**Object Oriented Programming CS1004**

**Assignment 02**

**Name: Muhammad Shahbaz Hassan**

**Roll Number: 24K-0884**

**Section: BCS-2C**

**Scenario 1**

**Code**

*/\**

*Name: Muhammad Shahbaz Hassan*

*Roll Number: 24K-0884*

*Class: BCS\_2C*

*\*/*

#include<iostream>

#include<vector>

using namespace std;

class Bus;

class Person;

class Route;

class Route{

    int id;

    vector<string> stops;

    Bus\* bus;

    public:

    Route(){}

    Route(int x):id(x),bus(nullptr){}

    void add\_stop(string s){

        stops.push\_back(s);

    }

    void remove\_stop(string s){

        for(int i=0;i<stops.size();i++){

            if(s**==**stops**[**i**]**){

                stops.erase(stops.begin()**+**i);

            }

        }

    }

    void add\_bus(Bus\* b){

        bus=b;

    }

    int get\_id(){

        return id;

    }

    void display(){

        cout**<<**"\nRoute ID: "**<<***this*->id**<<**" | ";

        if(stops.size()!=0){

            for(int j=0;j<*this*->stops.size();j++){

                cout**<<***this*->stops**[**j**]**;

                if(j<stops.size()-1){

                    cout**<<**" --> ";

                }

            }

            cout**<<**endl;

        }

        else{

            cout**<<**"Stops are not Assigned"**<<**endl;

        }

    }

    bool **operator==**(Route& r){

        bool check=false;

**if**(this->stops.size()==r.stops.size()){

            for(int i=0,j=0;i<*this*->stops.size();i++,j++){

                if(*this*->stops**[**i**]==**r.stops**[**j**]**){

                    check=true;

                }

                else{

                    check=false;

                    break;

                }

            }

        }

**if**(check==true) return true;

        else return false;

    }

    friend ostream& **operator<<**(ostream& out, Route& r);

};

class Bus{

    int busId;

    Route\* route;

    vector<Person\*> passenger;

    public:

    Bus(){}

    Bus(int x):busId(x),route(nullptr){}

    void add\_route(Route\* a){

        route=a;

    }

    void display\_bus();

    void add\_passenger(Person\* p){

        passenger.push\_back(p);

    }

    int get\_id(){

        return busId;

    }

    Route\* get\_route(){

            return route;

    }

};

class Person{

    string person\_type;

    int id;

    string name;

    int age;

    string pickup;

    string dropoff;

    Bus\* bus;

    Route\* r;

    bool transportation\_card;

    bool pay\_fee;

    bool attendance;

    public:

    Person(){}

    Person(string t,int a, string b, int c,string p, string d):person\_type(t),id(a),name(b),age(c),pickup(p),dropoff(d),bus(nullptr),transportation\_card(false),pay\_fee(false),attendance(false),r(nullptr){}

    void add\_bus(Bus\* b){

        bus=b;

        b->add\_passenger(*this*);

    }

    void add\_route\_person(Route\* r){

*this*->r=new Route(\*r);

    }

    virtual void fee\_pay(){

        pay\_fee=true;

        transportation\_card=true;

    }

    int get\_id(){

        return id;

    }

    string get\_name(){

        return name;

    }

    string get\_type(){

        return person\_type;

    }

    bool get\_transportation\_card(){

        return transportation\_card;

    }

    Route\* get\_route(){

        return r;

    }

    virtual void display(ostream& out){

        out**<<**"\nType: "**<<**person\_type**<<**endl;

        out**<<**"ID: "**<<**id**<<**endl;

        out**<<**"Name: "**<<**name**<<**endl;

        out**<<**"Age: "**<<**age**<<**endl;

        out**<<**"Pickup Stop: "**<<**pickup**<<**endl;

        out**<<**"Dropoff Stop: "**<<**dropoff**<<**endl;

        cout**<<**"Attendance: ";

        if(attendance==false) cout**<<**"Not Marked"**<<**endl;

        else cout**<<**"Marked"**<<**endl;

        if(bus!=nullptr) bus->display\_bus();

        else cout**<<**"Bus: Not Assigned"**<<**endl;

        if(r!=nullptr){

            out**<<**r;

        }

        out**<<**"Fee Status: ";

        if(pay\_fee==true){

            out**<<**"Payed"**<<**endl;

        }

        else{

            out**<<**"Not Payed"**<<**endl;

        }

        cout**<<**"Transportation Card: ";

        if(transportation\_card==true){

            out**<<**"Active"**<<**endl;

        }

        else{

            out**<<**"Not Active"**<<**endl;

        }

    }

    void mark\_attendance(){

        if(*this*->transportation\_card==true){

            attendance=true;

            cout**<<**"Attendance Marked"**<<**endl;

        }

        else{

            cout**<<**"Card not Active Yet | Pay your fee"**<<**endl;

        }

    }

    friend ostream& **operator<<**(ostream& out, Person& p);

};

class Student: public Person{

    float bus\_semester\_fee;

    public:

    Student(){}

    Student(int a, string b, int c,string p, string d, int fee):Person("Student",a,b,c,p,d),bus\_semester\_fee(fee){}

    void fee\_pay() override{

        cout**<<**"Semester fee of "**<<**bus\_semester\_fee**<<**" payed"**<<**endl;

        Person::fee\_pay();

    }

    void display\_student(ostream& out){

        Person::display(out);

        cout**<<**"Semester Fee: "**<<**bus\_semester\_fee**<<**endl;

    }

};

class Teacher: public Person{

    float bus\_monthly\_fee;

    public:

    Teacher(){}

    Teacher(int a, string b, int c,string p, string d, int fee):Person("Teacher",a,b,c,p,d),bus\_monthly\_fee(fee){}

    void fee\_pay() override{

        cout**<<**"Monthly fee of "**<<**bus\_monthly\_fee**<<**" payed"**<<**endl;

        Person::fee\_pay();

    }

    void display\_teacher(ostream& out){

        Person::display(out);

        cout**<<**"Monthly Fee: "**<<**bus\_monthly\_fee**<<**endl;

    }

};

class Staff: public Person{

    float bus\_annual\_fee;

    public:

    Staff(){}

    Staff(int a, string b, int c,string p, string d, int fee):Person("Staff",a,b,c,p,d),bus\_annual\_fee(fee){}

    void fee\_pay() override{

        cout**<<**"Annual fee of "**<<**bus\_annual\_fee**<<**" payed"**<<**endl;

        Person::fee\_pay();

    }

    void display\_staff(ostream& out){

        Person::display(out);

        cout**<<**"Annual Fee: "**<<**bus\_annual\_fee**<<**endl;

    }

};

ostream& **operator<<**(ostream& out,Route& r){

    out**<<**"\nRoute ID: "**<<**r.id**<<**endl;

    if(r.bus!=nullptr)  cout**<<**"Bus ID: "**<<**r.bus->get\_id()**<<**endl;

    else cout**<<**"Currently Not Assigned to any Bus"**<<**endl;

    if(r.stops.size()!=0){

        for(int i=0;i<r.stops.size();i++){

            out**<<**r.stops**[**i**]**;

            if(i<r.stops.size()-1){

                out**<<**" --> ";

            }

        }

        out**<<**endl;

    }

    else{

        out**<<**"Stops are not Assigned"**<<**endl;

    }

    return out;

}

ostream& **operator<<**(ostream& out, Person& p){

    p.display(out);

    return out;

}

void Bus::display\_bus(){

    cout**<<**"\nBus ID: "**<<**busId**<<**endl;

    if(route!=nullptr){

        route->display();

    }

    else{

        cout**<<**"No Routes Assigned to this Bus"**<<**endl;

    }

    if(passenger.size()==0){

        cout**<<**"No Passengers alloted to this Bus"**<<**endl;

    }

    else{

        cout**<<**"Passenger Name with ID: "**<<**endl;

        for(int i=0;i<passenger.size();i++){

            cout**<<**"ID: "**<<**passenger**[**i**]**->get\_id()**<<**" Name: "**<<**passenger**[**i**]**->get\_name()**<<**" Passenger Type: "**<<**passenger**[**i**]**->get\_type()**<<**endl;

        }

    }

}

int main(){

    int n,n1,n2,age,fee,i,j;

    string name,pick,drop,type;

    bool check,check2,check3;

    Bus\*\* bus=new Bus\*[100];

    Person\*\* person=new Person\*[100];

    Route\*\* route=new Route\*[100];

    for(int i=0;i<100;i++){

        bus[i]=nullptr;

        person[i]=nullptr;

        route[i]=nullptr;

    }

    while(1){

        cout**<<**"\n1. Person\n2. Routes\n3. Bus\n4. Exit\nEnter the Corresponding Number: ";

        cin**>>**n;

        if(n==1){

            while(1){

                cout**<<**"\nMAIN -> PERSON\n1. Add a Person\n2. Display complete info of a Person\n3. Display ID, Name of All\n4. Pay Semester Fee\n5. Mark Attendance\n6. Select Route\n7. Select a Bus\n8. Exit\nEnter the Corresponding Number: ";

                cin**>>**n1;

                if(n1==1){

                    cout**<<**"Enter the Person Age: ";

                    cin**>>**age;

                    cout**<<**"Enter the Person Name: ";

                    cin**>>**name;

                    cout**<<**"Enter the Pickup Stop: ";

                    cin**>>**pick;

                    cout**<<**"Enter the Dropoff Stop: ";

                    cin**>>**drop;

                    while(1){

                        cout**<<**"Is the Person a:\n1. Teacher\n2. Student\n3. Staff Member\nEnter the Corresponding Number: ";

                        cin**>>**n2;

                        for(i=0;i<100;i++){

                            if(person[i]==nullptr) break;

                        }

                        if(n2==1){

                            cout**<<**"Enter the Monthly Bus Fee for the Teacher: ";

                            cin**>>**fee;

                            person[i]=new Teacher(i+1000,name,age,pick,drop,fee);

                            break;

                        }

                        else if(n2==2){

                            cout**<<**"Enter the Semester Bus Fee for the Student: ";

