Object-Oriented Design Lab Report (Java 4)

Bisakh Mondal 001810501079

Format: Approach(if notable) | Code | Output

Java Assignment

(Q1)

It's not an example of a **threadsafe** program. It's taken place due to a multiple numbers of possible non-deterministic interleavings.

```
import java.util.*;
import java.io.*;

class Interaction
{
    int x;
    public Interaction(int y)
    {
        x=y;
    }
    public void increase(int temp)
    {
        x= x+temp;
        System.out.println(Thread.currentThread().getName()+" new
x: "+x);
    }
    public void decrease(int temp)
    {
        x=x-temp;
        System.out.println(Thread.currentThread().getName()+" new x
: "+x);
    }
}
class Test1 implements Runnable
{
    Interaction d;
    int temp;
    public Test1(Interaction d1,int temp1)
    {
```

```
        d=d1;
        temp=temp1;
    }
    public void run()
    {
        for(int i=0;i<15000;i++){
            d.increase(temp);
        }
    }

}
class Test2 implements Runnable
{
    Interaction d;
    int temp;
    public Test2(Interaction d1,int temp1)
    {
        d=d1;
        temp=temp1;
    }
    public void run()
    {
        for(int i=0;i<15000;i++)
            d.decrease(temp);
    }

}

class Assign_1
{
    public static void main(String[] args) {
        Interaction d=new Interaction(5);
        Test1 run1=new Test1(d,1);
        Test2 run2=new Test2(d,1);
        Thread t1=new Thread(run1);
        Thread t2=new Thread(run2);
        t1.start();
        t2.start();

        try {
            t1.join();
            t2.join();
        }
        catch(InterruptedException e) {
```

```
            e.printStackTrace();
        }

        System.out.println("Value after all threads finished : " +
d.);
    }
}
```

```
Thread-0 new x: -1
Thread-0 new x: 0
Thread-0 new x: 1
Thread-0 new x: 2
Thread-0 new x: 3
Thread-0 new x: 4
Value after all threads finished : 4
```

```
Thread-1 new x : 8
Thread-1 new x : 7
Thread-1 new x : 6
Thread-1 new x : 5
Value after all threads finished : 5
```

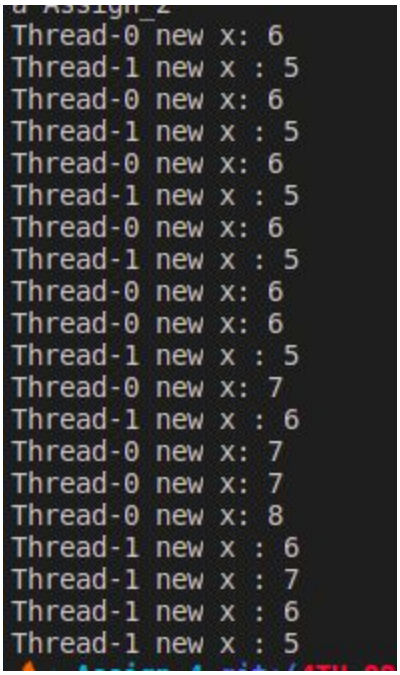
(Q2)

By making the blocks **synchronized** where the same variable gets accessed by multiple concurrent threads this scenario can be avoided. It's like a **Mutex.**

```
import java.util.*;
import java.io.*;

class Interaction
{
    int x;
    public Interaction(int y)
    {
        x=y;
    }
     public void increase(int temp)
    {
        synchronized(this){
```

```
        this.x+=temp;
        }
        System.out.println(Thread.currentThread().getName()+" new x:
"+x);
    }
     public void decrease(int temp)
    {
        synchronized(this){
        this.x-=temp;
        }
        System.out.println(Thread.currentThread().getName()+" new x :
"+x);
    }
}
class Test1 implements Runnable
{
    Interaction d;
    int temp;
    public Test1(Interaction d1,int temp1)
    {
        d=d1;
        temp=temp1;
    }
    public void run()
    {
        for(int i=0;i<10;i++)
            d.increase(temp);
    }

}
class Test2 implements Runnable
{
    Interaction d;
    int temp;
    public Test2(Interaction d1,int temp1)
    {
        d=d1;
        temp=temp1;
    }
```

```
    public void run()
    {
        for(int i=0;i<10;i++)
            d.decrease(temp);
    }

}

class Assign_2
{
    public static void main(String[] args) {
        Interaction d=new Interaction(5);
        Test1 run1=new Test1(d,1);
        Test2 run2=new Test2(d,1);
        Thread t1=new Thread(run1);
        Thread t2=new Thread(run2);
        t1.start();
        t2.start();
    }
}
```

