Object-Oriented Design Lab Report (Java 3)

Bisakh Mondal 001810501079

Format: Approach(if notable) | Code | Output

Java Assignment

(Q1)

Solution involving comparator interface, Arrarlist, iterators etc.

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

class Department
{
    String dept_code;
    String dept_name;
    String location;
    Department(String s1,String s2,String s3)
    {
        dept_code=s1;
        dept_name=s2;
        location=s3;
    }
    Department(String s1)
    {
        dept_code=s1;
    }
    public String toString()
    {
        return dept_code+" "+dept_name+" "+location;
    }
 }

class Employee{
    String emp_code;
    String emp_name;
    double basic;
    String dept_code;
    Employee(String s1,String s2,double s3,String s4)
    {
```

```
        emp_code=s1;
        emp_name=s2;
        basic=s3;
        dept_code=s4;
    }
    Employee(String s1)
    {
        emp_code=s1;
    }
    public String toString()
    {
        return emp_code+ " "+emp_name+" "+basic+" "+dept_code;
    }

}

class Management
{
    ArrayList<Employee> e;
    ArrayList<Department> d;
    Management()
    {
        e=new ArrayList<Employee>();
        d=new ArrayList<Department>();
    }
    public boolean existemployee(String s)
    {
        Iterator it=e.iterator();
        while(it.hasNext())
        {
            Employee e1=(Employee)it.next();
            if(e1.emp_code.equals(s))
            {
                return true;
            }
        }
        return false;
    }
    public boolean existemployee(Employee t)
    {
        return e.contains(t);
    }
    public boolean existdepartment(String t)
    {
        Iterator it=d.iterator();
        while(it.hasNext())
        {
```

```
            Department d1=(Department)it.next();
            if(d1.dept_code.equals(t))
            {
                return true;
            }
        }
        return false;
    }
    public boolean existdepartment(Department t)
    {
        return d.contains(t);
    }
    public void addEmployee()
    {
        Scanner sc=new Scanner(System.in);
        String s1,s2,s3;
        double s4;
        System.out.println("Enter emp_code");
        s1=sc.next();
        System.out.println("Enter emp_name");
        s2=sc.next();
        System.out.println("Enter basic pay");
        s4=sc.nextDouble();
        System.out.println("Enter dept_code");
        s3=sc.next();
        if(this.existemployee(s1))
        {
            System.out.println("Employee already exists");
        }
        else{
            if(!(this.existdepartment(s3)))
            {
                System.out.println("Department does not exist");
            }
            else{
            e.add(new Employee(s1,s2,s4,s3));
            System.out.println("Employee added successfully");
        }

        }
    }
    public void addDepartment()
    {
        Scanner sc=new Scanner(System.in);
        String s1,s2,s3;
        System.out.println("Enter dept_code");
        s1=sc.next();
```

```
        System.out.println("Enter dept_name");
        s2=sc.next();
        System.out.println("Enter location");
        s3=sc.next();
        if(this.existdepartment(s1))
        {
            System.out.println("Departmernt already exists");
        }
        else{
            d.add(new Department(s1,s2,s3));

        }
    }
    public void totalbasicSalary()
    {
        double total=0;
        Iterator it=e.iterator();
        while(it.hasNext())
        {
            Employee e1=(Employee)it.next();
            total+=e1.basic;
        }
        System.out.println("The total basic salary is:"+total);

    }
    public void removeEmployee()
    {
        Scanner sc=new Scanner(System.in);
        String s1;
        System.out.println("Enter emp_code");
        s1=sc.next();
        Employee e1=new Employee(s1);
        // System.out.println(e.contains(e1));
        if(e.contains(e1))
        {
            e.remove(e1);
            System.out.println("Object removed");
        }
        else{
            System.out.println("Employee code is invalid");
        }
    }
    public void displayAllEmployee()
    {
        Iterator it=e.iterator();
        while(it.hasNext())
        {
```

```
            Employee e1=(Employee)it.next();
            System.out.println(e1);
        }
    }
    public void displayAllDepartment()
    {
        Iterator it=d.iterator();
        while(it.hasNext())
        {
            Department e1=(Department)it.next();
            System.out.println(e1);
        }

    }
    public void displayEmployee()
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter employee id");
        String code=sc.next();
        Iterator it=e.iterator();
        while(it.hasNext())
        {
            Employee e1=(Employee)it.next();
            if(e1.emp_code.equals(code))
            {
                System.out.println(e1);
                code=e1.dept_code;
            }
        }
        Iterator it1=d.iterator();
        while(it1.hasNext())
        {
            Department e1=(Department)it1.next();
            if(e1.dept_code.equals(code))
            {
                System.out.println(e1);
            }
        }

    }
    public void modifyEmployee()
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter employee id");
        String code=sc.next();
        double s4;
        Iterator it=e.iterator();
```

