Collections

Set-A

1) **Write a java program to accept names of "n" cities , insert same into arraylist collection and display the contents of same arraylist, also remove all these elements.**

```
import  java.util.*;
import java.io.*;
public class ArrayListDemo
{
public static void main(String args[])throws Exception
{
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
ArrayList al=new ArrayList();
System.out.println("\nHow many City?");
Int n =Integer.parseInt(br.readLine());
System.out.println("\n Enter City names:");
for(int i=;i<=n;i++)
    {
    al.add(br.readLine());
    }
System.out.println("Entered cities are :"+al);
al.removeAll(al);
System.out.println("All cities are removed from the ArrayList:"+a);
}
}
```

2**)Write a java program to read 'n' names of your friends , store it into linked list , also display contents of the same.**

```
import java.util.*;

import java.io.*;

public class LinkedListDemo

{

Public static void main(String args[])throws Exception

{

int n;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

LinkedList li=new LinkedList();

System.out.println("\Enter number of your friends:");

n=Interger.parseInt(br.readLine()));

System.out.println("\Enter your friends names:");

for(int i=1;i<=n;i++)

{

li.add(br.readLine());

}

System.out.println("\Linked List content is:");

}

}
```

3) Write a program to create a new tree set , add some colors(String)and print out the tree set.

```
import java.util.*;

class TreeSetDemo

{

Public static void main(String args[])

{

TreeSet ts=new TreeSet();

ts.add(“Red”);

ts.add(“Yellow”);

ts.add(“Blue”);

ts.add(“Black”);

ts.add(“Orange”);

ts.add(“Green”);

System.out.println(“TreeSet is:”+ts);

}

}
```

4) Create the hastable that will maintain the mobile number and student name. Display the contact list .

```
import java.util.*;
import java.io.*;
class HashTableDemo
{
Public static void main(String args[])
{
Hashtable ht= new Hashtable();
Enumeration names;
String str;
ht.put(“Amar”,new Long(2222222222L));
ht.put(“Anil”,new Long(3333333333L));
ht.put(“Soham”,new Long(8888888888L));
ht.put(“Kiran”,new Long(9999999999L));
ht.put(“Williams”,new Long(4444444444L));
names=ht.keys();
while(names.hasMoreElements());
{
str= (String) names.nextElement();
System.out.println(str+”:”+ht.get(str));
}
```

```
}
}
```

SET-B-1)

*** ACCEPT 'n' INTEGERS FROM THE USER AND STORE THEM IN A COLLECTION.**

*** DISPLAY THEM IN THE SORTED ORDER.**

*** THE COLLECTION SHOULD NOT ACCEPT DUPLICATE ELEMENTS.**

*** (USE SUTABLE COLLECTION)**

```
 * */
import java.io.*;
import java.util.*;
public class ArrayListA1
{
public static void main(String args[])throws Exception
    {
        ArrayList<Integer> s=new ArrayList<Integer>();
        DataInputStream dis=new DataInputStream(System.in);
        System.out.print("how many elements you want to store = ");
        int n=Integer.parseInt(dis.readLine());
        for(int i=1;i<=n;i++)
        {
            System.out.print("Enter the num = ");
            int num=Integer.parseInt(dis.readLine());
            Integer numOb=new Integer(num);
```

```
                    if(s.contains(numOb))
                    {
                        System.out.println(num+" is  already present in the collection.....");
                        i--;

                    }
                    else
                    {
                        s.add(new Integer(numOb));
                    }
                }
                System.out.println("\nDISPLAYING ELEMENTS BEFORE SORT= "+s);
                Collections.sort(s);
                System.out.println("\nACEESSING ELEMENTS SEPARATLY\n");
                Iterator ir=s.iterator();
                while(ir.hasNext())
                {
                    System.out.println("Element = "+ir.next());
                }
        }
}
```