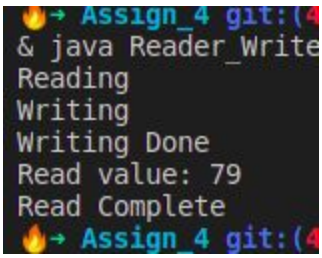


(Q3)

```
import java.util.*;

class Data{
    int d;
    Data(int d){
        this.d=d;
    }
    synchronized int read(){
        return d;
    }
    synchronized void write(int k){
        d=k;
    }
}
class OPSAllowed{
    int readC, writeC;
    OPSAllowed(){
        readC=0;
        writeC=0;
    }
    int SyncRead(Data dat){
        System.out.println("Reading");
        synchronized(this){
        while(writeC!=0)
            try{
                wait();
            }catch(Exception e){}
            readC++;
        }
        int r = dat.read();
        System.out.println("Read value: "+r);
        synchronized(this){
            readC--;
            notifyAll();
        }
        System.out.println("Read Complete");
        return r;
```

```
    }
    void SyncWrite(Data d,int k){
        System.out.println("Writing");
        synchronized(this){
        while(writeC!=0 && readC!=0){
            try{
                wait();
            }
            catch(Exception e){}
        }
            writeC++;
        }
        d.write(k);
        synchronized(this){
            writeC--;
            notifyAll();
        }
        System.out.println("Writing Done");
    }

}
class Reader implements Runnable{
    Data d;
    OPSAllowed ops;
    Reader(Data dat, OPSAllowed opp){
        d=dat;
        ops=opp;
    }
    public void run(){
        int dd=ops.SyncRead(d);
    }
}
class Writer implements Runnable{
    Data d;
    OPSAllowed ops;
    Writer(Data dat, OPSAllowed opp){
        d=dat;
        ops=opp;
    }
```

```
    public void run(){
        ops.SyncWrite(d,50);
    }
}

public class Reader_Writer {
    public static void main(String[] args) {
        Data d= new Data(79);
        OPSAllowed op = new OPSAllowed();
        Thread r = new Thread(new Reader(d,op));
        Thread w = new Thread(new Writer(d, op));

        r.start();
        w.start();

    }
}
```



(Q4)

