

PROJECT REPORT

Hospital Facility **System**

GROUP NO: (15)

W.T.T.Karunarathna PS-2019-010
E.K.H.De Soyza PS-2019-037
J.A.D.U.Sahan PS-2019-136
H.G.H. Eshan PS-2019-137
D.D.C.I.De Alwis PS-2019-235

Table Of Content

- **Acknowledgment**
- **Executive Summery**
- **Understanding the OOP concepts**
- **User manual for the Hospital Facility System Application**
- **Appendix**

Acknowledgement

We Would Like to Express Our Special Thanks and Gratitude to Our Lecturer in Charge in Module Dr.(Mrs.)B.M.Thosini.Kumarika Who Gave Us the Opportunity and Fruitful Guidance to Do This Project Under Introduction to Object Oriented Programming. By Doing This Project We Got Enriched With Many Information Which Can Help Us in Our Future.

Executive Summery

By This Assignment, Understanding the Basic Concepts of Object-Oriented Programming is Done.

In the Task 1 the Designs for the Hospital Management System Application is Done. In the Task 2 Identifying the Programming Concepts are Done.

By the Task 3 Program User Manual and the Coding Part are Done.

Learning outcomes covered

- Apply design and development principles in the construction of software systems of varying complexity.
- Develop the structures to represent objects and the algorithms to perform operations.
- Explain and utilize object-oriented concepts.
- Use an industry-leading Integrated Development Environment (IDE) to develop and manage software projects.

Scenario and the Task

Suwa Hospital is one of the new hospitals located in the Middle of the city. The company planned to implement an application in order to automate transaction process.

User Levels and Functionalities are Follows

MAIN CLASS

Data Requirements

- Doctor details
- Service details
- Patient details
- Hospital Bill details
- Payment details

PATIENT CLASS

- Manage all the operations of customer

Functional Requirements

- Add patient
- Search doctor
- Save patient
- Update patient
- Delete patient
- Search patient

FACILITIES CLASS

- Manage all the services patients want.

Functional Requirements

- MRI
- SCAN
- ECG
- EEG

DOCTOR CLASS

- Manage all the details of each doctor

Functional Requirements

- Add doctor
- Edit doctor
- Delete doctor
- Update doctor
- Search doctor
- Patient Number List

Understanding the OOP concepts

Object Oriented Programming System is a programming paradigm based on the concept of “objects” which contains Data and Methods. OOP concepts are the main pillars behind Java Object Oriented Programming. The Concepts of OOP will make understand how does it works. The primary purpose of OOP is to increase the Flexibility and the maintainability of the programs. Object Oriented programming brings together data and its behavior(methods) in a single location called object to makes it easier to understand how a program works. The basic OOP concepts are,

- Object
- Class
- Abstraction
- Inheritance
- Encapsulation
- Polymorphism

Object

Object is a set of data and its behavior (methods) , Object have two characteristics such as State and Behavior. Any entity that has state and behavior is known as an object. It can be physical and logical. An object has three characteristics:

State: it is the data or value of an object. Behavior: it is the variables and methods(functionality) of an object. Identity: it is the unique ID

Example:

Object = House

State = Address, Color, Area

Behavior = Open door, close door

Class

The class is a group of similar entities comprises with Logical component and not the physical entity. The class uses methods to define the behaviors of the objects. The class that contains the main method of a Java program represents the entire program. A class represents a concept, and an object represents the embodiment of the concept. Multiple objects can be created from same class. Object definition is done by calling the class constructor. a class in java contains

- Properties.
- Methods: exposes behaviors of an object (Code reusability and code optimization).
- Constructors: Having same method name and class name. (a special class of method).
- Blocks.
- Nested Class, Interfaces, abstract class and inner class

Abstraction

Abstraction Is a process of hide unnecessary “details” and show only the relevant “data” of an Object from the user. We only mentioned the useful data for our scenario. We should have added religion of the customer and so on but as a bank no point of using those type of data.

There are two types of Abstraction

- Abstract Class
- Interface

Abstract Class:

A class that is declared as abstract is known as abstract class. It needs to be extended and its method implemented. It cannot be instantiated. A method that is declared as abstract and does not have implementation is known as abstract method.

Interface:

An interface in java is a blueprint of a class. It has static constants and abstract methods. The interface in java is a mechanism to realize abstraction. There can be only abstract methods in the java interface not method body. It is used to realize abstraction and multiple inheritances in Java. Java Interface also represents IS-A relationship.

Inheritance

Inheritance is when a class needs properties and functionalities of another class (sub classes) which provides the idea of reusability of the code. When a class needs the properties and functionalities of another class that class inherit from that class. Simple idea is Parent class having child classes. The child class will have all the properties and functionalities as the parent class and some different properties and functionalities which are unique for that child class. It is used to achieve runtime polymorphism.

Encapsulation

Encapsulation is the OOP technique of wrapping the data and the code. Always in OOP variables in a class are always hidden from the other class. It can only access by using a method like Getter and setter of that current class. Encapsulation also known as data hiding. Due to this, the class will become read only or write only. An encapsulated object can be thought of as a black box (its inner workings are hidden from the client). Encapsulation also known as combination of data hiding and data abstractions. By encapsulation can achieve the code secure.

The whole idea behind encapsulation is to hide the implementation details from users. This accessible method can be done in four level,

- Private: Accessible only by the class.
- Public: Accessible by any class.
- Protector: Accessible only by the subclasses in other package.
- Default: Accessible is available to any other class in the same package.

Polymorphism

The Simple idea of polymorphism is many things performed by the same object. In other words, polymorphism allows describing one interface and having multiple applications. Polymorphism is the feature which allows to have common activities among the classes. And that behavior will change based on that requirements Polymorphism is widely used in implementing inheritance. There are two methods of achieving polymorphism in java,

1. Method Overloading.
2. Method Overriding

Method

Method is a set of code which is referred to its name and can be called invoked at any point in a program simply by utilizing the method's name.

Method Overloading:

The simple idea of overloading is in the same class there are two methods in the same name but different parameters or argument list.

