Q. Generic

Q.1 Write a generic method to count the number of elements in a collection that have a specific property (for example, odd integer, even number)

i/p : { 2, 4, 6, 7, 8, 9, 90, 78, 41, 56, 79, 45, 65, 85 }

output :

```
even : 7
odd : 7
```

→

```
import java.io.*;
import java.util.*;
class Algorithm {
public static <T> int CountIf (Collectio
    <T> C, UnaryPredicate <T> p){
  int count = 0;
  for( T elem : c)
    if ( p.test (elem))
       ++ count;
    return count;
 }
}
interface UnaryPredicate <T> {
 public boolean test (T obj);
}
class OddPredicate implements unaryPre
dicate < Integer> {
    < Integer> {
   public boolean  test (Integer i) {
   return i % 2 != 0; }
 }
Class EvenPredicate implements UnaryPredic-
 ate < Integer> {
 Public boolean test (Integer i)
 { return i % 2 == 0 ;
  }
 }


 Public class newx2 {
  Public static void main ( String args[])
{
collection < Integer> ci = Arrays. asList (1,2,3,
                            4);
 int count = Algorithm. countIf (ci, new
                    oddPredicate());
 int count2 = Algorithm. countIf (ci, new
                    EvenPredicate());
System. out. println ("number of odd
              Integers =" + count);
System. out. println ("number of Even
              Integer =" + count2);
 }
}
```

Q.2 Write a generic method to find the maximal element in the range [begin end] of a list.

i/p : {2, 62, 4, 78, 6, 10, 49, 20, 59, 43, 29, 30, 56, 89}

out : 89

→

```
import java.io.*;
import java.util.*;
class newex{
~~Public static void main~~ (string args[])
Public static < T extends Object &
  Comparable < ? super T >> T
  getMax (List < ? extends T > list,
  int begin, int end)
  {
T maxelem = list.get (begin);
for (++begin; begin < end ; ++begin)
 if ( maxelem. comparTo (list.get
                   (begin)) < 0)
  maxelem = list.get (begin);
return maxelem;
}
Public static void main (string args[])
{
List <Integer> arr = Arrays.asList (2, 62,
4, 78, 6, 10, 49, 20, 59, 43, 29, 30, 56, 89);
int x = newex. getMax (arr, 0, arr.
                 size());
system.out.println(" maximal number:"
                         + x) ;
  }
}
```