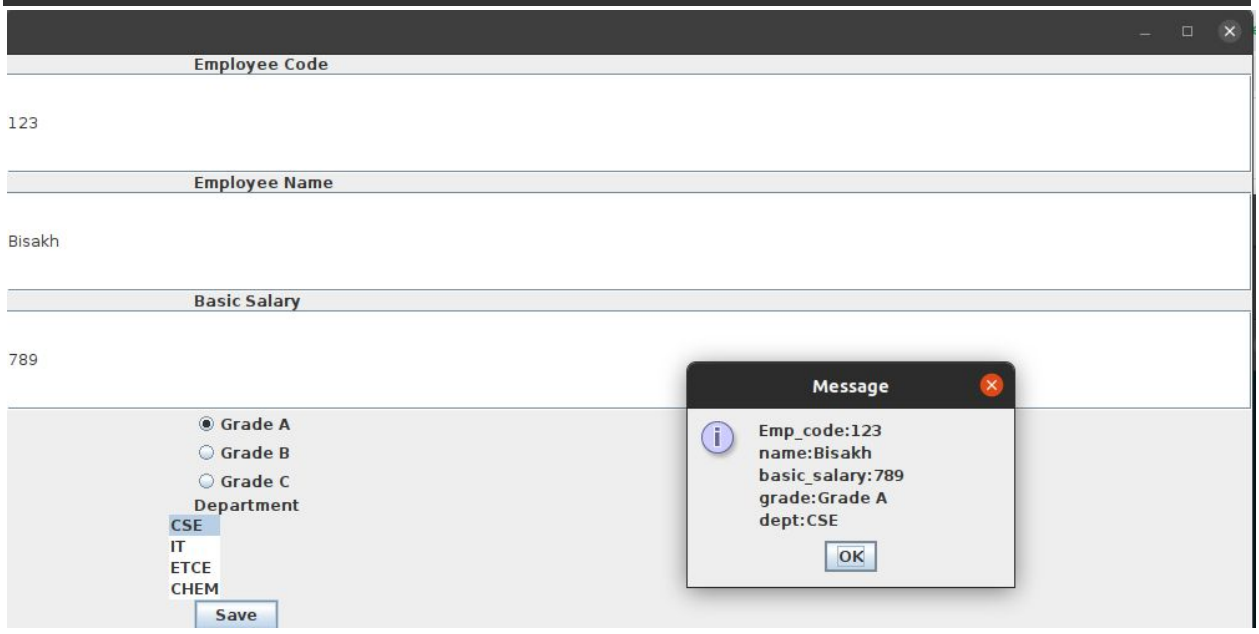
```
import java.util.*;
import java.awt.event.ActionListener;
import java.io.*;
import javax.swing.*;

class Employee
{
    String emp_code;
    String name;
    String basic_salary;
```

```
    String grade;
    String dept;
    public Employee(String s1,String s2,String s3,String s4,String
s5)
    {
        emp_code=s1;
        name=s2;
        basic_salary=s3;
        grade=s4;
        dept=s5;
    }
    public Employee(String s1)
    {
        emp_code=s1;
    }
    public boolean equals(Object t)
    {
        if(t instanceof Employee)
        {
            Employee s1=(Employee)t;
            return emp_code.equals(s1.emp_code);


        }
        else{
            return false;
        }
    }
    public String toString()
    {
        return
"Emp_code:"+emp_code+"\n"+"name:"+name+"\n"+"basic_salary:"+basic_sal
ary+"\n"+"grade:"+grade+"\n"+"dept:"+dept;
    }
}

class Render extends JFrame
{
    JPanel p;
```

```
    JLabel l1,l2,l3,l4,l5;
    JTextField t1,t2,t3;
    JRadioButton rb1,rb2,rb3;
    ButtonGroup bg1;
    JButton b;
    JList dept;
    JButton save;
    ArrayList<Employee> arr;
    public Render()
    {
        super();
        arr=new ArrayList<Employee>();
        setSize(1000,500);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        buildPanel();
        add(p);
        setVisible(true);

    }
    class MyListener implements ActionListener
    {
           @Override
           public void actionPerformed(java.awt.event.ActionEvent e)
{
            String s1,s2,s3,s4,s5;
            s1=t1.getText();
            s2=t2.getText();
            s3=t3.getText();
            s4="";
            if(rb1.isSelected()){
                s4="Grade A";
            }
            if(rb2.isSelected()){
                s4="Grade B";
            }
            if(rb3.isSelected()){
                s4="Grade C";
            }
            s5=(String)dept.getSelectedValue();
```

```
            if(arr.contains(new Employee(s1)))
            {
                JOptionPane.showMessageDialog(null,"Employee code
taken");
            }
            else{
                Employee e1=new Employee(s1,s2,s3,s4,s5);
                arr.add(e1);
                JOptionPane.showMessageDialog(null,e1.toString());
            }

           }
    }
    public void buildPanel()
    {
        p=new JPanel();
        p.setLayout(new BoxLayout(p, BoxLayout.Y_AXIS));
        l1=new JLabel("Employee Code");
        l2=new JLabel("Employee Name");
        l3=new JLabel("Basic Salary");
        t1=new JTextField("Code");
        t2=new JTextField("Name");
        t3=new JTextField("Salary");
        rb1=new JRadioButton("Grade A");
        rb2=new JRadioButton("Grade B");
        rb3=new JRadioButton("Grade C");
        bg1=new ButtonGroup();
        p.add(l1);
        p.add(t1);
        p.add(l2);
        p.add(t2);
        p.add(l3);
        p.add(t3);
        bg1.add(rb1);
        bg1.add(rb2);
        bg1.add(rb3);
        p.add(rb1);
        p.add(rb2);
        p.add(rb3);
```

```
        String arr[]={"CSE","IT","ETCE","CHEM"};
        dept=new JList(arr);
        dept.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        l4=new JLabel("Department");
        p.add(l4);
        p.add(dept);
        save=new JButton("Save");
        save.addActionListener(new MyListener());
        p.add(save);
}

}

class Assign_4
{
    public static void main(String[] args) {
        Render w=new Render();
    }
}
```