```
        while(it.hasNext())
        {
            Employee e1=(Employee)it.next();
            if(e1.emp_code.equals(code))
            {
                System.out.println("Enter basic pay");
                s4=sc.nextDouble();
                e1.basic=s4;
                System.out.println("Basic pay modified");
            }
        }

    }
    public void sortonemp()
    {
        Collections.sort(e,new Sortonemp());
        this.displayAllEmployee();
    }
    public void sortondept()
    {
        Collections.sort(e,new Sortondept());
        this.displayAllEmployee();
    }
}

class Sortonemp implements Comparator<Employee>
{
    public int compare(Employee e1,Employee e2)
    {
        String str1=e1.emp_code;
        String str2=e2.emp_code;
        int l1 = str1.length();
        int l2 = str2.length();
        int lmin = Math.min(l1, l2);

        for (int i = 0; i < lmin; i++) {
            int str1_ch = (int)str1.charAt(i);
            int str2_ch = (int)str2.charAt(i);

            if (str1_ch != str2_ch) {
                return str1_ch - str2_ch;
            }
        }
        if (l1 != l2) {
            return l1 - l2;
        }
```

```
        else {
            return 0;
        }
    }
}
class Sortondept implements Comparator<Employee>
{
    public int compare(Employee e1,Employee e2)
    {
        String str1=e1.dept_code;
        String str2=e2.dept_code;
        int l1 = str1.length();
        int l2 = str2.length();
        int lmin = Math.min(l1, l2);

        for (int i = 0; i < lmin; i++) {
            int str1_ch = (int)str1.charAt(i);
            int str2_ch = (int)str2.charAt(i);

            if (str1_ch != str2_ch) {
                return str1_ch - str2_ch;
            }
        }

        if (l1 != l2) {
            return l1 - l2;
        }
        else {
            return 0;
        }
    }
}

class Assign_1
{
    public static void main(String args[])
    {
        Management m=new Management();
        int choice;
        Scanner sc=new Scanner(System.in);
        while(true)
        {
            System.out.println("1:add employee\n2:add department\n3:display
employee\n4:display all employees\n5:total basic pay\n6:modify basic pay\n7:sort on
emp_code\n8:sort on dept_code\n9:Remove employee\n0:Exit");
            choice=sc.nextInt();
            if(choice==0)
```

```
            {
                break;
            }
            else{
                switch(choice)
                {
                    case 1:
                    m.addEmployee();
                    break;

                    case 2:
                    m.addDepartment();
                    break;

                    case 3:
                    m.displayEmployee();
                    break;

                    case 4:
                    m.displayAllEmployee();
                    break;

                    case 5:
                    m.totalbasicSalary();
                    break;

                    case 6:
                    m.modifyEmployee();
                    break;

                    case 7:
                    m.sortonemp();
                    break;

                    case 8:
                    m.sortondept();
                    break;

                    case 9:
                    m.removeEmployee();
                    break;

                    case 0:
                    return ;
    }
}}}
}
```

```
7
12 emp1 123.0 D1
123 Bis 456.0 D0
1:add employee
2:add department
3:display employee
4:display all employees
5:total basic pay
6:modify basic pay
7:sort on emp_code
8:sort on dept_code
9:Remove employee
0:Exit
8
123 Bis 456.0 D0
12 emp1 123.0 D1
```

(Q2)

```
import java.util.*;
class Account{
    String name, Acc_num;
    double balance;
    Account(String Acc,String n,double d){
        Acc_num=Acc;
        name=n;
        balance=d;
    }
    }
class Query{
    HashMap<String,Double> h=new HashMap<String,Double>();

    public void push(Account s){
        h.put(s.Acc_num, s.balance);
        // System.out.println(h.get(s.Acc_num));
    }
    public boolean exists(String acc){
        if(h.containsKey(acc))
        return true;
        System.out.println("Account "+acc+" does not exists.");
```

```
        return false;
    }
    public double fetch(String acc){
        if(exists(acc))
       return h.get(acc);
       return -1.0;
    }
}
class Assign2{
    public static void main(String[] args) {
        Account a=new Account("123","Bisakh",789.36);
        Account b=new Account("1234","Bis",79.367);
        Query q=new Query();
        q.push(a);
        q.push(b);
        System.out.println(q.fetch("12354"));
        System.out.println("Balance: "+q.fetch("123"));