2) Write a program to sort HashMap by keys and display the details before sorting and after sorting.

```
import java.util.*;

class HashMapDemo

{

Public static void main(String args[])

{

HashMap <Interger,String> ha=new HashMap<Interger,String>();

ha.put(10,”Java”);

ha.put(20,”Operating System”);

ha.put(30,”SoftwareTesting Tools”);

ha.put(40,”Complier Constructor”);

ha.put(50,”Web Technologies”);

System.out.println(          “Before Sorting:”);

Set set=ha.entrySet();

Iterator it=set.iterator();

While(it.hasNext())

{

Map.Entry  me=(Map.Entry)it.next();

System.out.print(me.getKeys()+”:”);
```

```
System.out.println(me.getValues());
}
 Map<Integer,String> map=new HashMap<Integer,String>(hm);
System.out.println("After Sorting:");
Set set1=map.entrySet();
Iterator it2=set2.it();
While(it2.hasNext())
{
Map.Entry me2=(Map.Entry)it2.next();
System.out.print(me2.getKeys()+":");
System.out.prinln(me2.getValues());
}
}
}
```

Set-B-3)

```
import java.util.*;
import java.io.*;
public class Phonebook
{
Public static void main(String args[])
{
Try
{
FileInputStream fis=new FileInputStream(“home/desktop/myfile.txt”);
Scanner(fis).useDelimiter(“\t”);
Hashtable<String,String> ht=new Hashtable<String,String>();
String [] strarray;
String a,str;
While(sc.hasNext())
{
a=sc.nextLine();
strarray=a.split(\t”);
ht.put(strarray[0],strarray[1]);
```

```
System.out.println(“Hashtable values are:”+strarray[0]+”:”+strarray[1]);
Scanner s=new Scanner(System.in);
System.out.println(“Enter the name as given in the phone book”);
Str=s.next();
If(ht.containsKey(str))
{
System.out.println(“Phone no is:”+ht,get(str));
}
else
{
System.out.println(“Name is not Matching with the phone book”);
}
}
Catch(Exception e)
{
System.out.println(e);
}
}
}
```

Myfile.txt

Akshay 123

Akash 235

Soham 569

Kiran 456

Multithreading

SETA

1) Program to define a thread for printing text on output screen for n number of times. Create 3 threads and run them. Pass the text 'n' parameters to the thread constructor.

1) First thread prints "COVID19" 10 times.
2) Second thread prints "LOCKDOWN2020" 20 times.
3) Third thread prints "VACCINATED2021" 30 times

```
Import java.io.*;

Import java.lang.String.*;

Class TestPrint extends Thread

{

String msg="";

int n;

TestPrint(String msg,int n)

{

this.msg=msg;

this.n=n;

}

Public void run()
```

```
{
Try
{
For(int i=1;i<=n;i++)
{
System.out.println(msg+" "+i+"times");
}
System.out.println("\n");
}
Catch(Exception e)
{
}}}
Class DemoMythread
{
Public static void main(String args[]);
{
Int n = Integer.parseInt(args[0])
TestPrint t1=new TestPrint("COVID",n);
T1.start();
TestPrint t2=new TestPrint("LOCKDOWN2020",n+10);
T2.start();
TestPrint t3=new TestPrint("Vaccinated2021",n+20);
T3.start();
}
}
```

2) Write a program in which thread sleep for 6 sec in the loop in reverse order from 100 to 1 and change the name of thread.

```
class MythreadDemo
  {
    public static void main(String args[])
    {
      Thread t=Thread.currentThread();
      System.out.prinltn("Current Thread is:"+t);
      try
      {
        for(int n=100;n>0;n--)
        {
          System.out.println(n);
          Thread.sleep(6000);
       }
            t.setName("Demo Thread");
              System.out.prinltn("After changing the name thread is :"+t);
```

```
    }
            catch(InterruptedException e)
            {
            System.out.println("Thread interrupted");
                    }
    }}
```