(Q4B)

```
import java.util.*;
import java.awt.event.ActionListener;
```

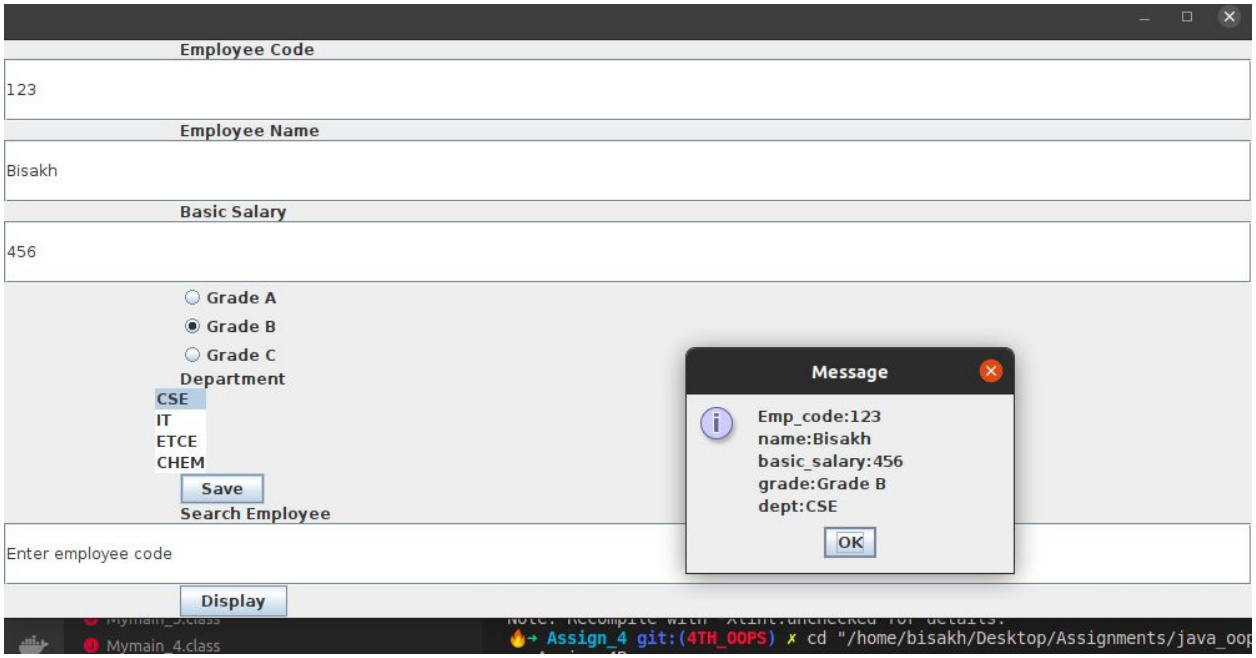
```
import java.io.*;
import javax.swing.*;

class Employee
{
    String emp_code;
    String name;
    String basic_salary;
    String grade;
    String dept;
    public Employee(String s1,String s2,String s3,String s4,String
s5)
    {
        emp_code=s1;
        name=s2;
        basic_salary=s3;
        grade=s4;
        dept=s5;
    }
    public Employee(String s1)
    {
        emp_code=s1;
    }
    public boolean equals(Object t)
    {
        if(t instanceof Employee)
        {
            Employee s1=(Employee)t;
            return emp_code.equals(s1.emp_code);


        }
        else{
            return false;
        }
    }
    public String toString()
    {
        return
```

