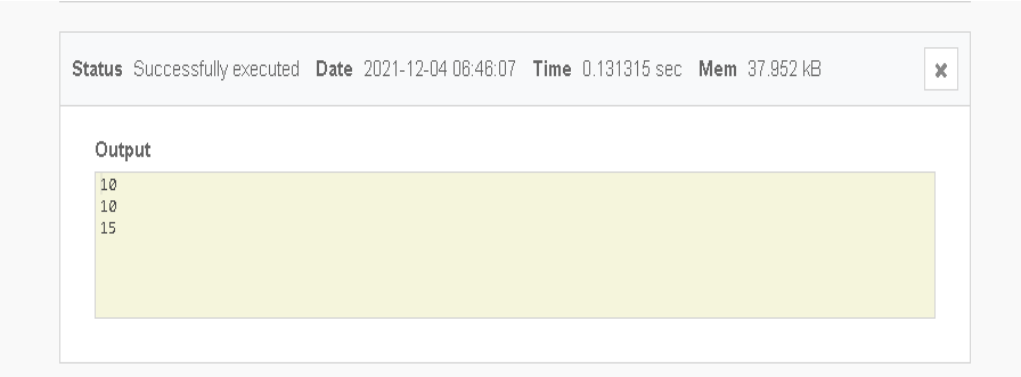
```
Size of LinkedHashSet = 5
Original LinkedHashSet:[A, B, C, D, E]
Removing D from LinkedHashSet: true
Trying to Remove Z which is not present: false
Checking if A is present=true
Updated LinkedHashSet: [A, B, C, E]
```

PRIORITYQUEUE:

```
import java.util.*;

class PriorityQueueDemo {

    // Main Method
   public static void main(String args[])
   {
       // Creating empty priority queue
       PriorityQueue<Integer> pQueue = new
PriorityQueue<Integer>();

       // Adding items to the pQueue using add()
       pQueue.add(10);
       pQueue.add(20);
       pQueue.add(15);

       // Printing the top element of PriorityQueue
       System.out.println(pQueue.peek());

       // Printing the top element and removing it
       // from the PriorityQueue container
       System.out.println(pQueue.poll());

       // Printing the top element again
       System.out.println(pQueue.peek());
   }
```

}

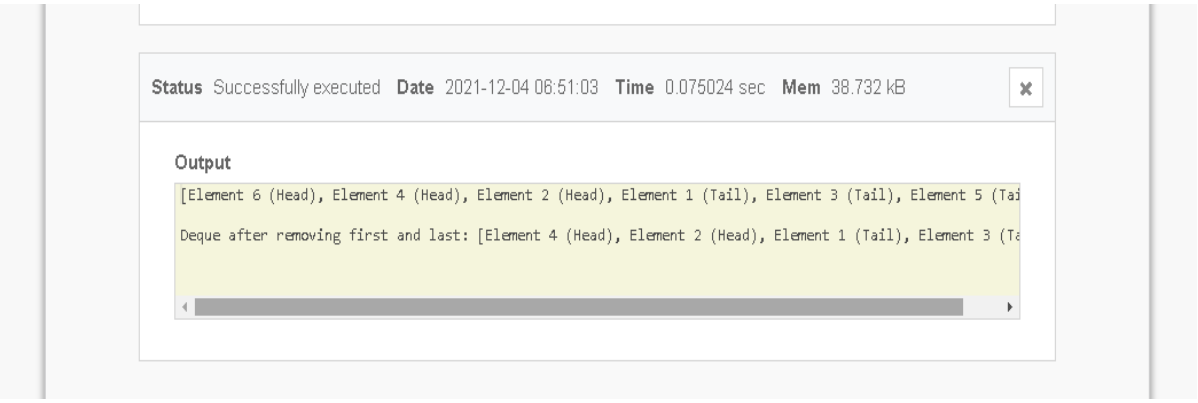
OUTPUT:


Status Successfully executed Date 2021-12-04 06:46:07 Time 0.131315 sec Mem 37.952 kB
Output
10
10
15

DEQUE:

```
import java.util.*;

public class DequeExample {
    public static void main(String[] args)
    {
        Deque<String> deque
            = new LinkedList<String>();

        // We can add elements to the queue
        // in various ways

        // Add at the last
        deque.add("Element 1 (Tail)");

        // Add at the first
        deque.addFirst("Element 2 (Head)");

        // Add at the last
        deque.addLast("Element 3 (Tail)");

        // Add at the first
        deque.push("Element 4 (Head)");

        // Add at the last
        deque.offer("Element 5 (Tail)");

        // Add at the first
        deque.offerFirst("Element 6 (Head)");

        System.out.println(deque + "\n");

        // We can remove the first element
        // or the last element.
        deque.removeFirst();
        deque.removeLast();
        System.out.println("Deque after removing "
                           + "first and last: "
                           + deque);
    }
}
```

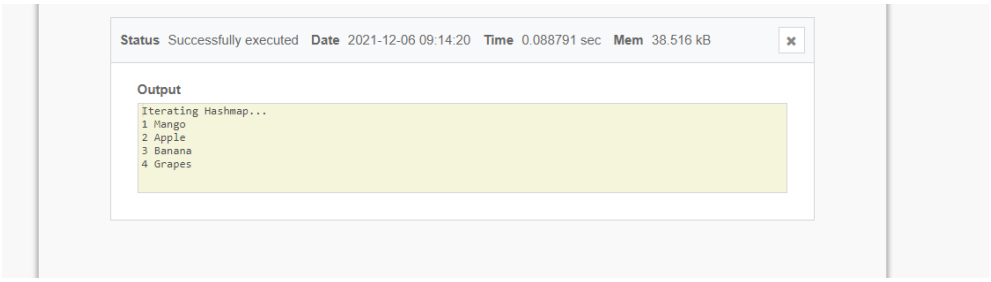
OUTPUT:


Status Successfully executed Date 2021-12-04 08:51:03 Time 0.075024 sec Mem 38.732 kB
Output
[Element 6 (Head), Element 4 (Head), Element 2 (Head), Element 1 (Tail), Element 3 (Tail), Element 5 (Tai
Deque after removing first and last: [Element 4 (Head), Element 2 (Head), Element 1 (Tail), Element 3 (Ta

HASH MAP();

```
import java.util.*;
public class HashMapExample1{
 public static void main(String args[]){
   HashMap<Integer,String> map=new
HashMap<Integer,String>();//Creating HashMap
   map.put(1,"Mango");  //Put elements in Map
   map.put(2,"Apple");
   map.put(3,"Banana");
   map.put(4,"Grapes");

   System.out.println("Iterating Hashmap...");
   for(Map.Entry m : map.entrySet()){
    System.out.println(m.getKey()+" "+m.getValue());
   }
}
}
```

OUTPUT:



LINKEDHASH MAP()

```
import java.util.LinkedHashMap;

class Main {
    public static void main(String[] args) {
        // Creating a LinkedHashMap of even numbers
        LinkedHashMap<String, Integer> evenNumbers = new
LinkedHashMap<>();
        evenNumbers.put("Two", 2);
        evenNumbers.put("Four", 4);
        System.out.println("LinkedHashMap1: " +
evenNumbers);

        // Creating a LinkedHashMap from other
LinkedHashMap
        LinkedHashMap<String, Integer> numbers = new
LinkedHashMap<>(evenNumbers);
        numbers.put("Three", 3);
        System.out.println("LinkedHashMap2: " + numbers);
    }
}
```

OUTPUT: