ASSIGNMENT 3
Name : Hrithvik Kondalkar
Roll : 002211001088

1. Write a generic method in Java that takes an array of any data type and sorts the array in ascending order using any sorting algorithm.

```
import java.util.Arrays;
class Main{
   public static <E extends Comparable<E>> void sort(E[] array){
      int length = array.length;
      E temp;
      //bubble sort
      for(int i=0;i<length;i++){
         for(int j=1;j<length-i;j++){
            if(array[j-1].compareTo(array[j])>0){
               temp = array[j-1];
               array[j-1] = array[j];
               array[j] = temp;
            }
         }
         System.out.println(Arrays.toString(array));
      }
   }

   public static void main(String[] args){
      Integer[] arr = new Integer[]{5,3,4,2,1};
      System.out.println(Arrays.toString(arr));
      sort(arr);
   }
}
```

```
[be2288@localhost 1]$ java Main
[5, 3, 4, 2, 1]
[3, 4, 2, 1, 5]
[3, 2, 1, 4, 5]
[2, 1, 3, 4, 5]
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
```

2. Write a generic method in Java that takes any type of an array as input and finds the frequency of each data element.

```
import java.util.HashMap;
import java.util.Map;
class Main{
        public static <E> void freq(E[] array){
                HashMap<E,Integer> map = new HashMap<>();
                for(E elem:array){
                        if(map.containsKey(elem)){
                        map.replace(elem,map.get(elem)+1);
                        }
                        else{
                        map.put(elem,1);
                        }
                }
                for(Map.Entry m:map.entrySet()){
                        System.out.println(m.getKey()+" = "+m.getValue());
                }
        }
        public static void main(String[] args){
        Integer[] array = new Integer[]{2,1,4,2,3,4,5,1,2};
        freq(array);
        }
}
```

```
[be2288@localhost 2]$ java Main
1 = 2
2 = 3
3 = 1
4 = 2
5 = 1
```

3. Design a generic Java class having a method that takes an array of any data type and prints all the duplicate elements.

```
import java.util.HashMap;
import java.util.Map;
class Main{
	public static <E> void dupl(E[] array){
		HashMap<E,Boolean> map = new HashMap<>();
		for(E elem:array){
			if(map.containsKey(elem)){
			map.replace(elem,true);
			}
			else{
			map.put(elem,false);
			}
		}
		System.out.print("Duplicate Elements : ");
		for(Map.Entry m:map.entrySet()){
			if((boolean)m.getValue()){System.out.print(m.getKey()+ " ");}
		}
	}

	public static void main(String[] args){
	Integer[] array = new Integer[]{2,1,4,2,3,4,5,1,2};
	dupl(array);
	}
}
```

```
[be2288@localhost 3]$ java Main
Duplicate Elements : 1 2 4 [be22
```

4. Test the functionalities of different java reflection APIs such as getClass(), getMethods(), getConstructors(), getDeclaredMethod(), getDeclaredField(), setAccessible() etc.

```
import java.lang.reflect.*;
class Test{
	public int field1;
	private int field2;

	public Test(int a, int b){
		field1 = a;
		field2 = b;
	}
	public void print(){
		System.out.println("field1 = "+field1);
		System.out.println("field2 = "+field2);
	}
	public void method1(){
		System.out.println("method1 called");
	}

	private void method2(){
		System.out.println("method2 called");
	}
}
class Main{
	public static void main(String[] a)throws Exception{
		Test obj = new Test(1,2);
		Class c = obj.getClass();
		System.out.println(c);
		System.out.println("-----------------------------------");

		Method[] m = c.getMethods();
		for(Method meth:m){System.out.println(meth);}
		System.out.println("-----------------------------------");

		Constructor[] cons = c.getConstructors();
		for(Constructor t:cons){System.out.println(t);}
		System.out.println("-----------------------------------");

		Field[] f = c.getFields();
		for(Field z : f){System.out.println(z);}
		System.out.println("-----------------------------------");

		Method m2 = c.getDeclaredMethod("method2");
		System.out.println(m2);
		m2.setAccessible(true);
```

```
                m2.invoke(obj);
                System.out.println("----------------------------------");

                Field f2 = c.getDeclaredField("field2");
                f2.setAccessible(true);
                System.out.println(f2 + " = " + f2.get(obj));
                System.out.println("----------------------------------");

        }
}
```

```
[be2288@localhost 4]$ java Main
class Test
-----------------------------------
public void Test.method1()
public void Test.print()
public final void java.lang.Object.wait(long,int) throws java.lang.InterruptedException
public final void java.lang.Object.wait() throws java.lang.InterruptedException
public final native void java.lang.Object.wait(long) throws java.lang.InterruptedException
public boolean java.lang.Object.equals(java.lang.Object)
public java.lang.String java.lang.Object.toString()
public native int java.lang.Object.hashCode()
public final native java.lang.Class java.lang.Object.getClass()
public final native void java.lang.Object.notify()
public final native void java.lang.Object.notifyAll()
-----------------------------------
public Test(int,int)
-----------------------------------
public int Test.field1
-----------------------------------
private void Test.method2()
method2 called
-----------------------------------
private int Test.field2 = 2
-----------------------------------
```