                            cin**>>**fee;

                            person[i]=new Student(i+1000,name,age,pick,drop,fee);

                            break;

                        }

                        else if(n2==3){

                            cout**<<**"Enter the Annual Bus Fee for the Staff Member: ";

                            cin**>>**fee;

                            person[i]=new Staff(i+1000,name,age,pick,drop,fee);

                            break;

                        }

                        else cout**<<**"Enter the Valid Number"**<<**endl;

                    }

                    cout**<<**"Person Added into the System"**<<**endl;

                    cout**<<**"\n --- Added Person Info ---";

                    cout**<<**\*person[i];

                }

                else if(n1==2){

                    check=false;

                    cout**<<**"Enter the ID of the Person whose Info to be Displayed: ";

                    cin**>>**age;

                    for(i=0;person[i]!=nullptr;i++){

                        if(age==person[i]->get\_id()){

                            check=true;

                            break;

                        }

                    }

                    if(check==true) cout**<<**\*person[age-1000];

                    else cout**<<**"No ID Found"**<<**endl;

                }

                else if(n1==3){

                    check=false;

                    for(i=0;person[i]!=nullptr;i++){

                        cout**<<**"Type: "**<<**person[i]->get\_type()**<<**" | ID: "**<<**person[i]->get\_id()**<<**" | Name: "**<<**person[i]->get\_name()**<<**endl;

                        check=true;

                    }

                    if(check==false) cout**<<**"Persons List Empty"**<<**endl;

                }

                else if(n1==4){

                    check=false;

                    cout**<<**"Enter the ID of the Person whose Fee is to be Payed: ";

                    cin**>>**age;

                    for(i=0;person[i]!=nullptr;i++){

                        if(i==age-1000){

                            check=true;

                            break;

                        }

                    }

                    if(check==true) person[age-1000]->fee\_pay();

                    else cout**<<**"No ID Found"**<<**endl;

                }

                else if(n1==5){

                    check=false;

                    cout**<<**"Enter the ID of the Person whose Fee is to be Payed: ";

                    cin**>>**age;

                    cout**<<**"Card Tapped..."**<<**endl;

                    for(i=0;person[i]!=nullptr;i++){

                        if(i==age-1000){

                            check=true;

                            break;

                        }

                    }

                    if(check==true) person[age-1000]->mark\_attendance();

                    else cout**<<**"No ID Found"**<<**endl;

                }

                else if(n1==6){

                    check=false;

                    cout**<<**"Enter the ID of the Person to select the route: ";

                    cin**>>**age;

                    for(i=0;person[i]!=nullptr;i++){

                        if(i==age-1000){

                            check=true;

                            break;

                        }

                    }

                    if(check==true){

                        if(person[i]->get\_transportation\_card()==1){

                            check=false;

                            check2=false;

                            for(j=0;route[j]!=nullptr;j++){

                                check=true;

                            }

                            if(check==true){

                                cout**<<**"Here is the Total List of Routes FAST Buses go to: ";

                                for(j=0;route[j]!=nullptr;j++){

                                    cout**<<**\*route[j];

                                }

                                cout**<<**"Enter the ID of the Route you chose: ";

                                cin**>>**age;

                                for(j=0;route[j]!=nullptr;j++){

                                    if(age==route[j]->get\_id()){

                                        person[i]->add\_route\_person(route[j]);

                                        cout**<<**"Route added to the person"**<<**endl;

                                    }

                                }

                            }

                            else cout**<<**"Routes are not Added in the System"**<<**endl;

                        }

                        else cout**<<**"Transportation Card Not Active Yet | Pay Fee"**<<**endl;

                    }

                    else cout**<<**"No ID Found"**<<**endl;

                }

                else if(n1==7){

                    check=false;

                    check2=false;

                    cout**<<**"Enter the ID of the Person to assign the bus: ";

                    cin**>>**age;

                    for(i=0;person[i]!=nullptr;i++){

                        if(i==age-1000){

                            check=true;

                            break;

                        }

                    }

                    if(check==true){

*// Route\* temp=person[i]->get\_route();*

                        if(person[i]->get\_transportation\_card()==1){

*// Route\* temp1=bus[j]->get\_route();*

                            cout**<<**"Here is the List of Buses Available to your route: "**<<**endl;

                            for(j=0;bus[j]!=nullptr;j++){

                                cout**<<**"\n";

                                bus[j]->display\_bus();

                                check2=true;

                            }

                            if(check2==true){

                                cout**<<**"Enter the ID of the Bus you want: ";

                                cin**>>**age;

                                for(j=0;bus[j]!=nullptr;j++){

                                    if(age==bus[j]->get\_id()){

                                            person[i]->add\_bus(bus[j]);

                                            cout**<<**"Bus Alloted to the Person"**<<**endl;

                                    }

                                }

                            }

                            else cout**<<**"No Bus Available for your Route"**<<**endl;

                        }

                        else cout**<<**"Transportation Card Not Active Yet | Pay Fee"**<<**endl;

                    }

                    else cout**<<**"No ID Found"**<<**endl;

                }

                else if(n1==8) break;

                else cout**<<**"Enter a Valid Number"**<<**endl;

            }

        }

        else if(n==2){

            while(1){

                cout**<<**"\nMAIN -> ROUTES\n1. Add a Route\n2. Display a Route Complete Information\n3. Display All Routes Id and stops\n4. Add a Stop\n5. Alot the route to a bus\n6. Exit\nEnter the Corresponding Number: ";

                cin**>>**n1;

                if(n1==1){

                    for(i=0;route[i]!=nullptr;i++){}

                    route[i] = new Route(i+1000);

                    cout**<<**"--- Stops to the Route ---"**<<**endl;

                    while(1){

                        cout**<<**"Enter the Stop name to add or enter zero to terminate stops adding: ";

                        cin**>>**name;

                        route[i]->add\_stop(name);

                        if(name**==**"0") break;

                    }

                    cout**<<**"Route Added to the System"**<<**endl;

                    cout**<<**\*route[i];

                }

                else if(n1==2){

                    check=false;

                    cout**<<**"Enter the Route ID: ";

                    cin**>>**age;

                    for(i=0;route[i]!=nullptr;i++){

                        if(age==route[i]->get\_id()){

                            check=true;

                            break;

                        }

                    }

                    if(check==true){

                        cout**<<**\*route[i];

                    }

                    else{

                        cout**<<**"No Route of this ID found"**<<**endl;

                    }

                }

                else if(n1==3){

                    check=false;

                    for(i=0;route[i]!=nullptr;i++){

                        route[i]->display();

                        check=true;

                    }

                    if(check==false) cout**<<**"Routes List Empty"**<<**endl;

                }

                else if(n1==4){

                    check=false;

                    cout**<<**"Enter the ID of the route in which to add a stop: ";

                    cin**>>**age;

                    for(i=0;route[i]!=nullptr;i++){

                        if(age==route[i]->get\_id()){

                            check=true;

                            break;

                        }

                    }

                    if(check==true){

                        cout**<<**"Enter the Name of the Stop to add: ";

                        cin**>>**name;

                        route[i]->add\_stop(name);

                        cout**<<**"Stop "**<<**name**<<**" Added to the Route"**<<**endl;

                    }

                    else cout**<<**"No Route of this ID found"**<<**endl;

                }

                else if(n1==5){

                    check=false;

                    check2=false;

                    cout**<<**"Enter the ID of the route: ";

                    cin**>>**age;

                    for(i=0;route[i]!=nullptr;i++){

                        if(age==route[i]->get\_id()){

                            check=true;

                            break;

                        }

                    }

                    if(check==true){

                        cout**<<**"Enter the ID of the bus: ";

                        cin**>>**age;

                        for(j=0;bus[j]!=nullptr;j++){

                            if(age==bus[j]->get\_id()){

                                check2=true;

                                break;

                            }

                        }

                        if(check2==true){

                            route[i]->add\_bus(bus[j]);

                            bus[j]->add\_route(route[i]);

                            cout**<<**"Bus added to the Route\nHere is the Complete Information of the Route"**<<**endl;

                            cout**<<**\*route[i];

                        }

                        else cout**<<**"No Bus of this ID found"**<<**endl;

                    }

                    else cout**<<**"No Route of this ID found"**<<**endl;

                }

                else if(n1==6) break;

                else cout**<<**"Enter a Valid Number"**<<**endl;

            }

        }

        else if(n==3){

           while(1){

                cout**<<**"\nMAIN -> BUS\n1. Add a Bus\n2. Display A Bus Complete Information\n3. Display All buses Id and route\n4. Exit\nEnter the Corresponding Number: ";

                cin**>>**n1;

                if(n1==1){

                    for(i=0;bus[i]!=nullptr;i++){}

                    bus[i]= new Bus(i+1000);

                    bus[i]->display\_bus();

                    cout**<<**"Note: The Passenger and Route is added to the Bus from the Route and Person Side"**<<**endl;

                }

                else if(n1==2){

                    check=false;

                    cout**<<**"Enter the Bus ID: ";

                    cin**>>**age;

                    for(i=0;bus[i]!=nullptr;i++){

                        if(age==bus[i]->get\_id()){

                            check=true;

                            break;

                        }

                    }

                    if(check==true){

                        bus[i]->display\_bus();

                    }

                    else{

                        cout**<<**"No Bus of this ID found"**<<**endl;

                    }

                }

                else if(n1==3){

                    for(i=0;bus[i]!=nullptr;i++){

                        cout**<<**"\nBus ID: "**<<**bus[i]->get\_id()**<<**endl;

                        if(bus[i]->get\_route()!=nullptr){

                            bus[i]->get\_route()->display();

                        }

                        else{

                            cout**<<**"No Route Alloted to the Bus"**<<**endl;

                        }

                    }

                }

                else if(n1==4) break;

                else cout**<<**"Enter a Valid Number"**<<**endl;