    }
}
```

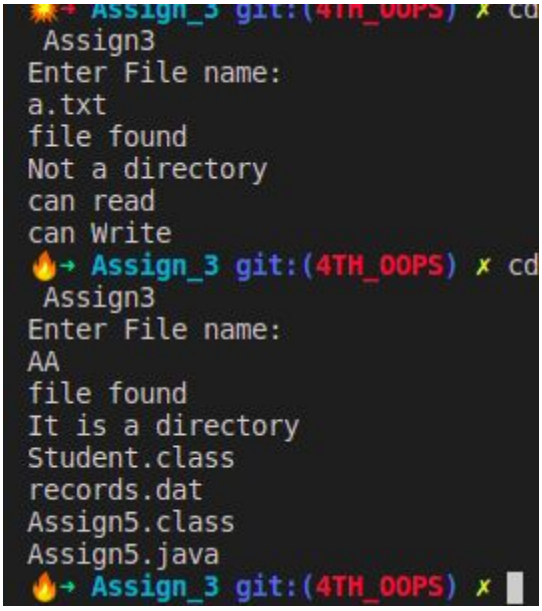
```
 Assign2
Account 12354 does not exists.
-1.0
Balance: 789.36
🔥→ Assign 3 git:(4TH OOPS) ✗
```

(Q3)

```
import java.io.*;
import java.util.*;
class Assign3{
    public static void main(String[] args) {
       System.out.println("Enter File name:");
       Scanner in =new Scanner(System.in);
       String filename=in.next();

       File obj=new File(filename);
```

```
        if(obj.exists()){
            System.out.println("file found");
        }
        else{
            System.out.println("File not Found");
            System.exit(-1);
        }
        if(obj.isDirectory()){
            System.out.println("It is a directory");
            for(File f1:obj.listFiles()){
                System.out.println(f1.getName());
            }
        }
        else{
            System.out.println("Not a directory");
            if(new File(filename).canRead()){
                System.out.println("can read");
            }
            else{
                System.out.println("can't Read");
            }
            if(new File(filename).canWrite()){
                System.out.println("can Write");
            }
            else{
                System.out.println("can't write");
            }
        }
    }
}
```



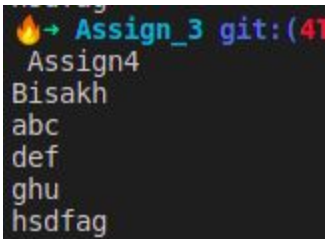
(Q4)

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

class Assign4 {
    public static void main(String[] args) {
        File fi = new File("a.txt");
        FileReader fr = null;
        try {
            fr = new FileReader(fi);
        } catch (FileNotFoundException e) {
            System.out.println("File Not found");
        }
        BufferedReader bf = new BufferedReader(fr);
        try {
            String name = null;
            while((name=bf.readLine())!=null){
                System.out.println(name);
            }

        } catch (IOException e) {
            e.printStackTrace();
```

```
        }
    // while((name=bf.readLine())!=null){
    //     System.out.println(name);
    // }
    }
}
```



(Q5)

```
import java.io.*;
class Student implements Serializable{
    private static final long serialVersionUID=1L;
    private String name,roll;
    private double score;
    Student(){};
    Student(String n,String r, double s){
        name=n;score=s;roll=r;
    }
    @Override
    public String toString(){
        return "Name: "+name+"\nRoll: "+roll+"\nScore: "+score;
    }
}
class Assign5{
    public static void main(String[] args) {
        Student s1 = new Student("Bisakh", "079", 89.2);
        Student s2 = new Student("Bisakh2", "0790", 899.5);
        try {
            System.out.println("\nwriting to file");
            FileOutputStream f=new FileOutputStream(new
File("records.dat"));
```

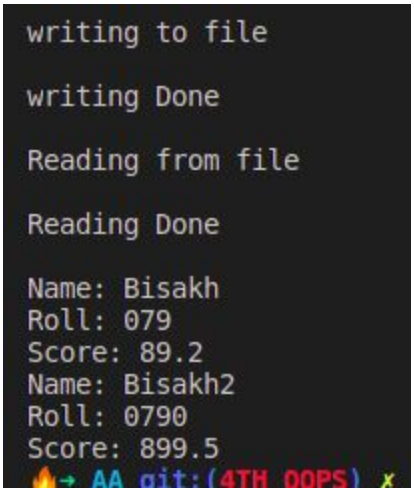
```
            ObjectOutputStream o = new ObjectOutputStream(f);
            o.writeObject(s1);
            o.writeObject(s2);
            o.close();
            f.close();
            System.out.println("\nwriting Done");

            System.out.println("\nReading from file");
            FileInputStream ff=new FileInputStream(new
File("records.dat"));
            ObjectInputStream i=new ObjectInputStream(ff) ;
            Student s3=(Student) i.readObject();
            Student s4=(Student) i.readObject();
            i.close();
            ff.close();

            System.out.println("\nReading Done\n");

            System.out.println(s3);
            System.out.println(s4);
        } catch (IOException ee) {
            System.out.println("Error Occured");
        }catch(ClassNotFoundException e){
            e.printStackTrace();
        }


    }
}
```


writing to file
writing Done
Reading from file
Reading Done
Name: Bisakh
Roll: 079
Score: 89.2
Name: Bisakh2
Roll: 0790
Score: 899.5