3) Write a program to solve producer consumer problem in which a producer produces a value and consumer consume the value before producer generate the next value.(Hint: Use thread synchronization).

```
import java.util.LinkedList;

public class Threadexample
{
   public static void main(String[] args) throws InterruptedException
   {
      // Object of a class that has both produce()
      // and consume() methods
      final PC pc = new PC();

      // Create producer thread
      Thread t1 = new Thread(new Runnable()
         {
         @Override
         public void run()
         {
            try
            {
               pc.produce();
            }
            catch (InterruptedException e)
            {
               e.printStackTrace();
            }
         }
      });

      // Create consumer thread
      Thread t2 = new Thread(new Runnable()
```

```
    {
        @Override
        public void run()
        {
            try
            {
                pc.consume();
            }
            catch (InterruptedException e)
            {
                e.printStackTrace();
            }
        }
    });

    // Start both threads
    t1.start();
    t2.start();

    // t1 finishes before t2
    t1.join();
    t2.join();
}

// This class has a list, producer (adds items to list
// and consumer (removes items).

        public static class PC
        {

    // Create a list shared by producer and consumer
    // Size of list is 2.
    LinkedList<Integer> list = new LinkedList<>();
    int capacity = 2;

    // Function called by producer thread
    public void produce() throws InterruptedException
    {
        int value = 0;
        while (true)
        {
            synchronized (this)
            {
                // producer thread waits while list
                // is full
                while (list.size() == capacity)
```

```
                wait();

                System.out.println("Producer produced-"+ value);

                // to insert the jobs in the list
                list.add(value++);

                // notifies the consumer thread that
                // now it can start consuming
                notify();

                // makes the working of program easier
                // to  understand
                Thread.sleep(1000);
            }
        }
    }

    // Function called by consumer thread
    public void consume() throws InterruptedException
    {
        while (true)
        {
            synchronized (this)
            {
                // consumer thread waits while list
                // is empty
                while (list.size() == 0)
                    wait();

                // to retrieve the first job in the list
                int val = list.removeFirst();

                System.out.println("Consumer consumed-" + val);

                // Wake up producer thread
                notify();

                // and sleep
                Thread.sleep(1000);
            }
        }
    }
}
```

SET-B

1) Write a program to calculate the sum and average of an array of 1000 integers (genearated randomly) using 10 threads. Each thread calculates the sum of 100 integers . Use these values to calculate average(Use join method).

```
import java.util.*;
class thread implements Runnable
{
	Thread t;
		int i,no,sum;
		int a[]=new int[1000];
		thread(String s,int n)
		{
			Random rs = new Random();
				t=new Thread(this,s);
				no=n;
				int j=0;
				for(i=1;i<=1000;i++)
				{
					a[j]=rs.nextInt()%100;;
						j++;
				}
			t.start();
		}
	public void run() {
		for(i=0;i<100;i++)
		{
			sum=sum+a[no];
				no++;
		}
		System.out.println("Sum = "+sum);
		System.out.println("Avg ="+sum/100);
```

```
	}

}
public class threaddemo
{
	public static void main(String[] arg) throws InterruptedException
	{
		thread t1=new thread("g",1);
			t1.t.join();
			thread t2=new thread("r",100);
			t2.t.join();
			thread t3=new thread("s",200);
			t3.t.join();
			thread t4=new thread("t",300);
			t4.t.join();
			thread t5=new thread("p",400);
			t5.t.join();
		thread t6=new thread("p",500);
			t5.t.join();
		thread t7=new thread("p",600);
			t5.t.join();
		thread t8=new thread("p",700);
			t5.t.join();
		thread t9=new thread("p",800);
			t5.t.join();
		thread t10=new thread("p",900);
			t5.t.join();

	}
}
```

3) Write a java program that implements a multithread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.

```
import java.util.Random;
```

```
class RandomNumberThread extends Thread
{
public void run()
{
Random random = new Random();
for (int i = 0; i &lt; 10; i++) {
int randomInteger = random.nextInt(100);
System.out.println(&quot;Random Integer generated : &quot; + randomInteger);
if((randomInteger%2) == 0) {
SquareThread sThread = new

SquareThread(randomInteger);


sThread.start();
}
else {
CubeThread cThread = new CubeThread(randomInteger);
cThread.start();
}
try {
Thread.sleep(1000);
}
catch (InterruptedException ex) {
System.out.println(ex);
```

```
}

}

}

}
```

JDBC

SetA

1) Create a PROJECT table with fields project_id, Project_name, Project_description, Project_Status. etc. Insert values in the table. Display all the details of the PROJECT table in a tabular format on the screen.

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;

public class Project {

	public static void main(String[] args) {
		Connection con = null;
		PreparedStatement ps=null;
		Statement stmt=null;
		BufferedReader br = new BufferedReader( new
InputStreamReader(System.in));
		try
		{
			Class.forName("org.postgresql.Driver");

	con=DriverManager.getConnection("jdbc:postgresql:test","postgres","Pass@word");
				stmt=con.createStatement();
				//ResultSet rs=stmt.executeQuery("create table
PROJECT(project_id int primary key,project_name varchar(30),project_desciption
varchar(50),project_status varchar(20))");
```

```
                    System.out.print("Enter Project ID:");
                    int id=Integer.parseInt(br.readLine());