            }

        }

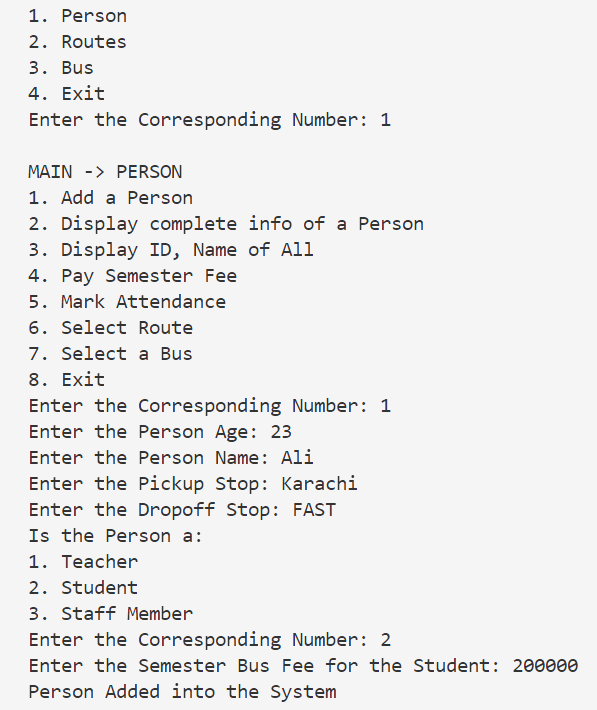
        else if(n==4) break;

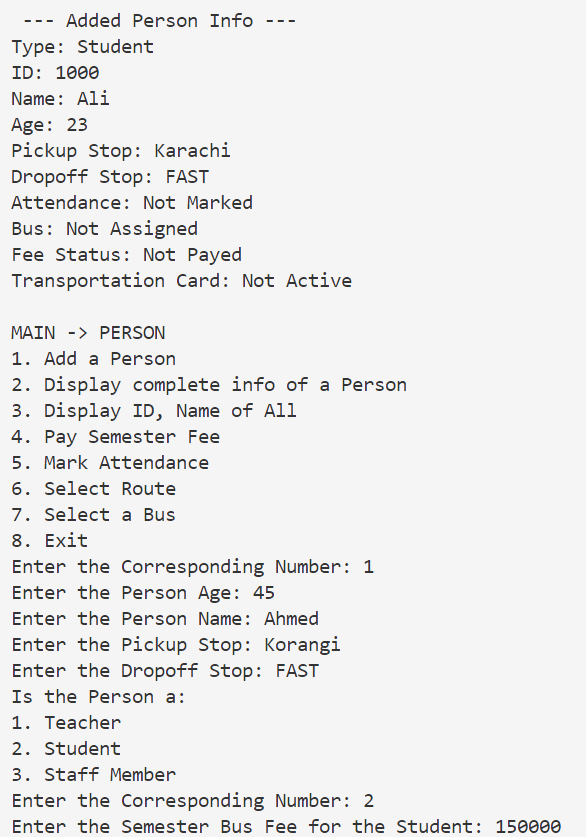
        else cout**<<**"Enter a Valid Number"**<<**endl;