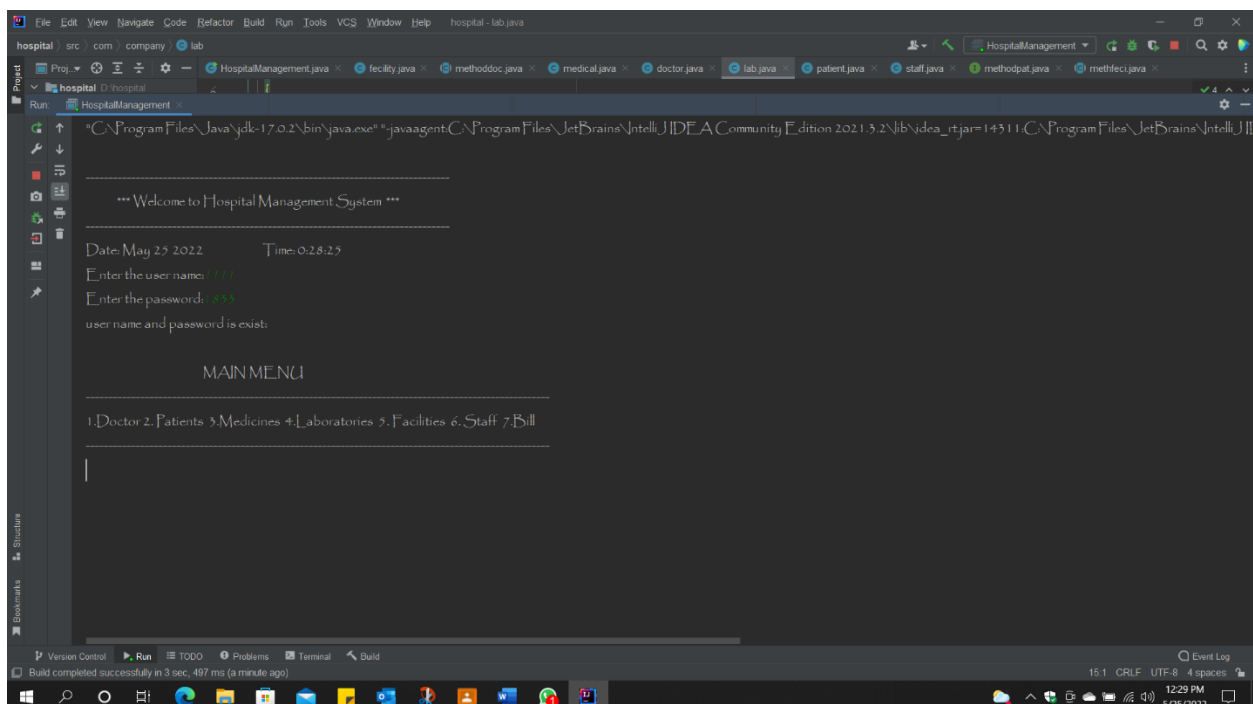
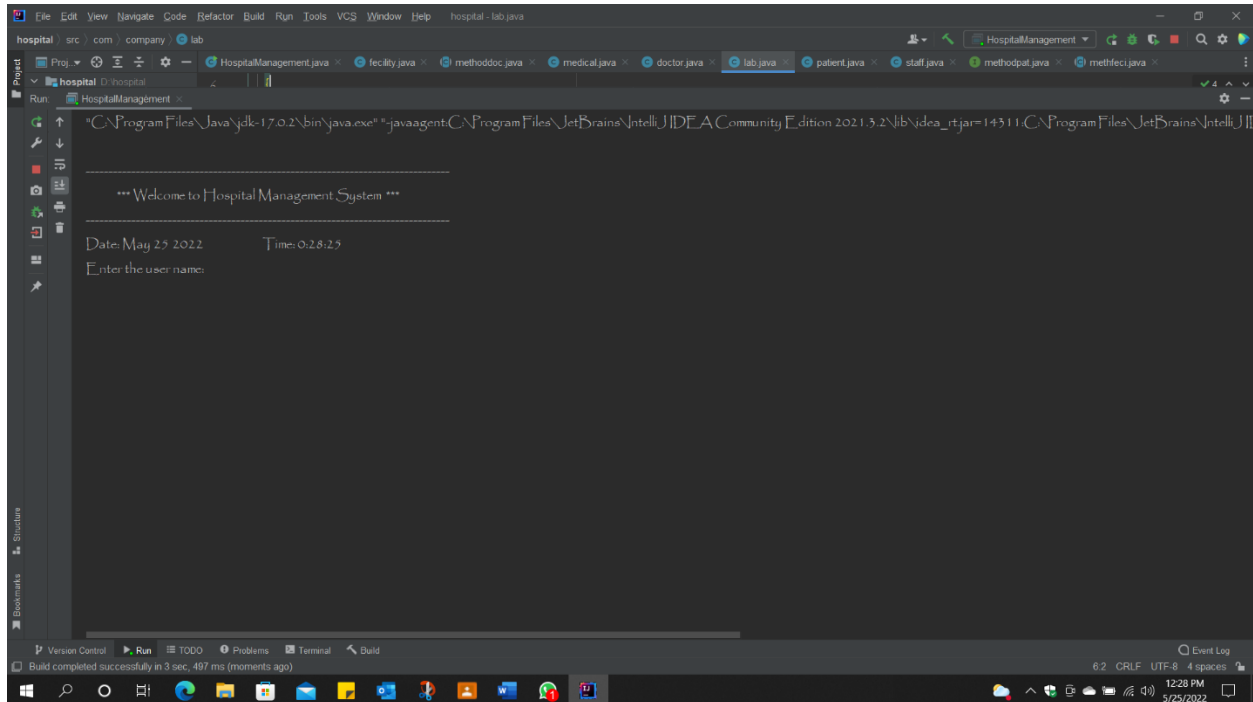
Method Overriding:

Child class having the same method of the parent class is overriding. In such situation child class overrides the parent class method without coding that again.

User Manual For The Hospital Facility System Application

Login

When you first open the application, it will open a window Hospital Management System.



The screenshot shows the IntelliJ IDEA IDE with the Hospital Management System project. The Run window displays the following output:

```
***Welcome to Hospital Management System***  
  
Date: May 25 2022      Time: 0:28:25  
Enter the user name: admin  
Enter the password: 12345  
user name and password is exist.  
  
MAIN MENU  
-----  
1.Doctor 2.Patients 3.Medicines 4.Laboratories 5.Facilities 6.Staff 7.Bill  
0.Exit  
-----  
--DOCTOR SECTION--  
-----  
1).Add New Entry  
2).Existing Doctors List  
0.Exit
```

The IDE interface shows the project structure on the left, the Run window at the bottom, and the status bar at the very bottom indicating the build completed successfully.

Doctor section

The screenshot shows the IntelliJ IDEA IDE with the Hospital Management System project. The Run window displays the following output:

```
--DOCTOR SECTION--  
-----  
1).Add New Entry  
2).Existing Doctors List  
0.Exit  
-----  
id->10  
name->Nikhil  
specilization->Physio  
work time->10:00  
qualification->M.D  
room no->  
  
Return to Back Press 1 and for Main Menu Press 0  
0.Exit  
-----  
1).Add New Entry  
2).Existing Doctors List  
0.Exit  
-----  
id Name Specialist Timing Qualification RoomNo.
```

The IDE interface shows the project structure on the left, the Run window at the bottom, and the status bar at the very bottom indicating the build completed successfully.

```
hospital - lab.java
Project: src \ com \ company \ lab
Run: HospitalManagement.java
specilization=
work time=
qualification=
room no.=

Return to Back Press 1 and for Main Menu Press 0

1)Add New Entry
2)Existing Doctors List

id Name Specilist Timing Qualification RoomNo.

22 Dr.Karunaratna ENT 9-10AM MBBSMD 12
32 Dr.Udara Physician 10-3AM MBBSMD 4
17 Dr.Eshan Surgeon 8-2AM BDM 8
33 Dr.Kavindu Artho 10-4PM MBBSMS 40
02 Kalpa Den 10-15 no 1

Return to Back Press 1 and for Main Menu Press 0
```

Patient Section

```
hospital - lab.java
Project: src \ com \ company \ lab
Run: HospitalManagement.java
user name and password exists:

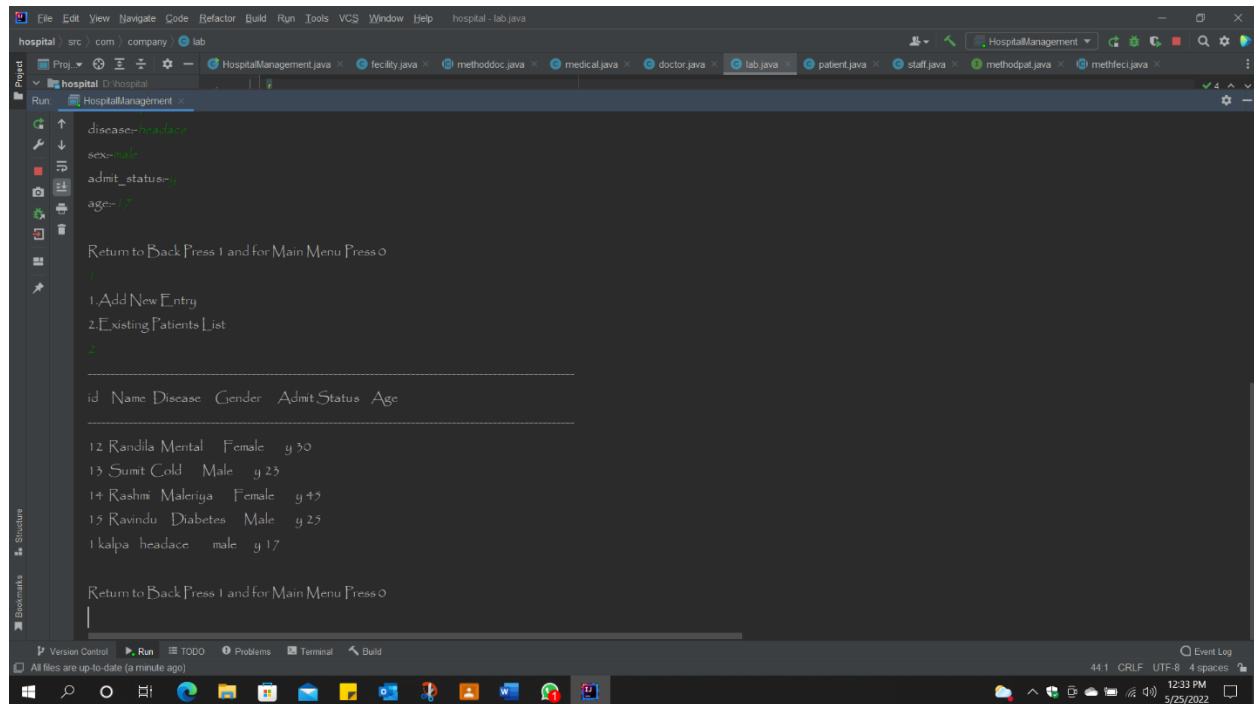
MAINMENU
1.Doctor 2.Patients 3.Medicines 4.Laboratories 5.Facilities 6.Staff 7.Bill

**PATIENT SECTION**

1.Add New Entry
2.Existing Patients List

id=
name=
disease=
sex=
admit_status=
age=

Return to Back Press 1 and for Main Menu Press 0
```



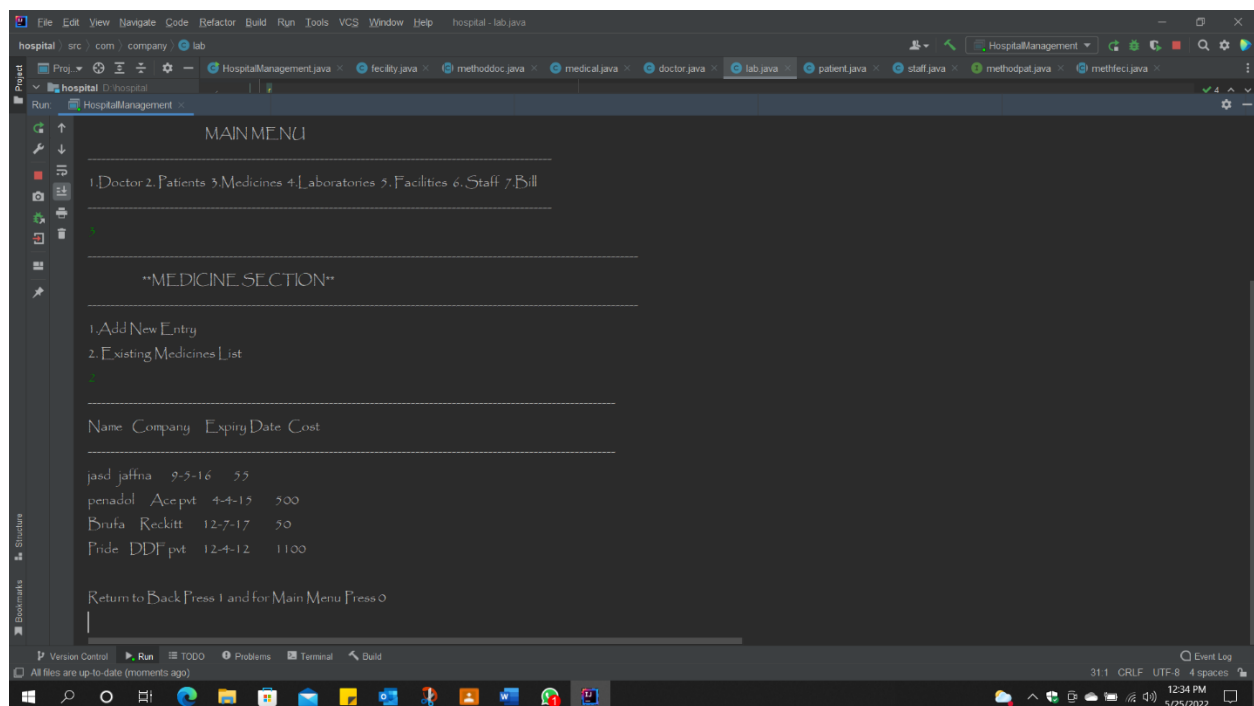
```
hospital - lab.java
hospital / src / com / company / lab
HospitalManagement.java x fecility.java x methoddoc.java x medical.java x doctor.java x lab.java x patient.java x staff.java x methodpat.java x methfecci.java x
Run: HospitalManagement x
disease-->
sex-->
admit_status-->
age-->

Return to Back Press 1 and for Main Menu Press 0
1
1.Add New Entry
2.Existing Patients List
2

id Name Disease Gender Admit Status Age
-----
12 Randila Mental Female y 30
13 Sumit Cold Male y 23
14 Rashmi Malaria Female y 45
15 Ravindu Diabetes Male y 25
1 kalpa headache male y 17

Return to Back Press 1 and for Main Menu Press 0
|
```