                    System.out.print("Enter Project Name:");
                    String name=br.readLine();

                    System.out.print("Enter Project Description:");
                    String desc=br.readLine();

                    System.out.print("Enter Project Status");
                    String Status=br.readLine();

                    ps=con.prepareStatement("insert into PROJECT values(?,?,?,?)");
                    ps.setInt(1, id);
                    ps.setString(2, name);
                    ps.setString(3,desc);
                    ps.setString(4, Status);
                    int i=ps.executeUpdate();
                    if(i==0)
                            System.out.println("Unable to insert");
                    else
                            System.out.println("Data Inserted Succesfully");
                    con.close();
            }
            catch(Exception e)
            {
                    e.printStackTrace();
            }

    }

}
```

SETA2) Write a program to display information about the database and list all the tables in the database. (Use DatabaseMetaData).

```
import java.sql.*;
class DatabaseMetaDataTest1
{
public static void main(String args[]) throws Exception
{
Class.forName("org.postgresql.Driver");
Connection con=DriverManager.getConnection("jdbc:postgresql:tybcs","postgres","");
```

```
DatabaseMetaData dbmd=con.getMetaData();

System.out.println("Database Name="+dbmd.getDatabaseProductName());
System.out.println("Database Version="+dbmd.getDatabaseProductVersion());
System.out.println("Database Driver Name="+dbmd.getDriverName());
System.out.println("Driver Major Version"+dbmd.getDriverMajorVersion());
System.out.println("Driver Major Version="+dbmd.getDriverMinorVersion());

System.out.println("URL of Database="+dbmd.getURL());
System.out.println("Current UserName="+dbmd.getUserName());

System.out.println("=========Tables=========");
String t[]={"TABLE"};
ResultSet rs=dbmd.getTables(null,null,null,t);
while(rs.next())
{
System.out.println(rs.getString("TABLE_NAME"));
}
System.in.read();
System.out.println("=========VIEWS==========");
String v[]={"VIEW"};
rs=dbmd.getTables(null,null,null,v);
while(rs.next())
{
System.out.println(rs.getString("TABLE_NAME"));
}
con.close();
}
}
```

SETA 3) Write a program to display information about all columns in the DONAR table using

ResultSetMetaData.

```
2) import java.sql.*;
3) class ResultSetMetaDataTest
4) {
5) public static void main(String args[])throws Exception
6) {
7) Class.forName("org.postgresql.Driver");
8) Connection
   con=DriverManager.getConnection("jdbc:postgresql:test","postgres","Pass@word");
```

```
9)   Statement stmt=con.createStatement();
10) ResultSet rs=stmt.executeQuery("select *from DONAR");
11) ResultSetMetaData rsmd=rs.getMetaData();
12) int n= rsmd.getColumnCount();
13) System.out.println("No.of Columns="+n);
14) for(int i=1;i<=n;i++)
15) {
16) System.out.println("Column Number:"+i);
17) System.out.println("====================");
18) System.out.println("Column Name="+rsmd.getColumnName(i));
19) System.out.println("Column Type="+rsmd.getColumnTypeName(i));
20) System.out.println("Column Width="+rsmd.getColumnDisplaySize(i));
21) System.out.println("Column Presicion="+rsmd.getPrecision(i));
22) System.out.println("Is Currency="+rsmd.isCurrency(i));
23) System.out.println("Is Read-Only"+rsmd.isReadOnly(i));
24) System.out.println("Is Writable="+rsmd.isWritable(i));
25) System.out.println("Is Searchable="+rsmd.isSearchable(i));
26) System.out.println("Is Signed="+rsmd.isSigned(i));
27) }
28) con.close();
29) }
30) }
31)
```

SETB

1) Create a MOBILE table with fields Model_Number, Model_Name, Model_Color, Sim_Type, NetworkType, BatteryCapacity, InternalStorage, RAM and ProcessorType. Insert values in the table. Write a menu driven program to pass the input using Command line argument to perform the following operations on MOBILE table.

1. Insert 2. Modify 3. Delete 4. Search 5. View All 6. Exit