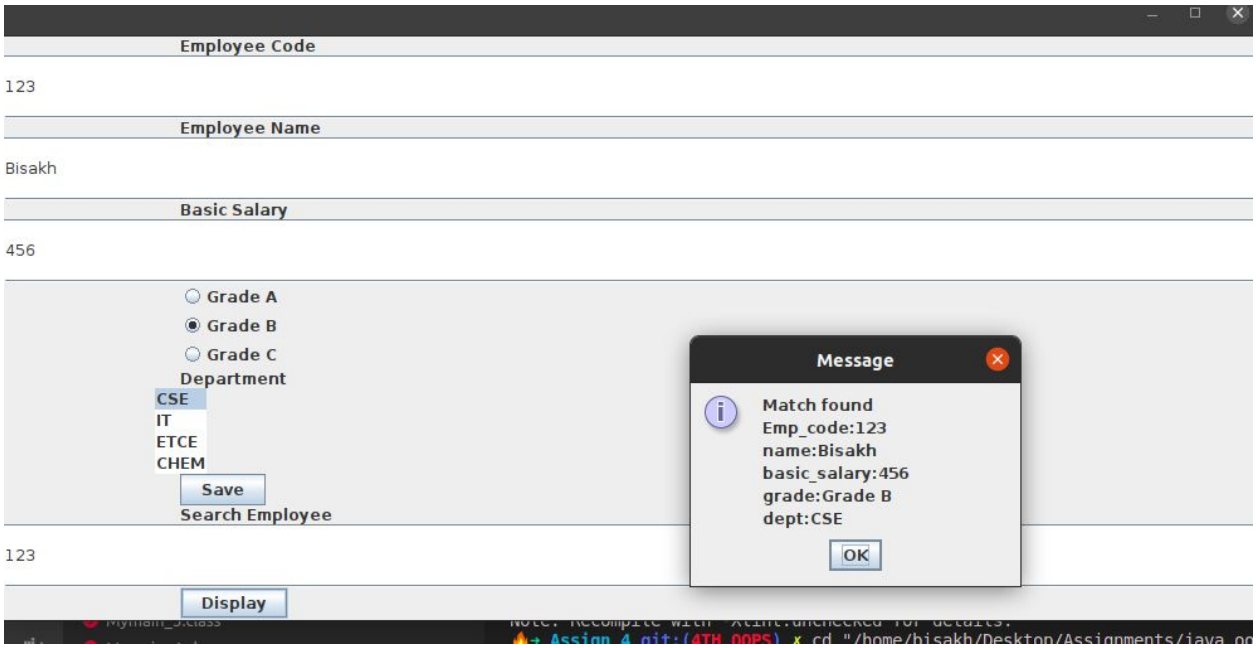
```
"Emp_code:"+emp_code+"\n"+"name:"+name+"\n"+"basic_salary:"+basic_sal
ary+"\n"+"grade:"+grade+"\n"+"dept:"+dept;
    }
}

class Render extends JFrame
{
    JPanel p;
    JLabel l1,l2,l3,l4,l5,l6;
    JTextField t1,t2,t3,t4;
    JRadioButton rb1,rb2,rb3;
    ButtonGroup bg1;
    // JButton b;
    JList dept;
    JButton save,disp;
    ArrayList<Employee> arr;
    public Render()
    {
        super();
        arr=new ArrayList<Employee>();
        setSize(1000,500);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        buildPanel();
        add(p);
        setVisible(true);

    }
    class SaveListener implements ActionListener
    {
           @Override
           public void actionPerformed(java.awt.event.ActionEvent e)
{
            String s1,s2,s3,s4,s5;
            s1=t1.getText();
            s2=t2.getText();
            s3=t3.getText();
            s4="";
            if(rb1.isSelected()){
                s4="Grade A";
```

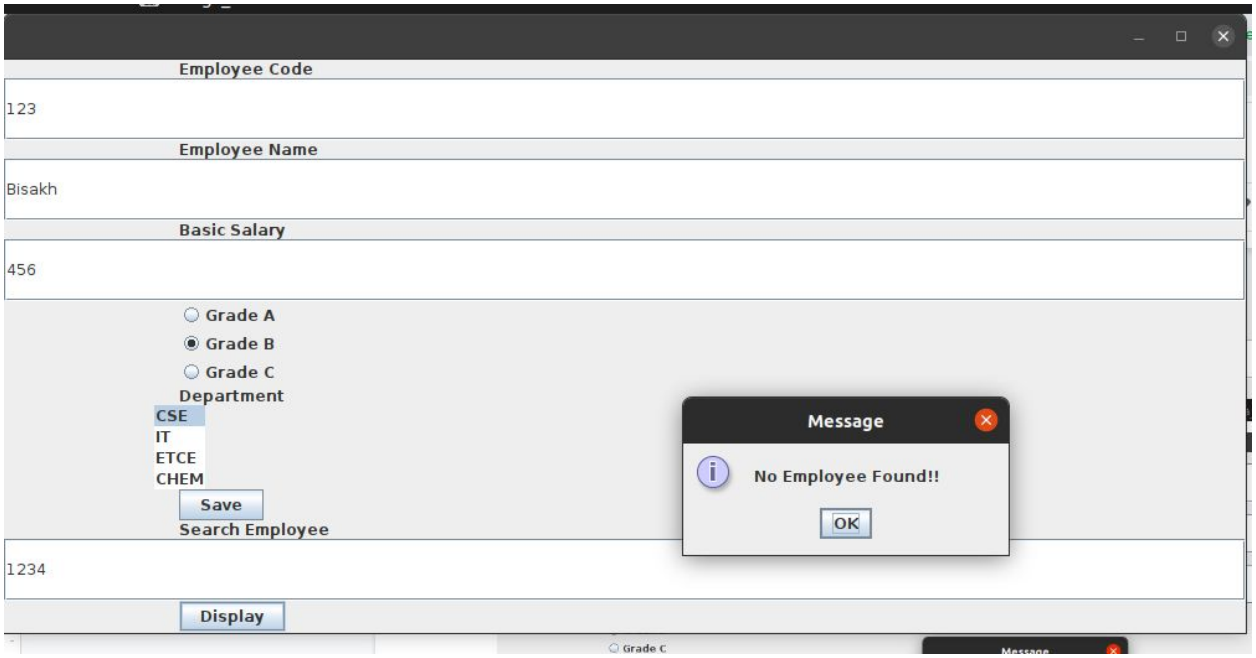
```
            }
            if(rb2.isSelected()){
                s4="Grade B";
            }
            if(rb3.isSelected()){
                s4="Grade C";
            }
            s5=(String)dept.getSelectedValue();
            if(arr.contains(new Employee(s1)))
            {
                JOptionPane.showMessageDialog(null,"Employee code
taken");
            }
            else{
                Employee e1=new Employee(s1,s2,s3,s4,s5);
                arr.add(e1);
                JOptionPane.showMessageDialog(null,e1.toString());
            }

           }
    }
    class DispListener implements ActionListener{

        @Override
        public void actionPerformed(java.awt.event.ActionEvent e){
            String empC= t4.getText();
            for(Employee emp: arr){
                if(emp.emp_code.equals(empC)){
                    JOptionPane.showMessageDialog(null, "Match
found\n"+emp.toString());
                    return;
                }
            }
            JOptionPane.showMessageDialog(null,"No Employee
Found!!");
        }
    }
    public void buildPanel()
    {
```

```
        p=new JPanel();
        p.setLayout(new BoxLayout(p, BoxLayout.Y_AXIS));
        l1=new JLabel("Employee Code");
        l2=new JLabel("Employee Name");
        l3=new JLabel("Basic Salary");
        t1=new JTextField("Code");
        t2=new JTextField("Name");
        t3=new JTextField("Salary");
        rb1=new JRadioButton("Grade A");
        rb2=new JRadioButton("Grade B");
        rb3=new JRadioButton("Grade C");
        bg1=new ButtonGroup();
        p.add(l1);
        p.add(t1);
        p.add(l2);
        p.add(t2);
        p.add(l3);
        p.add(t3);
        bg1.add(rb1);
        bg1.add(rb2);
        bg1.add(rb3);
        p.add(rb1);
        p.add(rb2);
        p.add(rb3);
        String arr[]={"CSE","IT","ETCE","CHEM"};
        dept=new JList(arr);
        dept.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        l4=new JLabel("Department");
        p.add(l4);
        p.add(dept);
        save=new JButton("Save");
        save.addActionListener(new SaveListener());
        p.add(save);
        l6 = new JLabel("Search Employee");
        p.add(l6);
        t4= new JTextField("Enter employee code");
        p.add(t4);
        disp = new JButton("Display");
        disp.addActionListener(new DispListener());
```

```
        p.add(disp);
}

}

class Assign_4B
{
    public static void main(String[] args) {
        Render w=new Render();
    }
}
```




Employee Code
123
Employee Name
Bisakh
Basic Salary
456
Grade A
Grade B
Grade C
Department
CSE
IT
ETCE
CHEM
Save
Search Employee
123
Display
Message
Match found
Emp_code:123
name:Bisakh
basic_salary:456
grade:Grade B
dept:CSE
OK


Employee Code
123
Employee Name
Bisakh
Basic Salary
456
Grade A
Grade B
Grade C
Department
CSE
IT
ETCE
CHEM
Save
Search Employee
1234
Display
Message
No Employee Found!!
OK