Medicine Section



```
hospital - lab.java
hospital / src / com / company / lab
HospitalManagement.java x fecility.java x methoddoc.java x medical.java x doctor.java x lab.java x patient.java x staff.java x methodpat.java x methfecci.java x
Run: HospitalManagement x
MAIN MENU
1.Doctor 2.Patients 3.Medicines 4.Laboratories 5.Facilities 6.Staff 7.Bill
1

--"MEDICINE SECTION"--
1.Add New Entry
2.Existing Medicines List
2

Name Company Expiry Date Cost
-----
jasd jafna 9-5-16 55
penadol Acept 4-4-15 500
Brufa Reckitt 12-7-17 50
Pride DDF prt 12-4-12 1100

Return to Back Press 1 and for Main Menu Press 0
|
```

Laboratory Section

The screenshot shows a Java IDE with the following content in the main editor:

```
MAIN MENU
-----
1.Doctor 2.Patients 3.Medicines 4.Laboratories 5.Facilities 6.Staff 7.Bill
-----

**LABORATORY SECTION**
-----
1.Add New Entry
2.Existing Laboratories List
-----

Facilities    Cost
-----
X-ray        1000
CT Scan      1200
OR Scan       500
Blood Bank    50

Return to Back Press 1 and for Main Menu Press 0
```

The IDE interface includes a menu bar (File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help), a toolbar, and a status bar at the bottom showing "56.1 CRLF UTF-8 4 spaces".

Facility Section

The screenshot shows a Java IDE with the following content in the main editor:

```
MAIN MENU
-----
1.Doctor 2.Patients 3.Medicines 4.Laboratories 5.Facilities 6.Staff 7.Bill
-----

**HOSPITAL FACILITY SECTION**
-----
1.Add New Facility
2.Existing Facilities List
-----

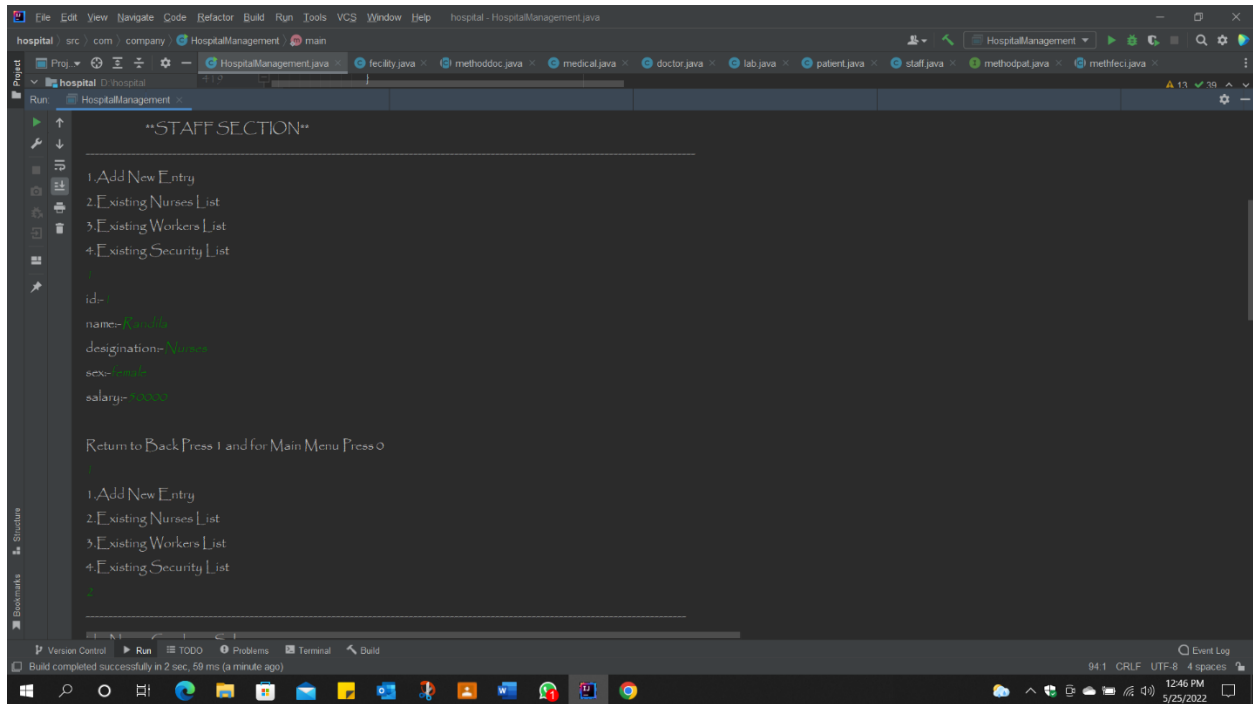
Hospital Facility are:
-----

Ambulance
Admt Facility
Canteen
Emergency

Return to Back Press 1 and for Main Menu Press 0
```

The IDE interface includes a menu bar (File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help), a toolbar, and a status bar at the bottom showing "81.1 CRLF UTF-8 4 spaces".

Staff section



The screenshot shows an IDE window titled "hospital - HospitalManagement.java". The code is in a dark theme. The main content area displays the following text:

```

**STAFF SECTION**

1.Add New Entry
2.Existing Nurses List
3.Existing Workers List
4.Existing Security List

id=
name=
designation=
sex=
salary=

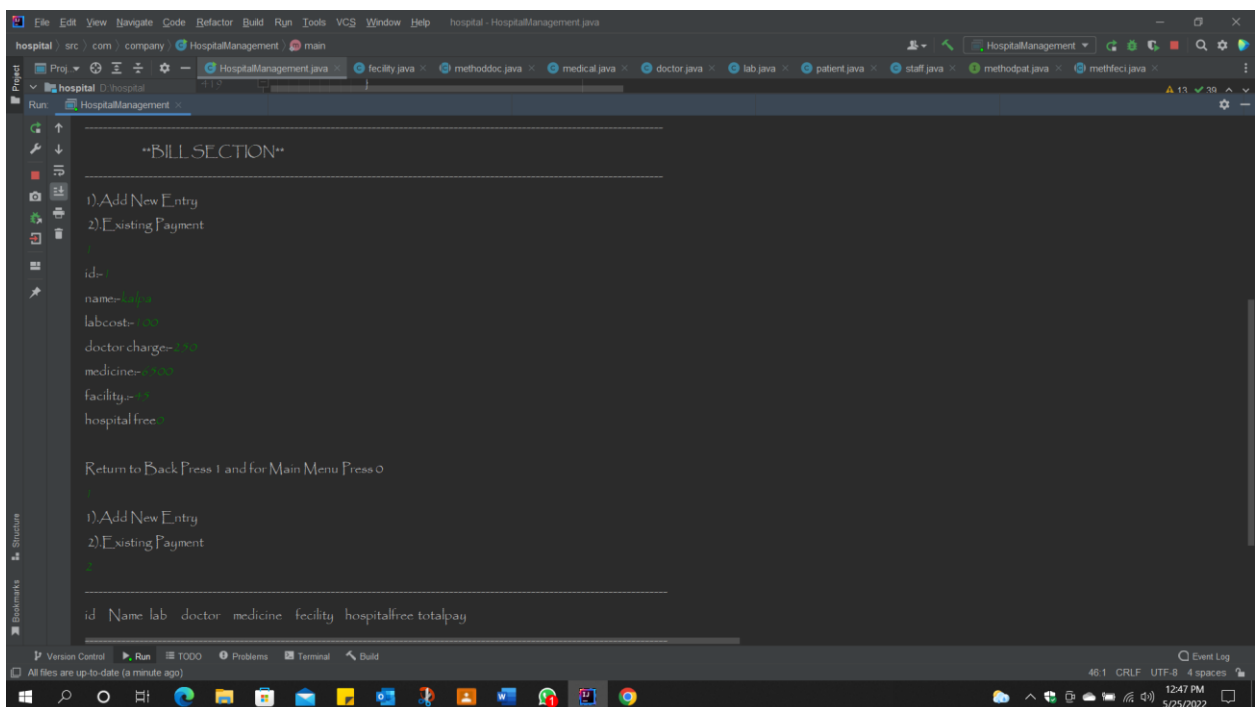
Return to Back Press 1 and for Main Menu Press 0

1.Add New Entry
2.Existing Nurses List
3.Existing Workers List
4.Existing Security List

```

The IDE interface includes a menu bar (File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help), a toolbar, and a status bar at the bottom showing "94.1 CRLF UTF-8 4 spaces" and "12:46 PM 5/25/2022".

Bill Section



The screenshot shows an IDE window titled "hospital - HospitalManagement.java". The code is in a dark theme. The main content area displays the following text:

```

**BILL SECTION**

1).Add New Entry
2).Existing Payment

id=
name=
labcost=
doctor charge=
medicine=
facility=
hospital free=

Return to Back Press 1 and for Main Menu Press 0

1).Add New Entry
2).Existing Payment

```

The IDE interface includes a menu bar (File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help), a toolbar, and a status bar at the bottom showing "46.1 CRLF UTF-8 4 spaces" and "12:47 PM 5/25/2022".

The screenshot shows an IDE with the following components:

- Menu Bar:** File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, VCS, Window, Help.
- Project Explorer:** hospital (src, com, company), HospitalManagement (main), HospitalManagement.java.
- Run Console:**

```

doctor charge-->0
medicine-->0
facility-->0
hospital free-->0

Return to Back Press 1 and for Main Menu Press 0

1).Add New Entry
2).Existing Payment

-----
id Name lab doctor medicine facility hospitalfree totalpay
-----
20 Kalana 1250.0 1500.0 1265.0 50000.0 1250.0 0.0
02 Kalana 1250.0 1500.0 1265.0 50000.0 1250.0 0.0
03 Kalana 1250.0 1500.0 1265.0 50000.0 1250.0 0.0
04 Kalana 1250.0 1500.0 1265.0 50000.0 1250.0 0.0
1 kalpa 100.0 250.0 6500.0 45.0 0.0 6895.0

Return to Back Press 1 and for Main Menu Press 0

```
- Status Bar:** Version Control, Run, TODO, Problems, Terminal, Build. All files are up-to-date (a minute ago).
- System Tray:** 46.1 CRLF, UTF-8, 4 spaces, 12:48 PM, 5/25/2022.

Appendix

HospitalManagement.java

```
package com.company;
import jdk.jshell.spi.ExecutionControl;

import java.util.*;
import java.util.Calendar;

public class HospitalManagement
{
    public static void main(String[] args) {
        //Calander
        String[] months = {
            "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
        };
        Calendar calendar = Calendar.getInstance();
        //System.out.println("-----");
        int count1 = 4, count2 = 4, count3 = 4, count4 = 4, count5 = 4, count6 = 4, count7 = 4;
        System.out.println("\n-----");
        System.out.println("        *** Welcome to Hospital Management System ***");
        System.out.println("-----");
        System.out.print("Date: " + months[calendar.get(Calendar.MONTH)] + " " + calendar.get(Calendar.DATE) + " " +
calendar.get(Calendar.YEAR));
        System.out.println("\t\t\t\t\tTime: " + calendar.get(Calendar.HOUR) + ":" + calendar.get(Calendar.MINUTE) + ":" +
calendar.get(Calendar.SECOND));
        //endCalander
        //array class
        doctor[] d = new doctor[25];
        patient[] p = new patient[100];
        lab[] l = new lab[20];
        fecility[] f = new fecility[20];
        medical[] m = new medical[100];
        staff[] s = new staff[100];
        bill[] b = new bill[200];
        //end array class
        int i;
        for (i = 0; i < 25; i++)
            d[i] = new doctor();
        for (i = 0; i < 100; i++)
            p[i] = new patient();
        for (i = 0; i < 20; i++)
            l[i] = new lab();
        for (i = 0; i < 20; i++)
            f[i] = new fecility();
        for (i = 0; i < 100; i++)
            m[i] = new medical();
        for (i = 0; i < 100; i++)
            s[i] = new staff();
        for (i = 0; i < 200; i++)
            b[i] = new bill();
```

```
//first doctor
d[0].did = "22";
d[0].dname = "Dr.Karunarathna";
d[0].specilist = "ENT";
d[0].appoint = "5-10AM";
d[0].doc_qual = "MBBS,MD";
d[0].droom = 12;
//
d[1].did = "32";
d[1].dname = "Dr.Udara";
d[1].specilist = "Physician";
d[1].appoint = "10-3AM";
d[1].doc_qual = "MBBS,MD";
d[1].droom = 4;
//second doctor
d[2].did = "17";
d[2].dname = "Dr.Eshan";
d[2].specilist = "Surgeon";
d[2].appoint = "8-2AM";
d[2].doc_qual = "BDM";
d[2].droom = 8;
//
d[3].did = "33";
d[3].dname = "Dr.Kavindu";
d[3].specilist = "Artho";
d[3].appoint = "10-4PM";
d[3].doc_qual = "MBBS,MS";
d[3].droom = 40;
//first pacient
p[0].pid = "12";
p[0].pname = "Randila";
p[0].disease = "Mental";
p[0].sex = "Female";
p[0].admit_status = "y";
p[0].age = 30;
//
p[1].pid = "13";
p[1].pname = "Sumit";
p[1].disease = "Cold";
p[1].sex = "Male";
p[1].admit_status = "y";
p[1].age = 23;
//
p[2].pid = "14";
p[2].pname = "Rashmi";
p[2].disease = "Maleriya";
p[2].sex = "Female";
p[2].admit_status = "y";
p[2].age = 45;
//
p[3].pid = "15";
p[3].pname = "Ravindu";
p[3].disease = "Diabetes";
p[3].sex = "Male";
p[3].admit_status = "y";
p[3].age = 25;
```

```

// 1 medical os
    m[0].med_name = "jasd";
    m[0].med_comp = "jaffna";
    m[0].exp_date = "9-5-16";
    m[0].med_cost = 55;
    m[0].count = 8;
//2 medical os
    m[1].med_name = "penadol";
    m[1].med_comp = "Ace pvt";
    m[1].exp_date = "4-4-15";
    m[1].med_cost = 500;
    m[1].count = 5;
//3 medical os
    m[2].med_name = "Brufa";
    m[2].med_comp = "Reckitt";
    m[2].exp_date = "12-7-17";
    m[2].med_cost = 50;
    m[2].count = 56;
//4 medical os
    m[3].med_name = "Pride";
    m[3].med_comp = "DDF pvt";
    m[3].exp_date = "12-4-12";
    m[3].med_cost = 1100;
    m[3].count = 100;
//lab 1
    l[0].fecility = "X-ray ";
    l[0].lab_cost = 1000;
//lab2
    l[1].fecility = "CT Scan ";
    l[1].lab_cost = 1200;
//lab 3
    l[2].fecility = "OR Scan ";
    l[2].lab_cost = 500;
//lab 4
    l[3].fecility = "Blood Bank";
    l[3].lab_cost = 50;
//f1
    f[0].fec_name = "Ambulance";
//f2
    f[1].fec_name = "Admit Facility ";
//f3
    f[2].fec_name = "Canteen";
//f4
    f[3].fec_name = "Emergency";
//staff 1
    s[0].sid = "22";
    s[0].sname = "ishara";
    s[0].desg = "Worker";
    s[0].sex = "Male";
    s[0].salary = 5000;
//staff2
    s[1].sid = "23";
    s[1].sname = "Gihani";
    s[1].desg = "Nurse";
    s[1].sex = "Female";
    s[1].salary = 2000;

```

```
//staff 3
s[2].sid = "24";
s[2].sname = "Sanju";
s[2].desg = "Worker";
s[2].sex = "Male";
s[2].salary = 5000;
//staff 4
s[3].sid = "25";
s[3].sname = "Ranil";
s[3].desg = "Nurse";
s[3].sex = "Female";
s[3].salary = 20000;
//bill
```

```
b[0].bida = "20";
b[0].bname = "Kalana";
b[0].lab = 1250.00;
b[0].doctor = 1500;
b[0].medicine = 1263;
b[0].fecility = 30000;
b[0].hosfree = 1250;
```

```
//
```

```
b[1].bida = "02";
b[1].bname = "Kalana";
b[1].lab = 1250.00;
b[1].doctor = 1500;
b[1].medicine = 1263;
b[1].fecility = 30000;
b[1].hosfree = 1250;
```

```
//
```

```
b[2].bida = "03";
b[2].bname = "Kalana";
b[2].lab = 1250.00;
b[2].doctor = 1500;
b[2].medicine = 1263;
b[2].fecility = 30000;
b[2].hosfree = 1250;
```

```
//
```

```
b[3].bida = "04";
b[3].bname = "Kalana";
b[3].lab = 1250.00;
b[3].doctor = 1500;
b[3].medicine = 1263;
b[3].fecility = 30000;
b[3].hosfree = 1250;
//password
```

```
userPassword up = new userPassword();
Scanner in = new Scanner(System.in);
int un;
int pw;
do {
    try {
```

```
        System.out.print("Enter the user name:");
        un = in.nextInt();
```

```

        up.setUserName(un);
        System.out.print("Enter the password:");
        pw = in.nextInt();
        up.setPassword(pw);

        if ((up.getUserName() == 1111) && (up.getPassword() == 1833)) {
            System.out.println("user name and password is exist:");
        } else {
            System.out.println("re enter the user name and password");
        }
    } catch (Exception e) {
        System.out.println("There is an error");
    }
} while (!(up.getUserName() == 1111) && (up.getPassword() == 1833)));

//end password
//en class initialization
try {
    Scanner input = new Scanner(System.in);

    int choice, j, c1, status = 1;
    int s1, s2, s3, s4, s5, s6, s7;
    while (status == 1) {
        System.out.println("\n                                MAIN MENU");
        System.out.println("-----");
        System.out.println("1.Doctor 2. Patients 3.Medicines 4.Laboratories 5. Facilities 6. Staff 7.Bill");
        System.out.println("-----");

        choice = input.nextInt();

        switch (choice) {
            //doctor section
            case 1: {
                System.out.println("-----");
                System.out.println("                                **DOCTOR SECTION**");
                System.out.println("-----");
                s1 = 1;
                while (s1 == 1) {
                    System.out.println("1).Add New Entry\n 2).Existing Doctors List");
                    c1 = input.nextInt();
                    switch (c1) {
                        case 1: {
                            d[count1].new_doctor();
                            count1++;
                            break;
                        }
                        case 2: {
                            System.out.println("-----");

                            System.out.println("id \t Name\t Specilist \t Timing \t Qualification \t Room No.");
                            System.out.println("-----");

                            for (j = 0; j < count1; j++) {
                                d[j].doctor_info();
                            }
                            break;
                        }
                    }
                }
            }
        }
    }
}