```
import java.sql.*;
import java.io.*;
public class MobileDemo
{
   public static void main(String[] args) throws Exception
   {
   Connection con;
   ResultSet rs;
   Statement t;
   PreparedStatement ps=null;
   BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
   Class.forName("org.postgresql.Driver");
   con=DriverManager.getConnection("jdbc:postgresql://localhost/tydb","root","root123");
   do
   {
   System.out.println("\n1.Insert\n2.Modify\n3.Delete\n4.Search\n5.View all\n6.Exit");
   System.out.println("Enter the choice");
   int ch=Integer.parseInt(br.readLine());
   switch(ch)
   {
      case 1:
         System.out.println("Enter the Modelno");
         int mno=Integer.parseInt(br.readLine());

         System.out.println("Enter the Modelname");
         String mname=br.readLine();

         System.out.print("Enter Model Color:");
         String mcolor=br.readLine();

         System.out.print("Enter Sim type:");
         String mstype = br.readLine();
```

```
System.out.print("Enter Network Type:");
String mntype=br.readLine();

System.out.print("Enter Battery Capacity:");
String capacity= br.readLine();

System.out.print("Enter Internal Storage:");
int storage=Integer.parseInt(br.readLine());

System.out.println("Enter the RAM");
int ram=Integer.parseInt(br.readLine());

System.out.print("Enter Processor Type:");
String ptype=br.readLine();

ps=con.prepareStatement("insert into MobileInfo values(?,?,?,?,?,?,?,?,?)");
                        ps.setInt(1, mno);
                        ps.setString(2, mname);
                        ps.setString(3,mcolor);
                        ps.setString(4, mstype);
                        ps.setString(5, mntype);
                        ps.setString(6, capacity);
                        ps.setInt(7, storage);
                        ps.setInt(8, ram);
                        ps.setString(9, ptype);
                        int i=ps.executeUpdate();
                        if(i==0)
                                System.out.println("Unable to insert");
                        else
                                System.out.println("Data Inserted Succesfully");
break;

case 2:
   System.out.println("Enter to Modify record");
System.out.println("Enter the Modelno");
 mno=Integer.parseInt(br.readLine());

System.out.println("Enter the Modelname");
 mname=br.readLine();

System.out.print("Enter Model Color:");
mcolor=br.readLine();

System.out.print("Enter Sim type:");
mstype = br.readLine();
```

```
System.out.print("Enter Network Type:");
 mntype=br.readLine();

System.out.print("Enter Battery Capacity:");
 capacity= br.readLine();

System.out.print("Enter Internal Storage:");
 storage=Integer.parseInt(br.readLine());

System.out.println("Enter the RAM");
ram=Integer.parseInt(br.readLine());

System.out.print("Enter Processor Type:");
 ptype=br.readLine();

ps=con.prepareStatement("insert into MobileInfo values(?,?,?,?,?,?,?,?,?)");
                        ps.setInt(1, mno);
                        ps.setString(2, mname);
                        ps.setString(3,mcolor);
                        ps.setString(4, mstype);
                        ps.setString(5, mntype);
                        ps.setString(6, capacity);
                        ps.setInt(7, storage);
                        ps.setInt(8, ram);
                        ps.setString(9, ptype);
                         i=ps.executeUpdate();
                        if(i==0)
                                System.out.println("Unable to insert");
                        else
                                System.out.println("Data Inserted Succesfully");
break;

case 3:
System.out.println("Enter the Model Number for delete record");
int no=Integer.parseInt(br.readLine());
ps=con.prepareStatement("delete from MobileInfo where model_number=?");
ps.setInt(1, no);
i=ps.executeUpdate();
                        if(i==0)
                                System.out.println("Unable to Delete");
                        else
                                System.out.println("Data deleted Succesfully");
break;