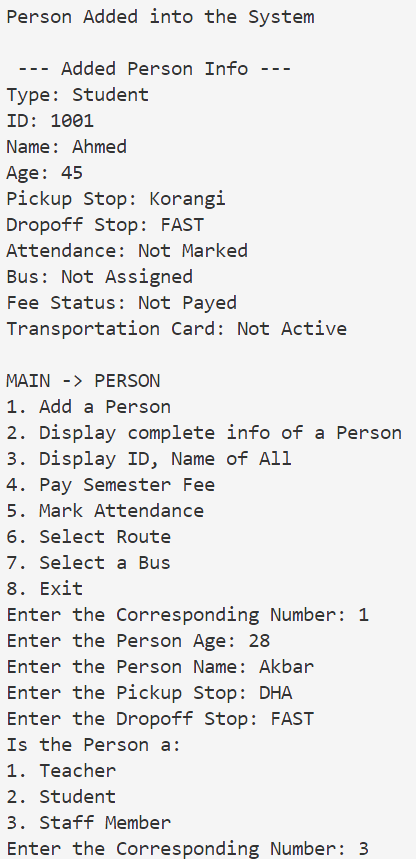
    }

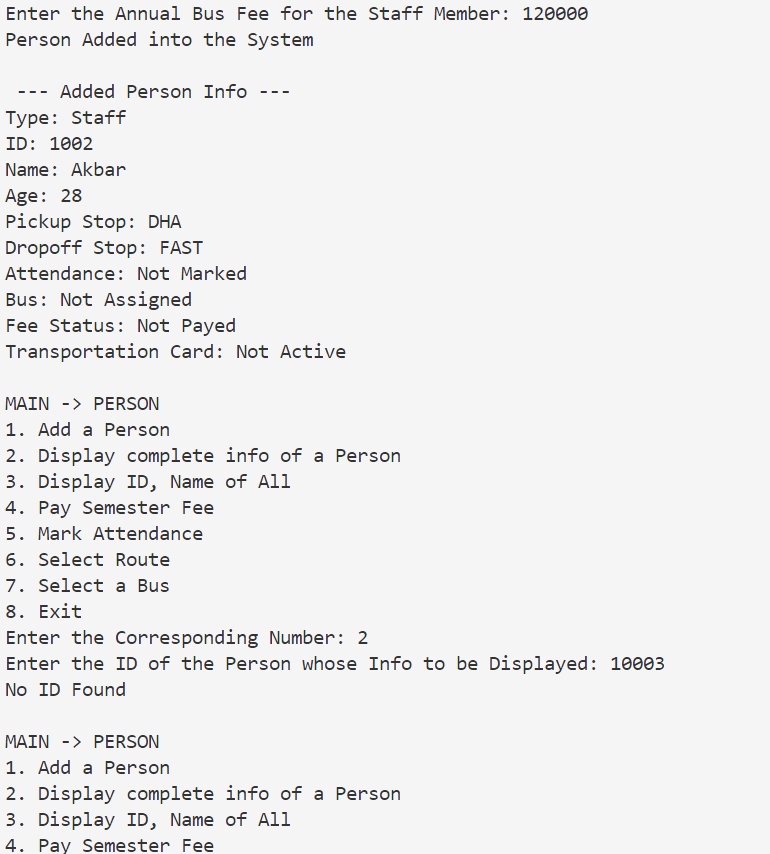
}

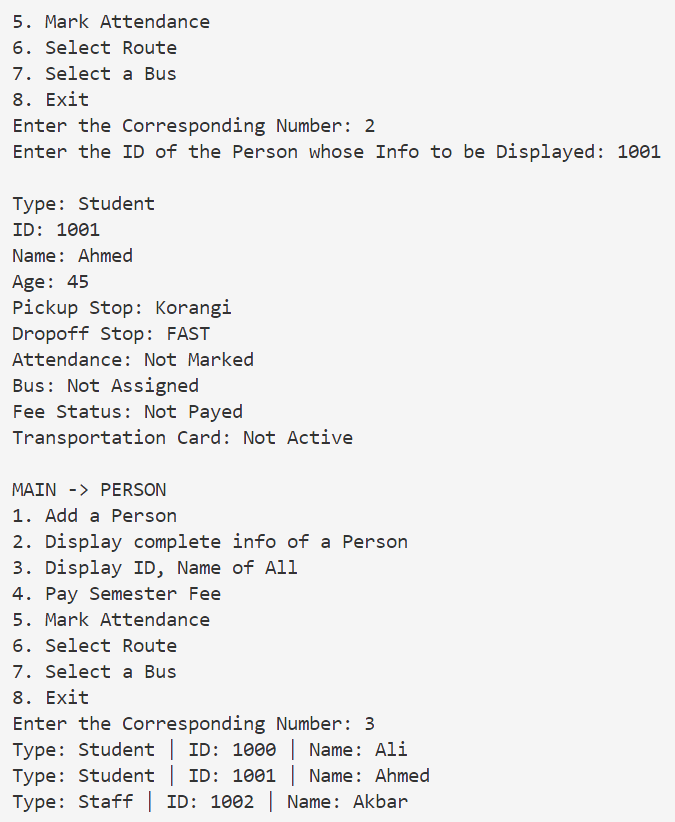
**Output Screenshots**

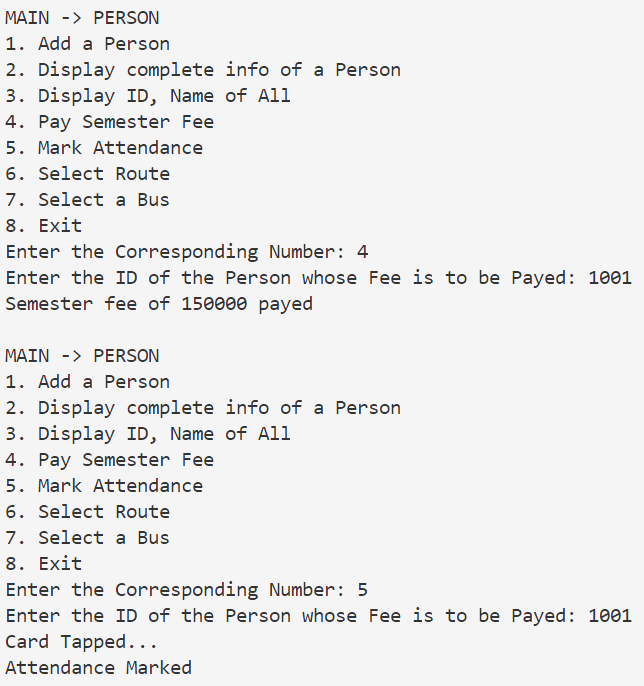
****

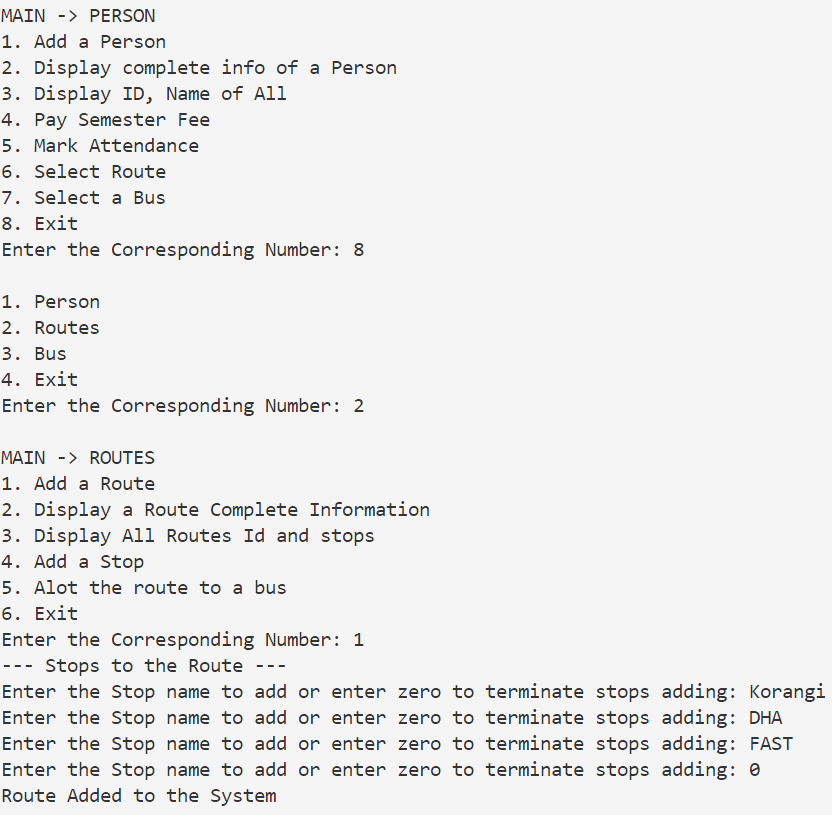
****

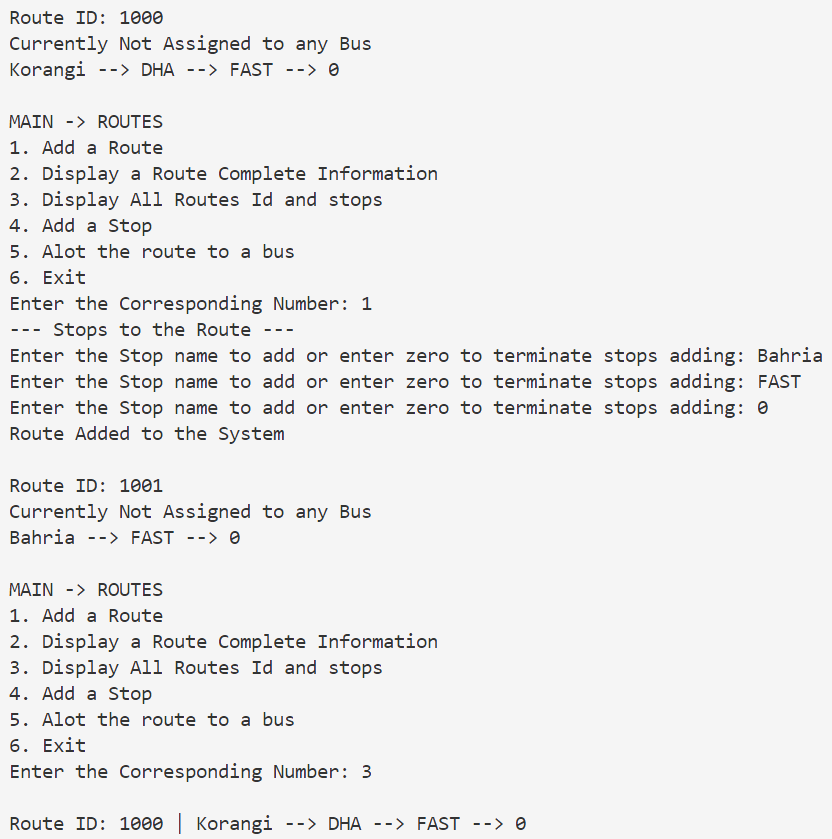
****

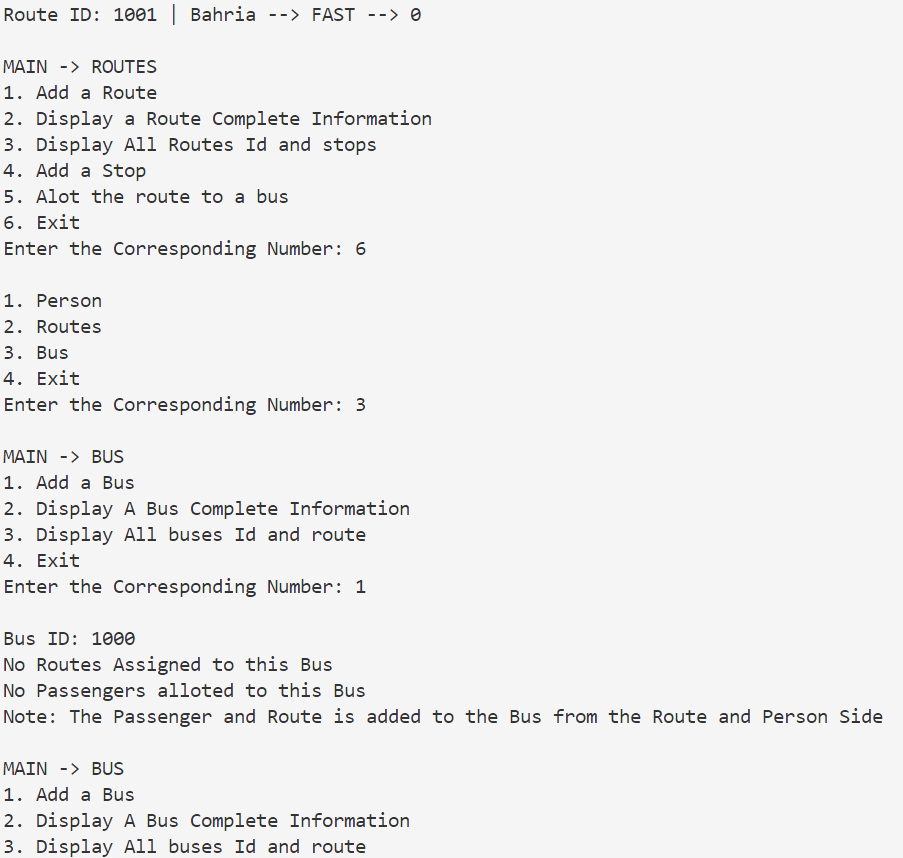
****

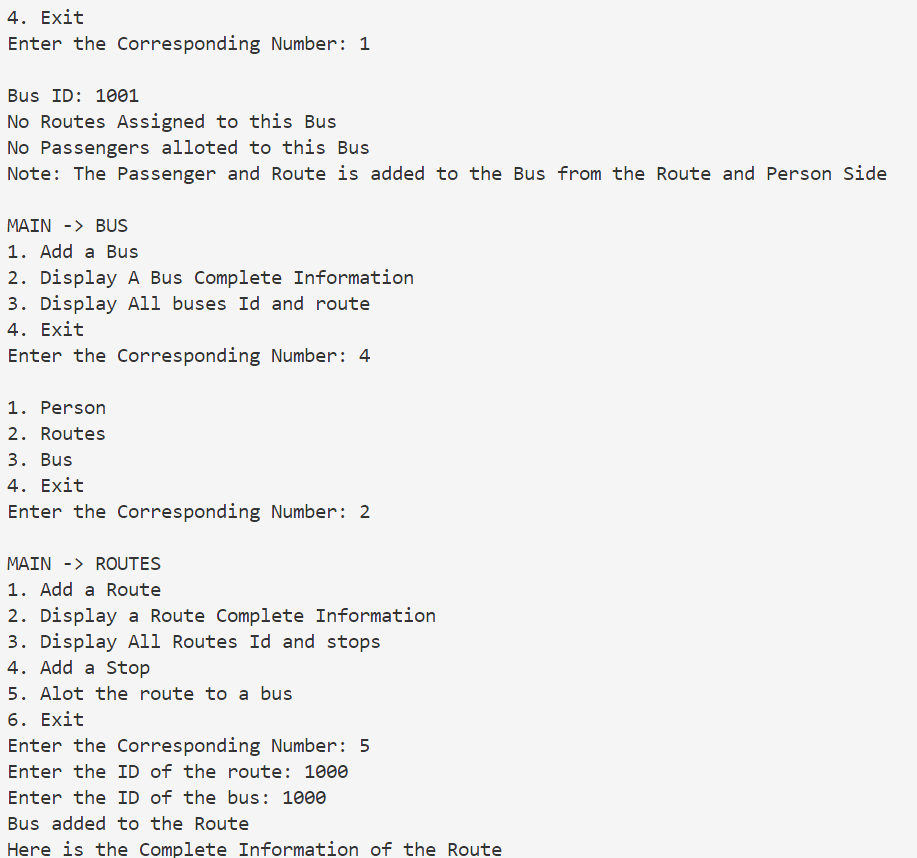
****

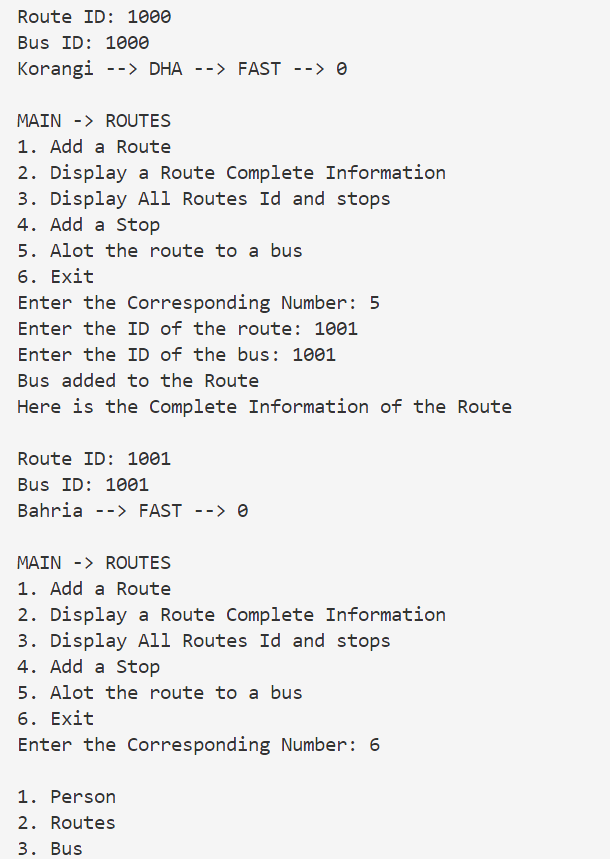
****

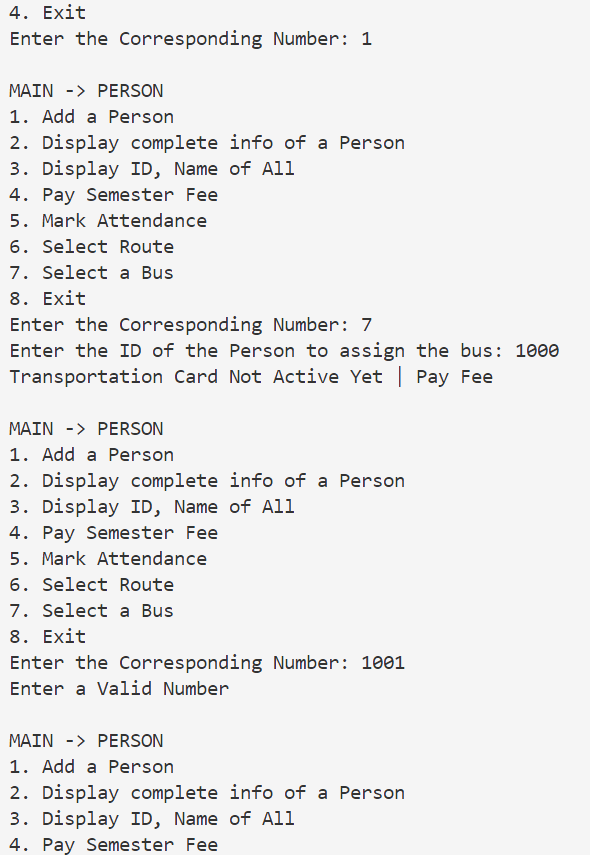
****

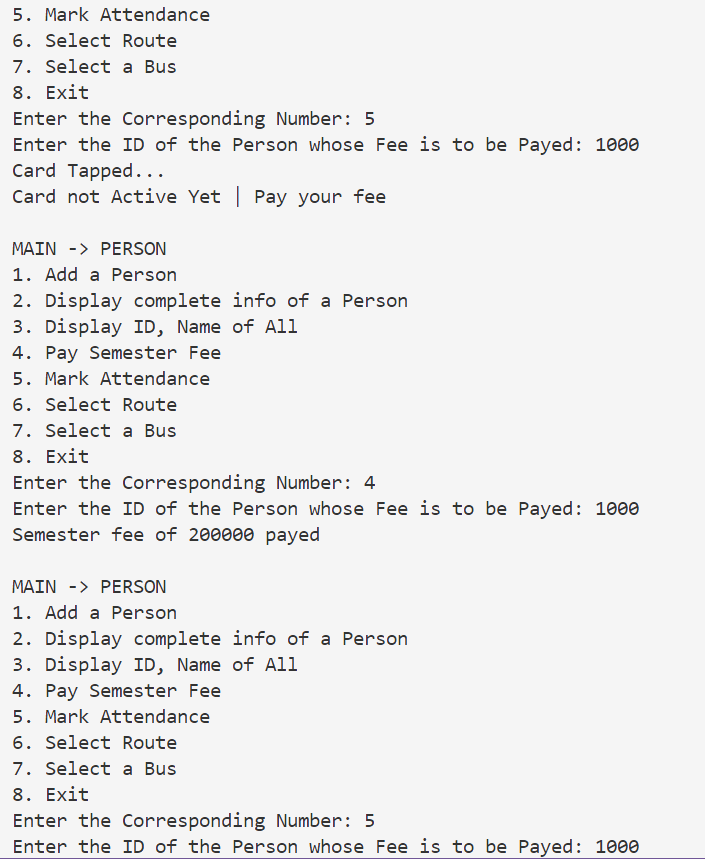
****

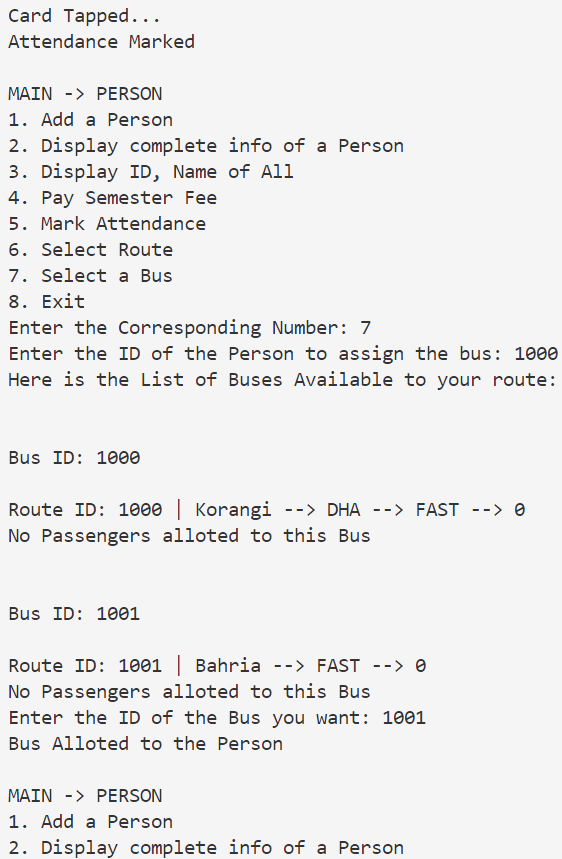
****

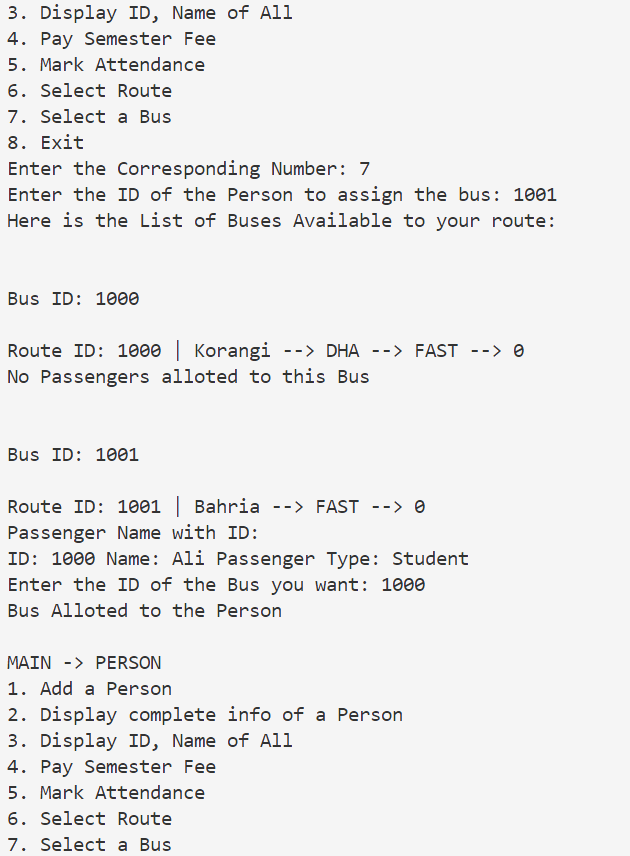
****

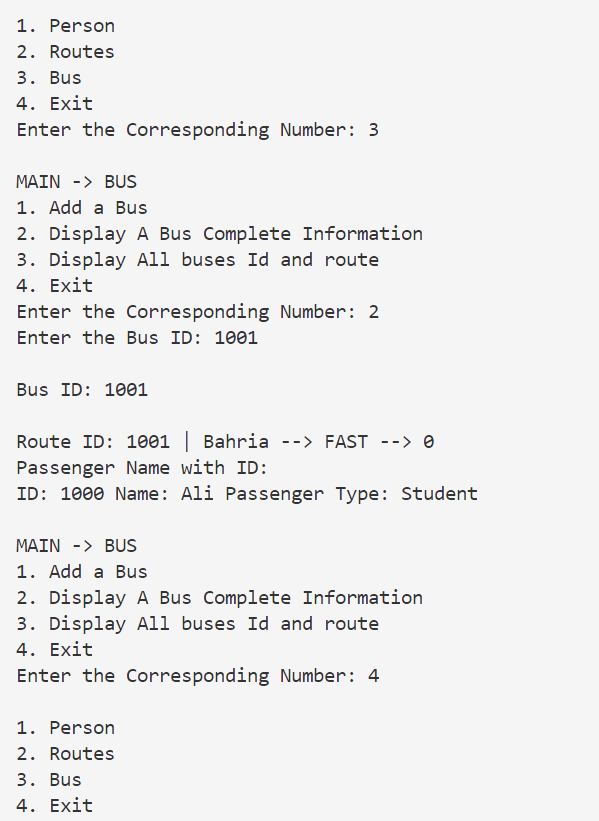




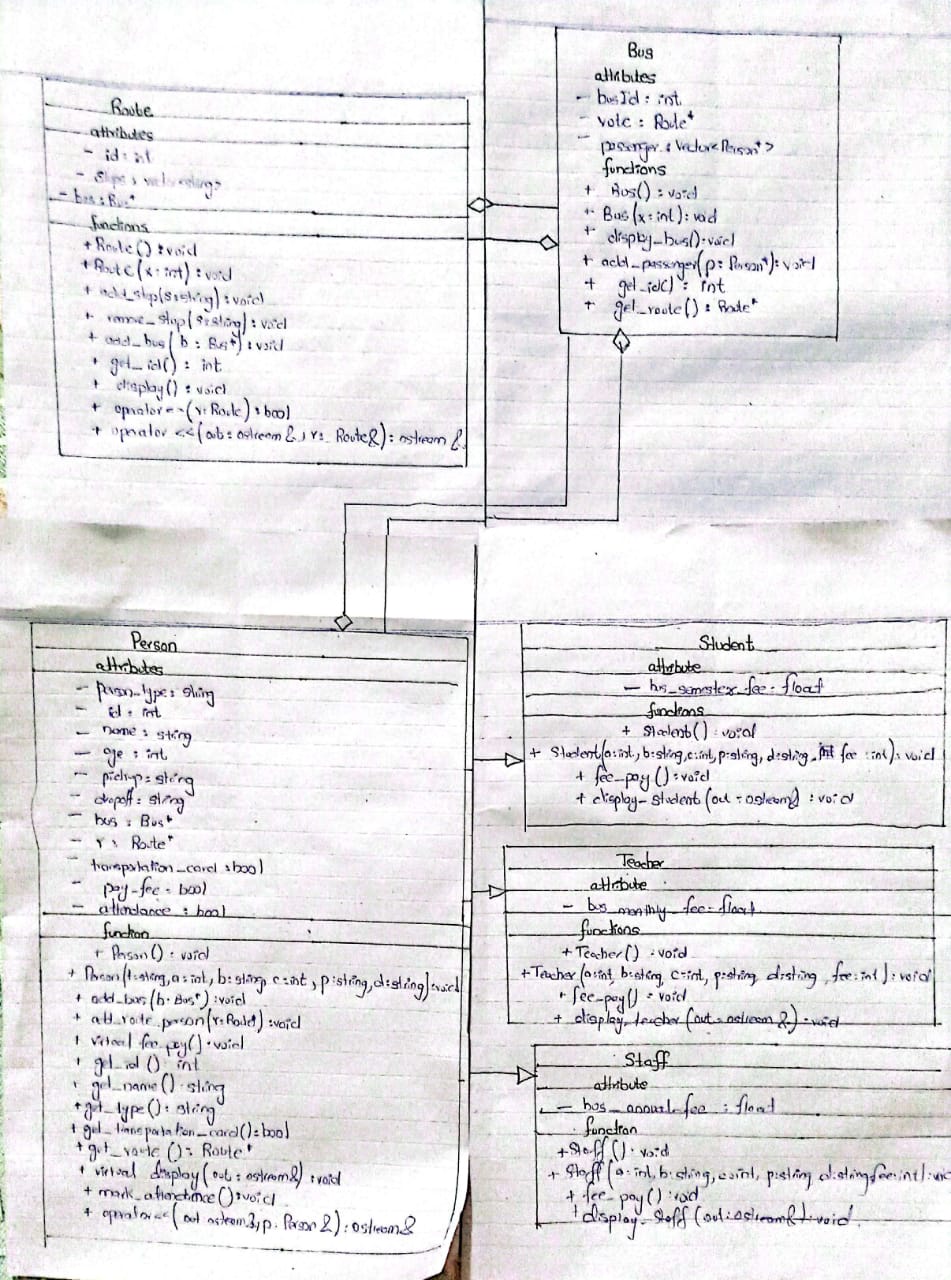








**Class Diagram**



**Scenario 2**

**Code**

*/\**

*Name: Muhammad Shahbaz Hassan*

*Roll Number: 24K-0884*

*Class: BCS\_2C*

*\*/*

#include<iostream>

#include<string>

using namespace **std**;

class **Ghost**;

class **HauntedHouse**;

class **Visitor**;

class **Ghost**{

**string** name;

    int level;

    public:

**Ghost**(){}

**Ghost**(**string** a, int b):**name**(a){

        if(b>=0&&b<=10){

            level=b;

        }

        else {

            cout**<<**"Enter the Level in the valid range 0-10\nIn this Case the Scare level of the ghost will be considered as 5"**<<endl**;

            level=5;

        }

    }

    virtual void **Haunting**(){}

**Ghost** **operator +** (const **Ghost**& g){

        return **Ghost**(this->name**+**g.name,this->level+g.level);

    }

**string** **get\_name**(){

        return name;

    }

    int **get\_level**(){

        return level;

    }

    friend **ostream**& **operator <<** (**ostream**& out,const **Ghost**& g);

};

class **Poltergeists**: virtual public **Ghost**{

    public:

**Poltergeists**(){}

**Poltergeists**(**string** a, int b):**Ghost**(a,b){}

    void **Haunting**() override{

        cout**<<**"Move Objects"**<<endl**;

    }

};

class **Banshees**: public **Ghost**{

    public:

**Banshees**(){}

**Banshees**(**string** a, int b):**Ghost**(a,b){}

    void **Haunting**() override{

        cout**<<**"Scream Loudly"**<<endl**;

    }

};

class **ShadowGhosts**: virtual public **Ghost**{

    public:

**ShadowGhosts**(){}

**ShadowGhosts**(**string** a, int b):**Ghost**(a,b){}

    void **Haunting**() override{

        cout**<<**"Whisper Creeply"**<<endl**;

    }

};

class **ShadowPoltergeist**:public **Poltergeists**, public **ShadowGhosts**{

    public:

**ShadowPoltergeist**(){}

**ShadowPoltergeist**(**string** a, int b):**Poltergeists**(a,b),**ShadowGhosts**(a,b){}

    void **Haunting**(){

**Poltergeists**::**Haunting**();

**ShadowGhosts**::**Haunting**();

    }

};

class **HauntedHouse**{

**Ghost**\*\* ghosts;

    int size;

    public:

**HauntedHouse**(){}

**HauntedHouse**(int n):**size**(n){

        ghosts=new **Ghost**\*[size];

    }

    void **add\_ghosts**(){

        int a,b;

**string** name;

        cout**<<**"--- Ghosts Information ---"**<<endl**;

        for(int i=0;i<size;i++){

            cout**<<**"\nGhost number: "**<<**i+1**<<endl**;

            cout**<<**"Enter the name of the Ghost: ";

            cin**>>**name;

            while(1){

                cout**<<**"Enter the scare Level of the ghost: ";

                cin**>>**b;

                if(b>=1&&b<=10) break;

                else{

                    cout**<<**"Scare Level must be between 1 and 10 inclusive"**<<endl**;

                }

            }

            while(1){

                cout**<<**"1. Poltergeists\n2. Banshees\n3. Shadow Ghosts\n4. ShadowPoltergeist"**<<endl**;

                cout**<<**"Enter the corresponding number: ";

                cin**>>**a;

                if(a==1){

                     ghosts[i]=new **Poltergeists**(name,b);

                     break;

                }

                else if(a==2){

                     ghosts[i]=new **Banshees**(name,b);

                     break;

                }

                else if(a==3){

                     ghosts[i]=new **ShadowGhosts**(name,b);

                     break;

                }

                else if(a==4){

                     ghosts[i]= new **ShadowPoltergeist**(name,b);

                     break;

                }

                else cout**<<**"Enter the Valid ghost type"**<<endl**;

            }

        }

    }

    void **display\_ghosts**(){

        cout**<<**"Ghosts Information: "**<<endl**;