```

```

        }
    }
    System.out.println("\nReturn to Back Press 1 and for Main Menu Press 0");
    s1 = input.nextInt();
}
break;
}
//patient section
case 2: {
    System.out.println("-----");
-   ");
    System.out.println("                **PATIENT SECTION**");
    System.out.println("-----");
---");
    s2 = 1;
    while (s2 == 1) {
        System.out.println("1.Add New Entry\n2.Existing Patients List");
        c1 = input.nextInt();
        switch (c1) {
            case 1: {
                p[count2].new_patient();
                count2++;
                break;
            }
            case 2: {
                System.out.println("-----");
-----");
                System.out.println("id \t Name \t Disease \t Gender \t Admit Status \t Age");
                System.out.println("-----");
-----");
                for (j = 0; j < count2; j++) {
                    p[j].patient_info();
                }
                break;
            }
        }
        System.out.println("\nReturn to Back Press 1 and for Main Menu Press 0");
        s2 = input.nextInt();
    }
    break;
}
//medicine section
case 3: {
    s3 = 1;
    System.out.println("-----");
-----");
    System.out.println("                **MEDICINE SECTION**");
    System.out.println("-----");
-----");
    while (s3 == 1) {
        System.out.println("1.Add New Entry\n2. Existing Medicines List");
        c1 = input.nextInt();
        switch (c1) {
            case 1: {
                m[count3].new_medi();
                count3++;

```

```

        break;
    }
    case 2: {
        System.out.println("-----");
        System.out.println("Name \t Company \t Expiry Date \t Cost");
        System.out.println("-----");
        for (j = 0; j < count3; j++) {
            m[j].find_medi();
        }
        break;
    }
}
System.out.println("\nReturn to Back Press 1 and for Main Menu Press 0");
s3 = input.nextInt();
}
break;
}
//laborartory
case 4: {
    s4 = 1;
    System.out.println("-----");
    System.out.println("                **LABORATORY SECTION**");
    System.out.println("-----");
    while (s4 == 1) {
        System.out.println("1.Add New Entry \n2.Existing Laboratories List");
        c1 = input.nextInt();
        switch (c1) {
            case 1: {
                l[count4].new_feci();
                count4++;
                break;
            }
            case 2: {
                System.out.println("-----");
                System.out.println("Facilities\t\t Cost");
                System.out.println("-----");
                for (j = 0; j < count4; j++) {
                    l[j].feci_list();
                }
                break;
            }
        }
        System.out.println("\nReturn to Back Press 1 and for Main Menu Press 0");
        s4 = input.nextInt();
    }
    break;
}
//hospital facility
case 5: {
    s5 = 1;

```

```

System.out.println("-----");
");
System.out.println("        **HOSPITAL FACILITY SECTION**");
System.out.println("-----");
");
while (s5 == 1) {
    System.out.println("1.Add New Facility\n2.Existing Fecilities List");
    c1 = input.nextInt();
    switch (c1) {
        case 1: {
            f[count5].add_feci();
            count5++;
            break;
        }
        case 2: {
            System.out.println("-----");
            System.out.println("Hospital Facility are:");
            System.out.println("-----");
            for (j = 0; j < count5; j++) {
                f[j].show_feci();
            }
            break;
        }
    }
    System.out.println("\nReturn to Back Press 1 and for Main Menu Press 0");
    s5 = input.nextInt();
}
break;
}
//staff section
case 6: {
    s6 = 1;
    System.out.println("-----");
    System.out.println("        **STAFF SECTION**");
    System.out.println("-----");
    while (s6 == 1) {
        String a = "nurse", bo = "worker", c = "security";
        System.out.println("1.Add New Entry \n2.Existing Nurses List\n3.Existing Workers List \n4.Existing
Security List");
        c1 = input.nextInt();
        switch (c1) {
            case 1: {
                s[count6].new_staff();
                count6++;
                break;
            }
            case 2: {
                System.out.println("-----");
                System.out.println("id \t Name \t Gender \t Salary");
                System.out.println("-----");
            }

```



```

        for (j = 0; j < count6; j++) {
            if (a.equals(s[j].desg))
                s[j].staff_info();
        }
        break;
    }
    case 3: {
        System.out.println("-----");
        System.out.println("id \t Name \t Gender \t Salary");
        System.out.println("-----");
        for (j = 0; j < count6; j++) {
            if (b.equals(s[j].desg))
                s[j].staff_info();
        }
        break;
    }
    case 4: {
        System.out.println("-----");
        System.out.println("id \t Name \t Gender \t Salary");
        System.out.println("-----");
        for (j = 0; j < count6; j++) {
            if (c.equals(s[j].desg))
                s[j].staff_info();
        }
        break;
    }
    }
    System.out.println("\nReturn to Back Press 1 and for Main Menu Press 0");
    s6 = input.nextInt();
}
break;
}
default: {
    System.out.println(" You Have Enter Wrong Choice!!!");
}
//bill
{
    System.out.println("-----");
    System.out.println("                **BILL SECTION**");
    System.out.println("-----");
    s7 = 1;
    while (s7 == 1) {
        System.out.println("1).Add New Entry\n 2).Existing Payment");
        c1 = input.nextInt();
        switch (c1) {
            case 1: {
                b[count7].new_bill();
                count7++;
                break;
            }
        }
    }
}

