# Practical No. 01

**Q1.Develop a Java Program to demonstrate a generic concept with generic method to count occurrences in an array.**

**Code:**

```
public class CountOccurence {
	public static <T> int count(T[] array, T element)
	{
		int count=0;
		for (T t : array) {
			if (t.equals(element)) {
			count++;
			}
		}
		return count;
	}
	public static void main(String[] args) {
		// TODO Auto-generated method stub
	    Integer[] intArray = {1, 2, 2, 3, 2};
	    int find = 2;
	    int countInt = count(intArray, find);
	    System.out.println("The element " + find + " appears " + countInt + "
times in the Integer array.");
	}
}
```

**Output:**

```
The element 2 appears 3 times in the Integer array.
```

**Q2.Write a Java program using Lambda Expression with single parameters.**

**Code:**

```
interface example{
	double Area(double l,double b);
}
public class LamdaExample {
	public static void main(String[] args) {
		example objL=(l,b)-> l*b;
		System.out.println("area is "+objL.Area(10, 20));
		}
}
```

**Output:**

```
area is 200.0
```

**Q3.Create a program that demonstrates the use of the generic Stack class with different data types, such as integers, strings, and custom objects.**

**Code:**

```
import java.util.Stack;
public class StackDemo {
	String name;
	public StackDemo(String name) {
		super();
		this.name = name;
	}
	@Override
   public String toString() {
      return name;
   }
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Stack<Integer> stk=new Stack<Integer>();
		System.out.println("Empty Stack"+stk);
		System.out.println("is Empty: "+stk.isEmpty());
		stk.push(101); stk.push(102);stk.push(103);
		System.out.println("is Empty: "+stk);
		System.out.println("pop operation :"+stk.pop());
		System.out.println("is Empty: "+stk);
		System.out.println(stk.search(102));
		System.out.println("");
		//String
		Stack<String> stk1=new Stack<String>();
		System.out.println("Empty Stack"+stk1);
		System.out.println("is Empty: "+stk1.isEmpty());
		stk1.push("Guarav"); stk1.push("Pritesh");stk1.push("Vivek");
		System.out.println("is Empty: "+stk1);
		System.out.println("pop operation :"+stk1.pop());
		System.out.println("is Empty: "+stk1);
		System.out.println("");
```

```
        //custom object
        Stack<StackDemo> objectStack = new Stack<>();
        System.out.println("is Empty: "+objectStack.isEmpty());
        objectStack.push(new StackDemo("Sahil"));
        objectStack.push(new StackDemo("Vishal"));
        System.out.println(objectStack);
    }}
```

**Output:**

```
Empty Stack[]
is Empty: true
is Empty: [101, 102, 103]
pop operation :103
is Empty: [101, 102]
1

Empty Stack[]
is Empty: true
is Empty: [Guarav, Pritesh, Vivek]
pop operation :Vivek
is Empty: [Guarav, Pritesh]

is Empty: true
[Sahil, Vishal]
```

**Q4. Develop a Java Program to demonstrate Java Generics for user defined class – College with members like generic aicteNo, String clgNm, double intake, Sting clgAdd.**

**Code:**

```
public class College<T>{
    T aicteNo;
    String clgNm;
    double intake;
    String clgAdd;

    public College(T aicteNo, String clgNm, double intake, String clgAdd) {
        super();
        this.aicteNo = aicteNo;
        this.clgNm = clgNm;
```

```
		this.intake = intake;
		this.clgAdd = clgAdd;
	}
	public void display() {
		System.out.println("college aictNo is "+aicteNo+" of "+clgNm+"
intake student is "+intake+" student at department "+clgAdd);
	}

	T getOb() {
		return aicteNo;
	}
	void showType() {
		System.out.println("Type of T is"+getOb().getClass().getName());
	}

	@Override
	public String toString() {
		return "College [aicteNo= " + aicteNo + ", clgNm= " + clgNm + ",
intake= " + intake + ", clgAdd= " + clgAdd + "]";
	}
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		College<Integer> objCollege=new College<Integer>(101, "Famt",
120, "MCA");
		objCollege.display();
		System.out.println(objCollege.toString());;
		objCollege.showType();
	}
}
```

**Output:**

```
college aictNo is 101 of Famt  intake student is 120.0 student at department MCA
College [aicteNo= 101, clgNm= Famt, intake= 120.0, clgAdd= MCA]
Type of T isjava.lang.Integer
```

**Q5. Create a program that demonstrates the use of the generic HashMap class that accepts objects of College class (created in question 4) with aicteNo as key and perform various operations.**

**Code:**

**1] College.java**

```
public class College<T>{
```

```
	T aicteNo;
	String clgNm;
	double intake;
	String clgAdd;

	public College(T aicteNo, String clgNm, double intake, String clgAdd) {
		super();
		this.aicteNo = aicteNo;
		this.clgNm = clgNm;
		this.intake = intake;
		this.clgAdd = clgAdd;
	}
	public void display() {
		System.out.println("college aictNo is "+aicteNo+" of "+clgNm+"
intake student is "+intake+" student at department "+clgAdd);
	}

	T getOb() {
		return aicteNo;
	}
	void showType() {
		System.out.println("Type of T is"+getOb().getClass().getName());
	}

	@Override
	public String toString() {
		return "College [aicteNo= " + aicteNo + ", clgNm= " + clgNm + ",
intake= " + intake + ", clgAdd= " + clgAdd + "]";
	}
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		College<Integer> objCollege=new College<Integer>(101, "Famt",
120, "MCA");
		objCollege.display();
		System.out.println(objCollege.toString());;
		objCollege.showType();
	}
}
```

**HashMap.java**

```
import java.util.HashMap;
import java.util.Map;

public class HashMap {
```

```
	public static void main(String[] args) {
		Map<Integer, College<Integer>> collegeMap = new HashMap<>();

		// Creating some College objects
		College<Integer> college1 = new College<>(101, "Famt", 120,
"MCA");
		College<Integer> college2 = new College<>(102, "NMIT", 200,
"CSE");
		College<Integer> college3 = new College<>(103, "VIT", 150, "IT");

		// Adding College objects to the HashMap
		collegeMap.put(college1.aicteNo, college1);
		collegeMap.put(college2.aicteNo, college2);
		collegeMap.put(college3.aicteNo, college3);

		System.out.println("Hasmap:\n"+college1);
		System.out.println("Hasmap:\n"+college2);
		System.out.println("Hasmap:\n"+college3);

	}

}
```

**Output:**

```
Hasmap:
College [aicteNo= 101, clgNm= Famt, intake= 120.0, clgAdd= MCA]
Hasmap:
College [aicteNo= 102, clgNm= NMIT, intake= 200.0, clgAdd= CSE]
Hasmap:
College [aicteNo= 103, clgNm= VIT, intake= 150.0, clgAdd= IT]
```