        for(int i=0;i<size;i++){

            cout**<<**"Ghost Name: "**<<**ghosts[i]->**get\_name**()**<<endl**;

            cout**<<**"Ghost Scare level: "**<<**ghosts[i]->**get\_level**()**<<endl**;

        }

    }

    int **get\_size**(){

        return size;

    }

    friend void **Visit**(**HauntedHouse**& h, **Visitor** v[], int size);

};

class **Visitor**{

**string** name;

    int bravery;

    public:

**Visitor**(){}

**Visitor**(**string** a, int b):**name**(a),**bravery**(b){}

    friend void **Visit**(**HauntedHouse**& h, **Visitor** v[], int size);

};

**ostream**& **operator <<** (**ostream**& out, **Ghost**& g){

    out**<<**"Ghost Name: "**<<**g.**get\_name**()**<<endl**;

    out**<<**"Ghost Level: "**<<**g.**get\_level**()**<<endl**;

    return out;

}

void **Visit**(**HauntedHouse**& h, **Visitor** v[], int size) {

    int x,y;

    for(int i=0;i<size;i++){

        if(v[i].bravery>=1&&v[i].bravery<=4){

            x=1;

            y=4;

        }

        else if(v[i].bravery>=5 && v[i].bravery<=7 ){

            x=5;

            y=7;

        }

        else if(v[i].bravery>=8&& v[i].bravery<=10){

            x=8;

            y=10;

        }

        for(int j=0;j<h.size;j++){

            h.ghosts[j]->**Haunting**();

            if(h.ghosts[j]->**get\_level**()<x){

                cout**<<**"\nVisitor "**<<**v[i].name**<<**" laughs at the Ghost "**<<**h.ghosts[j]->**get\_name**()**<<endl**;

            }

            else if(h.ghosts[j]->**get\_level**()>=x&&h.ghosts[j]->**get\_level**()<=y){

                cout**<<**"\nVisitor "**<<**v[i].name**<<**" have a shaky voice but stays"**<<endl**;

            }

            else{

                cout**<<**"\nVisitor "**<<**v[i].name**<<**" screamed and Ran away"**<<endl**;

            }

        }

    }

}

int **main**(){

**Ghost**\* g1=new **Poltergeists**("Jake",1);

**Ghost**\* g2=new **Banshees**("John",2);

**Ghost** g3 = \*g1**+**\*g2;

    cout**<<**g3;

**Visitor**\* visitor=new **Visitor**[5];

    visitor[0]**=Visitor**("Jake",4);

    visitor[1]**=Visitor**("Jim",9);

    visitor[2]**=Visitor**("John",1);

    visitor[3]**=Visitor**("Kim",3);

    visitor[4]**=Visitor**("Tim",10);