```

```

        case 2: {
            // lab,doctor,medicine,fecility,hosfree,totalpay;
            System.out.println("-----");
            System.out.println("id \t Name\t lab \t doctor \t medicine \t fecility \thospitalfree \ttotalpay");
            System.out.println("-----");
            for (j = 0; j < count7; j++) {
                b[j].bill_info();
            }
            break;
        }
        System.out.println("\nReturn to Back Press 1 and for Main Menu Press 0");
        s1 = input.nextInt();
    }
    break;
}
//end bill
}
System.out.println("\nReturn to MAIN MENU Press 1");
status = input.nextInt();
}
} catch (Exception e1){
    System.out.println(e1);
}
}
}

```

Bill.java

```
package com.company;

import java.util.Scanner;

public class bill {
    String bida, bname;
    double lab,doctor,medicine,fecility,hosfree,totalpay=0;
    void new_bill()
    {
        Scanner input = new Scanner(System.in);
        System.out.print("id:-");
        bida = input.nextLine();
        System.out.print("name:-");
        bname = input.nextLine();
        System.out.print("labcost:-");
        lab=input.nextDouble();
        System.out.print("doctor charge:-");
        doctor=input.nextDouble();
        System.out.print("medicine:-");
        medicine=input.nextDouble();
        System.out.print("facility:-");
        fecility=input.nextDouble();
        System.out.print("hospital free");
        hosfree=input.nextDouble();
        totalpay=(lab+doctor+medicine+fecility+hosfree)+totalpay;
    }
    public void bill_info()
    {
        System.out.println(bida + "\t" +bname + " \t" + lab + " \t" + doctor + " \t" + medicine + " \t" + fecility+"
\t"+hosfree+" \t" + totalpay);
    }
}
```

Doctor.java

```
package com.company;

import java.util.Scanner;

public class doctor extends methoddoo
{
    String did, dname, specilist, appoint, doc_qual;
    int droom;
    void new_doctor()
    {
        Scanner input = new Scanner(System.in);
        System.out.print("id:-");
        did = input.nextLine();
        System.out.print("name:-");
        dname = input.nextLine();
```

```

        System.out.print("specilization:-");
        specilist = input.nextLine();
        System.out.print("work time:-");
        appoint = input.nextLine();
        System.out.print("qualification:-");
        doc_qual = input.nextLine();
        System.out.print("room no.:-");
        droom = input.nextInt();
    }
    public void doctor_info()
    {
        System.out.println(did + "\t" + dname + " \t" + specilist + " \t" + appoint + " \t" + doc_qual + " \t" +
        droom);
    }
}

```

Fecility.java

```
package com.company;
```

```
import java.util.Scanner;
```

```
public class doctor extends methoddoc
```

```

{
    String did, dname, specilist, appoint, doc_qual;
    int droom;
    void new_doctor()
    {
        Scanner input = new Scanner(System.in);
        System.out.print("id:-");
        did = input.nextLine();
        System.out.print("name:-");
        dname = input.nextLine();
        System.out.print("specilization:-");
        specilist = input.nextLine();
        System.out.print("work time:-");
        appoint = input.nextLine();
        System.out.print("qualification:-");
        doc_qual = input.nextLine();
        System.out.print("room no.:-");
        droom = input.nextInt();
    }
    public void doctor_info()
    {
        System.out.println(did + "\t" + dname + " \t" + specilist + " \t" + appoint + " \t" + doc_qual + " \t" +
        droom);
    }
}

```

lab.java

```
package com.company;

import java.util.Scanner;

class lab
{
    String fecility;
    int lab_cost;
    void new_feci()
    {
        Scanner input = new Scanner(System.in);
        System.out.print("fecility:-");
        fecility = input.nextLine();
        System.out.print("cost:-");
        lab_cost = input.nextInt();
    }
    void feci_list()
    {
        System.out.println(fecility + "\t\t" + lab_cost);
    }
}
```

Medical.java

```
package com.company;

import java.util.Scanner;

class medical
{
    String med_name, med_comp, exp_date;
    int med_cost, count;
    void new_medi()
    {
        Scanner input = new Scanner(System.in);
        System.out.print("name:-");
        med_name = input.nextLine();
        System.out.print("comp:-");
        med_comp = input.nextLine();
        System.out.print("exp_date:-");
        exp_date = input.nextLine();
        System.out.print("cost:-");
        med_cost = input.nextInt();
        System.out.print("no of unit:-");
        count = input.nextInt();
    }
    void find_medi()
    {

```

```
        System.out.println("med_name + " \t" + med_comp + " \t" + exp_date + " \t" + med_cost);
    }
}
```

Methfeci.java

```
package com.company;

abstract class methfeci {
    abstract void add_feci();
    abstract void show_feci();
}
```

Methoddoc.java

```
package com.company;

abstract public class methoddoc {
    abstract void new_doctor();
    abstract void doctor_info();
}
```

Methodpat.java

```
package com.company;

public interface methodpat {
    void new_patient();
    void patient_info();
}
```

Patient.java

```
package com.company;

import java.util.Scanner;

class patient implements methodpat
{
    String pid, pname, disease, sex, admit_status;
    int age;
    public void new_patient()
    {
```

```

Scanner input = new Scanner(System.in);
System.out.print("id:-");
pid = input.nextLine();
System.out.print("name:-");
pname = input.nextLine();
System.out.print("disease:-");
disease = input.nextLine();
System.out.print("sex:-");
sex = input.nextLine();
System.out.print("admit_status:-");
admit_status = input.nextLine();
System.out.print("age:-");
age = input.nextInt();
}
public void patient_info()
{
    System.out.println(pid + "\t" + pname + "\t" + disease + "\t" + sex + "\t" + admit_status + "\t" + age);
}
}

```

Staff.java

```

package com.company;

import java.util.Scanner;

class staff
{
    String sid, sname, desg, sex;
    int salary;
    void new_staff()
    {
        Scanner input = new Scanner(System.in);
        System.out.print("id:-");
        sid = input.nextLine();
        System.out.print("name:-");
        sname = input.nextLine();
        System.out.print("designation:-");
        desg = input.nextLine();
        System.out.print("sex:-");
        sex = input.nextLine();
        System.out.print("salary:-");
        salary = input.nextInt();
    }
    void staff_info()
    {
        System.out.println(sid + "\t" + sname + "\t" + sex + "\t" + salary);
    }
}

```

```
}  
}
```

Userpassword.java

```
package com.company;
```

```
public class userPassword {  
    private int userName;  
    private int password;
```

```
  
    public int getUserName() {  
        return userName;  
    }
```

```
  
    public void setUserName(int userName) {  
        this.userName=userName;  
    }
```

```
  
    public int getPassword() {  
        return password;  
    }
```

```
  
    public void setPassword(int password) {  
        this.password = password;  
  
    }
```