EXPERIMENT 8

PART A

A.1 AIM: -

a) Implement a static generic method PairUtil.swap whose parameter is a Pair object, using the generic class having a method swap to the pair class so that both values have same type that is used to swap the first and second element of the pair. The method should return a new pair, with the first and second element swapped.

b) Write a static generic method PairUtil.minmax that computes the minimum and maximum elements of an array of type T and returns a pair containing the minimum and maximum value.

c) Write a Java Program to compare two array list.

d) Write a Java program to display the elements and their positions in the linked list.

e) Write a java program to check if a particular element exits in the linked list.

A.2 Prerequisite
C Programming

A.3 Outcome
After successful completion of this experiment students will be able to add stability to the code by making bugs detectable at runtime using Generics.

A.4 Theory
Generics:

- Collections can store Objects of any Type
- Generics restricts the Objects to be put in a collection
- Generics ease identification of runtime errors at compile time

Collections:

The **Collection in Java** is a framework that provides an architecture to store and manipulate the group of objects.

Java Collections can achieve all the operations that you perform on a data such as searching, sorting, insertion, manipulation, and deletion.

Java Collection means a single unit of objects. Java Collection framework provides many interfaces (Set, List, Queue, Deque) and classes (ArrayList, Vector, LinkedList, PriorityQueue, HashSet, LinkedHashSet, TreeSet).

PART B

(PART B: TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)

Roll No.: N036	Name: Nischaya Sharma
Program: MBATechCS	Division: B
Semester: 3	Batch : B2
Date of Experiment: 28/9/19	Date of Submission: 28/9/19
Grade :	

B.1 Software Code written by student:

1)

```
package Pair;
import java.util.*;
public class PairUtil {
   public static void main(String[] args)
   {
      int a,b;
      System.out.println("Enter the integers");
      Scanner sc = new Scanner(System.in);
      a = sc.nextInt();
      b = sc.nextInt();
     pair <Integer> p= new pair <Integer>(a,b);
     System.out.println("Before swapping a= "+p.getFirst()+" b= "+p.getSecond());
     p.PairSwap();
     System.out.println("After swapping a= "+p.getFirst()+" b= "+p.getSecond());
   }


   }
class pair<T>
{
 private T a;
 private T b;
 public pair (T c, T d)
 {
  a=c;
  b=d;
 }
 T getFirst()
 {
```

```
  return a;
 }
 T getSecond()
 {
  return b;
 }
 void PairSwap()
 {
  T temp=a;
  a=b;
  b=temp;
 }
}
```

2)

```
import java.util.*;

public class pairminmax
{
   public static void main(String args[])
   {
      int a, i;
      System.out.println("Enter the 5 elements Nischaya N036");
      Scanner sc = new Scanner(System.in);
      List<Integer> list = new ArrayList<Integer>();
      for(i=5;i>=0;i--)
      {
         a = sc.nextInt();
        list.add(a);
      }

      System.out.println("List: " + list);
      int minList = Collections.min(list);
      int maxList = Collections.max(list);
      System.out.println("Minimum value of list is: " + minList);
      System.out.println("Maximum value of list is: " + maxList);
   }
}
```

3)

```
import java.util.Comparator;
import java.util.*;
import java.util.ArrayList;
import java.util.List;

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

```
  {
     int a,b,c,i;
           System.out.println("Enter the 5 elements of first list Nischaya N036");
       Scanner sc = new Scanner(System.in);
       ArrayList<Integer> list1 = new ArrayList<Integer>();
       for(i=5;i>0;i--)
       {
           a = sc.nextInt();
          list1.add(a);
       }
       System.out.println("Enter the 5 elements of second list Nischaya N036");

       ArrayList<Integer> list2 = new ArrayList<Integer>();
       for(i=5;i>0;i--)
       {
           b = sc.nextInt();
          list2.add(b);
       }

         ArrayList<Integer> list3 = new ArrayList<Integer>();

//compareTo method does not work in NetBeans or Online compiler. Unable to dereference
pointer.

         for (int e : list1)
            list3.add(list2.contains(e) ? 0 : 1);
          System.out.println(list3);
  }
```

4)

```
import java.util.*;
import java.util.LinkedList;
import java.util.Iterator;
import java.util.Scanner;
  public class Main {
  public static void main(String[] args) {
     LinkedList<String> newList = new LinkedList<String>();
     int a,i,j;
     System.out.println("Enter the number of elements Nischaya");
     Scanner sc = new Scanner(System.in);
     a = sc.nextInt();
     System.out.println("Enter the elements N036");
     String s;
     for(i = 0; i<=a;i++ )
     {
       s = sc.nextLine();
       newList.add(s);
```

```
    }

   System.out.println("Original linked list:" + newList);
  for(i=0; i <= a; i++)
   {
     System.out.println("Element at index "+i+": "+newList.get(i));
    }
 }
}
```

5)

```
import java.util.LinkedList;
import java.util.Scanner;
  public class Main {
  public static void main(String[] args) {
     LinkedList<String> newList = new LinkedList<String>();
     int a,b=1,i,j;
     System.out.println("Enter the number of elements Nischaya");
     Scanner sc = new Scanner(System.in);
     a = sc.nextInt();
     System.out.println("Enter the elements N036");
     String s,z;
     for(i = 0; i<=a;i++ )
     {
       s = sc.nextLine();
       newList.add(s);
     }
     while(b!=0)
     {
         z ="The Phantom of the Opera is HHHEEERRRRREEEEE";
         System.out.println("Enter element to search");
         z = sc.next();

         if (newList.contains(z))
         {
             System.out.println("Element exists");

         }
         else
         {
             System.out.println("Element does not exist");
         }

       System.out.println("Press 0 to exit, 9 to continue searching Nischaya");
       b = sc.nextInt();

     }
```

```
 }
}
```

B.2 Input and Output:

1)

```
Output - JavaApplication106 (run)
run:
Enter the integers
5
9
Before swapping a= 5 b= 9
After swapping a= 9 b= 5
BUILD SUCCESSFUL (total time: 16 seconds)
```

2)

```
Output - JavaApplication106 (run)
run:
Enter the 5 elements Aditya N027
4
6578
33333333
987
1
54
List: [4, 6578, 33333333, 987, 1, 54]
Minimum value of list is: 1
Maximum value of list is: 33333333
BUILD SUCCESSFUL (total time: 13 seconds)
```

3)

```
CompareLists > main >
utput - JavaApplication2 (run) #2
Enter the 5 elements of first list Aditya N027
1
2
3
45
5
Enter the 5 elements of second list Aditya N027
1
2
3
4
5
[0, 0, 0, 1, 0]
BUILD SUCCESSFUL (total time: 9 seconds)
```

4)

```
Main > main >
utput - JavaApplication2 (run) #2 ×
run:
Enter the number of elements Aditya
5
Enter the elements N027
as
sd
we
qw
zx
Original linked list:[, as, sd, we, qw, zx]
Element at index 0:
Element at index 1: as
Element at index 2: sd
Element at index 3: we
Element at index 4: qw
Element at index 5: zx
BUILD SUCCESSFUL (total time: 10 seconds)
```

5)

```
Find:  Previous  Next  Select
Main > main > while (b != 0) >
Output - JavaApplication2 (run) #4 ×
run:
Enter the number of elements Aditya
4
Enter the elements N027
as
Castle
Mr.Robot
Friends
Enter element to search
B99
Element does not exist
Press 0 to exit, 9 to continue searching Aditya
9
Enter element to search
Castle
Element exists
Press 0 to exit, 9 to continue searching Aditya
0
BUILD SUCCESSFUL (total time: 38 seconds)
```

B.3 Conclusion:

After successful completion of this experiment I am able to add stability to the code by making bugs detectable at runtime using Generics.