case 4:
```

```
            System.out.println("Enter the Model Number for search");
            no=Integer.parseInt(br.readLine());
            t=con.createStatement();
            rs=t.executeQuery("select * from MobileInfo where mno="+no);
            while(rs.next())
            {
               System.out.println("Model Number="+rs.getInt(1));
                System.out.println("Model Name="+rs.getString(2));
                System.out.println("Model Color="+rs.getString(3));
                System.out.println("Sim Type:" +rs.getString(4));
                System.out.println("Network Type=" +rs.getString(5));
                System.out.println("Battery Capacity=:" +rs.getString(6));
                System.out.println("Internal Strorage=" +rs.getInt(7));
                System.out.println("RAm=" +rs.getInt(8));
                System.out.println("Processor:" +rs.getString(9));
            }
            break;
        case 5:
            t=con.createStatement();
            rs=t.executeQuery("select * from MobileInfo");
            while(rs.next())
            {
               System.out.println("Model Number="+rs.getInt(1));
               System.out.println("Model Name="+rs.getString(2));
               System.out.println("Model Color="+rs.getString(3));
               System.out.println("Sim Type:" +rs.getString(4));
               System.out.println("Network Type=" +rs.getString(5));
               System.out.println("Battery Capacity=:" +rs.getString(6));
               System.out.println("Internal Strorage=" +rs.getInt(7));
               System.out.println("RAm=" +rs.getInt(8));
               System.out.println("Processor:" +rs.getString(9));
               }
            break;
        case 6:
            System.exit(0);
            break;
    }
    }while(true);
    }
}
```

SERVLET

SETA-1)

Design a servlet that provides information about HTTP request from a client such as IP address and browser type. The servlet also provides information about the server on which the servlet is running, such as the operation system type and the names of currently loaded servlets

serverInfo.java

```
 import java.io.*;
       import javax.servlet.*;
       import javax.servlet.http.*;
public class serverInfo extends HttpServlet implements Servlet
{
   protected void doGet(HttpServletRequest req,HttpServletResponse res)throws
IOException,ServletException
   {
       res.setContentType("text/html");
       PrintWriter pw=res.getWriter();
       pw.println("<html><body><h2>Information about Http Request</h2>");
       pw.println("<br>Server Name: "+req.getServerName());
       pw.println("<br>Server Port: "+req.getServerPort());
       pw.println("<br>Ip Address: "+req.getRemoteAddr());
//pw.println("<br>Server Path: "+req.getServerPath());       pw.println("<br>Client Browser:
"+req.getHeader("User-Agent"));
       pw.println("</body></html>");
       pw.close();
   }
}
```

Web.xml

```
<web-app>
<servlet>
<servlet-name>serverInfo</servlet-name>
<servlet-class>ServerInfo</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>serverInfo</servlet-name>
<url-pattern>/server</url-pattern>
</servlet-mapping>
</web-app>
```

SETB

1) **Design an HTML page which passes customer number to a search search. The Servlet searches for the customer number in a database(customer table) and returns customer details if found the number details if found the number otherwise display error message.**

Click on the file –New- Dynamic Web project-Project Name-Click on the next-next-Click on the Checkbok Generate Web.xml deployment descriptor-finish.
Right click on WebContent –new-HTML file-index.html

index.html

```
<!DOCTYPE html>
<html>
<head>
<title> Insert title here</title>
</head>
<body>
<form action="./servlet" >
<table>
    <tr>
<td> Enter the ID:</td>
<td><input type="text" name="id"></td>
</tr>
<tr>
<td colspan=”2”><input type=”submit” value=”search”></td>
</tr>
</table>
</form>
</body>
</html>
```

Right click on src folder- New- Servlet-Class name-Servlet1-Next-Next-Finish

```
Import java.IOException;
Import javax.servlet.*;
Import javax.servlet.http.*;
Public class Servlet1 extends HttpServlet
{
Private static final long serialversionUID=1L;
Public Servlet()
{
Super();
Protected void doGet (HttpServletRequest request, HttpServletRequest reponse)
PrintWriter out=response.getWriter();
String id=request.getParameter("id");
Out.print("<h1>Display the record</h1>");

Out.print("<table border='1'><tr><th>ID</th><th>Name</th><th>Address</th></tr>");
Try{
Class.forName("jdbc.postgresql.Driver");
Connection con=DriverManager.getConnection("jdbc:postgres:test","postgres","Pass@word");
Statement stmt=con.createStatement();
ResultSet rs=stmt.executeQuery("select *from customer where id="+id+");
While(rs.next())
{
Out.print("<tr><td>");\
Out.println(rs.getInt(1));
Out.print("</td>");
```

```
Out.print(rs.getString(2));
Out.print("</td>");
Out.print("</td>");
Out.print(rs.getString(3));
Out.print("</td>");
Out.print(i);
Out.print("</tr>");
}
}
Catch(Exception p)
{ System.out.println(p);
}
Out.print("</table>");
}
```

<u>Create table customer and insert the values for customer id , customer name and customer address</u>

Copy the jar files –web Content- Web INF-right click on lib- paste

Right Click on jar file-Buildpath-add to Build Path

Right click on index.html file – run as Tomcat Server.