**HauntedHouse** **h1**(5);

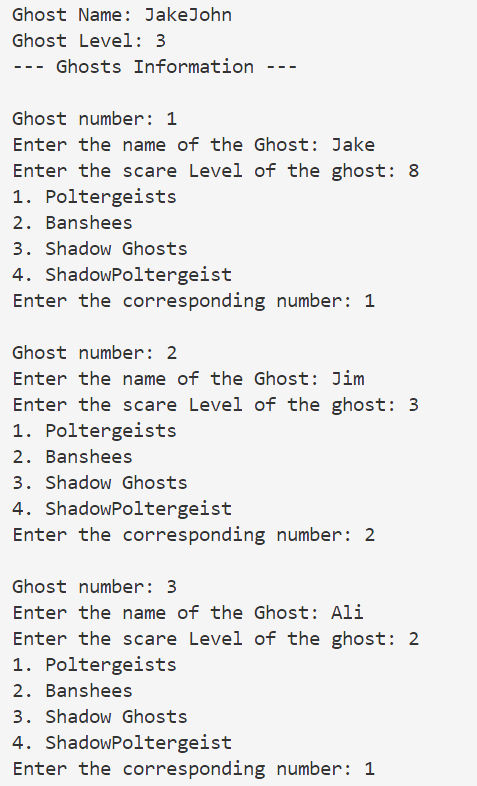
    h1.**add\_ghosts**();

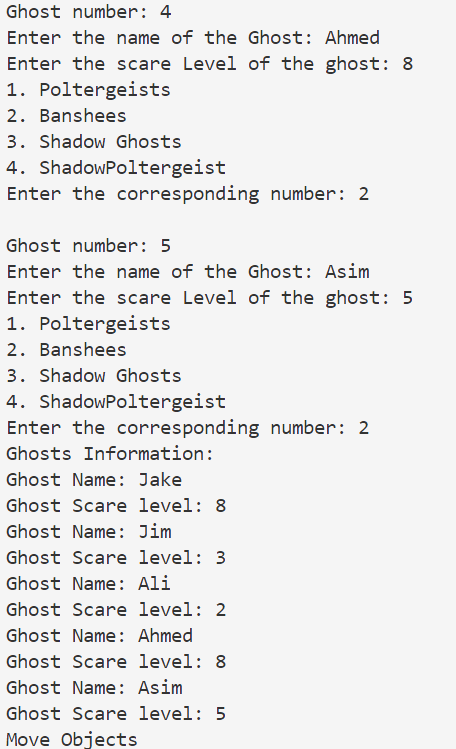
    h1.**display\_ghosts**();

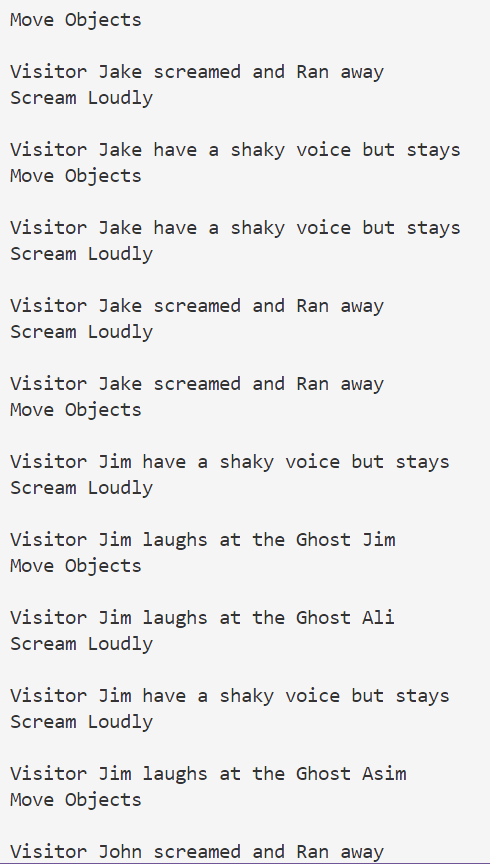
**Visit**(h1,visitor,5);

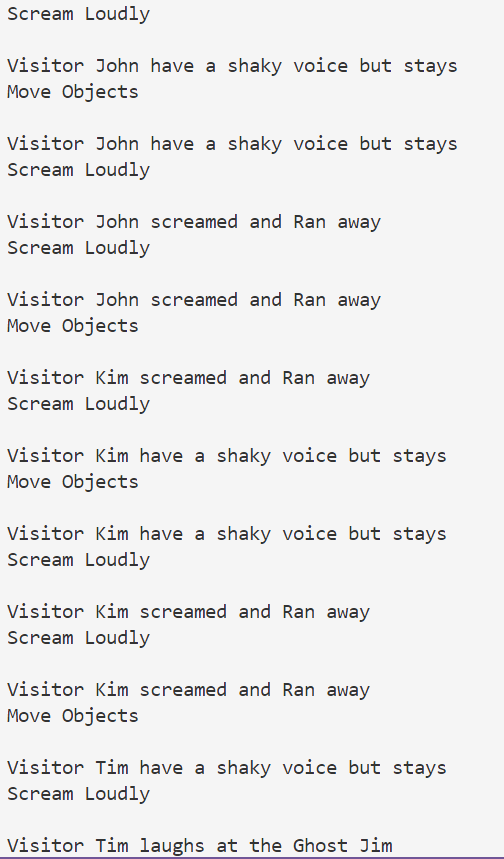
}

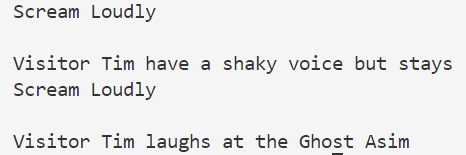
**Output Screenshot**











**Scenario 3**

**Code**

*/\**

*Name: Muhammad Shahbaz Hassan*

*Roll Number: 24K-0884*

*Class: BCS\_2C*

*\*/*

#include<iostream>

using namespace **std**;

class **Vehicle**{

    int id;

    static int total;

    int speed;

    int capacity;

    int energyEfficiency;

    public:

**Vehicle**(){}

**Vehicle**(int x,int y, int z, int i):**id**(x),**speed**(y),**capacity**(z),**energyEfficiency**(i){}

    virtual void **calculateRoute**(**string** s){}

    virtual void **command**(**string** a,int b ){

        cout<<"Package "<<a<<" with package id "<<b<<" will be delivered"<<endl;

    }

    virtual void **command**(**string** a,int b , **string** c){

        cout<<"Package "<<a<<" with package id "<<b<<" and urgency level "<<c<<" will be delivered"<<endl;

        total++;

    }

    bool operator==(const **Vehicle**& v){

        if(v.speed==this->speed&&v.energyEfficiency==this->energyEfficiency&&v.capacity==this->capacity) return true;

        else return false;

    }

    int **get\_id**(){

        return id;

    }

    int **get\_total**(){

        return total;

    }

    int **get\_speed**(){

        return speed;

    }

    int **get\_efficiency**(){

        return energyEfficiency;

    }

    int **get\_capacity**(){

        return capacity;

    }

    void **add\_total**(){

        total++;

    }

    friend **ostream**& operator << (**ostream**& out,const **Vehicle**& v);

    friend **Vehicle** **compareEfficieny**(**Vehicle**& v1, **Vehicle**& v2);

};

int **Vehicle**::total=0;

class **RamzanDrone**: public **Vehicle**{

    public:

**RamzanDrone**(){}

**RamzanDrone**(int x,int y, int z, int i):**Vehicle**(x,y,z,i){}

    void **calculateRoute**(**string** s){

        cout<<"Calculating Aerial Route to "<<s<<endl;

    }

    void **command**(**string** a, int b, **string** c) override {

        if(c=="urgent"){

            cout<<"Activating High Speed for the package delivery"<<endl;

            this->**add\_total**();

        }

        else{

**Vehicle**::**command**(a,b,c);

        }

    }

};

class **RamzanTimeShip**: public **Vehicle**{

    public:

**RamzanTimeShip**(){}

**RamzanTimeShip**(int x,int y, int z, int i):**Vehicle**(x,y,z,i){}

    void **calculateRoute**(**string** s){

        cout<<"Verfying Date... "<<s<<endl;

        cout<<"Date Verified. Now the Time ship can deliver"<<endl;

    }

    void **command**(**string** a, int b, **string** c){

        if(c=="urgent"){

            cout<<"Confirming Historical Accuracy before moving"<<endl;

            this->**add\_total**();

        }

        else{

**Vehicle**::**command**(a,b,c);

        }

    }

};

class **RamzanHyperPod**: public **Vehicle**{

    public:

**RamzanHyperPod**(){}

**RamzanHyperPod**(int x,int y, int z, int i):**Vehicle**(x,y,z,i){}

    void **calculateRoute**(**string** s){

        cout<<"Navigating an Underground Route to "<<s<<endl;

    }

};

**Vehicle** **compareEfficieny**(**Vehicle**& v1, **Vehicle**& v2){

    Vehicle **temp**(-1,-1,-1,-1);

    if(v1==v2){

        return temp;

    }

    else{

        if(v1.energyEfficiency>v2.energyEfficiency) return v1;

        else return v2;

    }

}

**ostream**& operator << (**ostream**& out, **Vehicle**& v){

    out<<"Vehicle id: "<<v.**get\_id**()<<endl;

    out<<"Vehicle Speed: "<<v.**get\_speed**()<<endl;

    out<<"Vehicle Capacity: "<<v.**get\_capacity**()<<endl;

    out<<"Vehicle Efficiency: "<<v.**get\_efficiency**()<<endl;

    out<<"Packages: "<<v.**get\_total**()<<endl;

    return out;

}

int **main**(){

    Vehicle\*\* vehicle = new Vehicle\*[3];

    vehicle[0]=new **RamzanDrone**(1,200,45,400);

    vehicle[1]=new **RamzanTimeShip**(1,200,45,400);

    vehicle[2]=new **RamzanHyperPod**(1,100,30,400);

    vehicle[0]->**calculateRoute**("Dubai");

    vehicle[0]->**command**("Dubai",101,"urgent");

    vehicle[1]->**calculateRoute**("London");

    vehicle[1]->**command**("London",102,"urgent");

    vehicle[2]->**calculateRoute**("Berlin");

    if(\*vehicle[0]==\*vehicle[1]){

        cout<<"Vehicle 0 and Vehicle 1 are same"<<endl;

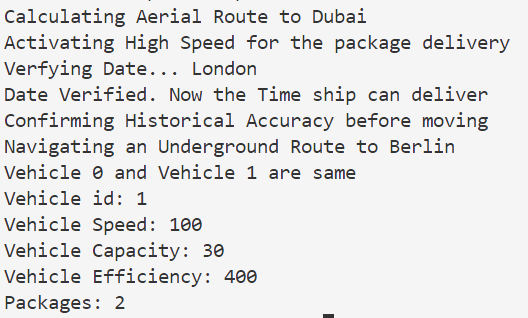
    }

    Vehicle v3=**compareEfficieny**(\*vehicle[0],\*vehicle[2]);

    cout<<v3;

}

**Output Screenshot**

****

**Scenario 4**

**Code**

*/\**

*Name: Muhammad Shahbaz Hassan*

*Roll Number: 24K-0884*

*Class: BCS\_2C*

*\*/*

#include<iostream>

#include<vector>

using namespace **std**;

class **User**;

bool **authenticateAndPerformAction**(**User**\* user, **string** action);

long long **generate\_hashes**(**string** a){

    long long hash=5381;

    for(int i=0;i<a.**size**();i++){

        hash=hash\*33+a**[**i**]**;

    }

    return hash;

}

class **User**{

**string** name;

    int id;

**vector**<**string**> permissions;

**string** email;

    long long hashed\_password;

    public:

**User**(){}

**User**(int d, **string** a, **string** b,**string** c):**id**(d),**name**(a),**email**(b){

        this->hashed\_password=**generate\_hashes**(c);

    }

    void **add\_permissions**(**string** a){

        permissions.**push\_back**(a);

    }

    int **get\_id**(){

        return id;

    }

**string** **get\_name**(){

        return name;

    }

**string** **get\_email**(){

        return email;

    }

    long long **get\_hash**(){

        return hashed\_password;

    }

**vector**<**string**> **get\_permission**(){

        return permissions;

    }

**string** **get\_permission\_index**(int i){

        return permissions**[**i**]**;

    }

    void **authenticate**(**string** a){

        if(**generate\_hashes**(a)==this->hashed\_password){

            cout**<<**"Correct Password.\nAuthentication Provided"**<<endl**;

        }

        else{

            cout**<<**"Incorrect Password"**<<endl**;

        }

    }

    virtual void **Display**(){

        cout**<<**"User Id: "**<<**id**<<endl**;

        cout**<<**"User name: "**<<**name**<<endl**;

        cout**<<**"Email: "**<<**email**<<endl**;

        cout**<<**"Hashed Password: "**<<**hashed\_password**<<endl**;

        if(permissions.**size**()==0) cout**<<**"No Permissions have been given to this User"**<<endl**;

        else{

            cout**<<**"List of Permissions: "**<<endl**;

            for(int i=0;i<permissions.**size**();i++){

                cout**<<**"  "**<<**permissions**[**i**]<<endl**;

            }

        }

    }

    void **access\_lab**(){

        if( **authenticateAndPerformAction**(this,"Lab Access")==true) cout**<<**"Lab Access Granted"**<<endl**;

        else cout**<<**"Lab Access not granted"**<<endl**;

    }

    virtual void **give\_assignment**(){}

    virtual void **submit\_assignment**(){}

};

bool **authenticateAndPerformAction**(**User**\* user, **string** action){

    bool check=false;

**vector**<**string**> permission=user->**get\_permission**();

    for(int i=0;i<permission.**size**();i++){

        if(permission**[**i**]==**action){

            check=true;

            break;

        }

    }

    if(check==false) return false;

    else return true;

}

class **Student**:public **User**{

**vector**<bool> assignments;

    public:

**Student**(){};

**Student**(int d,**string** a, **string** b, **string** c):**User**(d,a,b,c){};

    void **Display**() override{

        cout**<<**"\nStudent Id: "**<<**this->**get\_id**()**<<endl**;

        cout**<<**"Student name: "**<<**this->**get\_name**()**<<endl**;

        cout**<<**"Email: "**<<**this->**get\_email**()**<<endl**;

        cout**<<**"Hashed Password: "**<<**this->**get\_hash**()**<<endl**;

        if(this->**get\_permission**().**size**()==0) cout**<<**"No Permissions have been given to this User"**<<endl**;

        else{

            cout**<<**"List of Permissions: "**<<endl**;

            for(int i=0;i<this->**get\_permission**().**size**();i++){

                cout**<<**"  "**<<**this->**get\_permission\_index**(i)**<<endl**;

            }

        }

        if(assignments.**size**()==0) cout**<<**"No Assignments Provided Yet"**<<endl**;

        else{

            cout**<<**"Assignments Submitted: "**<<endl**;

            for(int i=0;i<assignments.**size**();i++){

                cout**<<**"  Assignment "**<<**i+1**<<**": "**<<**assignments**[**i**]<<endl**;

            }

        }

    }

    void **give\_assignment**() override{

        assignments.**push\_back**(false);

    }

    void **submit\_assignment**(){

        int j;

        bool check=false;

        if(assignments.**size**()==0){

            cout**<<**"No Assignment have been Provided Yet"**<<endl**;

        }

        else{

            cout**<<**"Enter the Sequence Number of Assignment: ";

            cin**>>**j;

            for(int i=0;i<assignments.**size**();i++){

                if(j==i+1){

                    cout**<<**"Assignment Submitted"**<<endl**;

                    assignments**[**i**]=**true;

                    check=true;

                    break;

                }

            }

        }

        if(check==false) cout**<<**"No Assignment of this Index Exist"**<<endl**;

    }

};

class **TA**: public **Student**{

**vector**<**Student**> assigned\_student;

**vector**<**string**> projects;

    public:

**TA**(){}

**TA**(int d,**string** a, **string** b, **string** c):**Student**(d,a,b,c){}

    void **Display**() override{

        cout**<<**"\nTA Id: "**<<**this->**get\_id**()**<<endl**;

        cout**<<**"TA name: "**<<**this->**get\_name**()**<<endl**;

        cout**<<**"Email: "**<<**this->**get\_email**()**<<endl**;

        cout**<<**"Hashed Password: "**<<**this->**get\_hash**()**<<endl**;

        if(this->**get\_permission**().**size**()==0) cout**<<**"No Permissions have been given to this User"**<<endl**;

        else{

            cout**<<**"List of Permissions: "**<<endl**;

            for(int i=0;i<this->**get\_permission**().**size**();i++){

                cout**<<**"  "**<<**this->**get\_permission\_index**(i)**<<endl**;

            }

        }

        if(assigned\_student.**size**()==0) cout**<<**"No Students have been Assigned yet"**<<endl**;

        else{

            cout**<<**"Assigned Students: "**<<endl**;

            for(int i=0;i<assigned\_student.**size**();i++){

                cout**<<**assigned\_student**[**i**]**.**get\_name**()**<<endl**;

            }

        }

    }

    void **assign\_Student**(**Student**& s){

        bool check=false;

        if(this->**get\_permission**().**size**()>=10) cout**<<**"Limit Reached!\nNot more than 10 Students can be Assigned"**<<endl**;

        else{

            if(**authenticateAndPerformAction**(this,"Assigning")==true){

                assigned\_student.**push\_back**(s);

                cout**<<**"Student Added"**<<endl**;

            }

            else cout**<<**"You have not the permission to Assign the Student"**<<endl**;

        }

    }

    void **view\_projects**(){

        if(projects.**size**()==0) cout**<<**"Project List Empty";

        else{

            for(int i=0;i<projects.**size**();i++){

                cout**<<**"Project "**<<**i+1**<<**": "**<<**projects**[**i**]<<endl**;

            }

        }

    }

    void **new\_project**(**string** s){

        if(projects.**size**()>=2) cout**<<**"Limit Reached"**<<endl**;

        else{

            projects.**push\_back**(s);

            cout**<<**"Project Added"**<<endl**;

        }

    }

    int **get\_projects\_no**(){return projects.**size**();}

*// string get\_name(){return name;}*

};

class **Professor**: public **User**{

    public:

**Professor**(){}

**Professor**(int d, **string** a, **string** b, **string** c):**User**(d,a,b,c){}

    virtual void **Display**(){

        cout**<<**"\nProfessor Id: "**<<**this->**get\_id**()**<<endl**;

        cout**<<**"Professor name: "**<<**this->**get\_name**()**<<endl**;

        cout**<<**"Email: "**<<**this->**get\_email**()**<<endl**;

        cout**<<**"Hashed Password: "**<<**this->**get\_hash**()**<<endl**;

        if(this->**get\_permission**().**size**()==0) cout**<<**"No Permissions have been given to this User"**<<endl**;

        else{

            cout**<<**"List of Permissions: "**<<endl**;

            for(int i=0;i<this->**get\_permission**().**size**();i++){

                cout**<<**"  "**<<**this->**get\_permission\_index**(i)**<<endl**;

            }

        }

    }

    void **assign\_TA**(**TA**& a, **string** b){

        if(**authenticateAndPerformAction**(this,"Assigning TA")==true){

            if(a.**get\_projects\_no**()>=2) cout**<<**"TA Cannot be assigned to this Project, Max limit reached"**<<endl**;

            else{

                a.**new\_project**(b);

                cout**<<**"Professor "**<<**this->**get\_name**()**<<**" Assigned this "**<<**b**<<**" to "**<<**a.**get\_name**()**<<endl**;

            }

        }

        else{

            cout**<<**"Professor does not have permission to Assign Projects to TA"**<<endl**;

        }

    }

};

int **main**(){

**Student**\* users=new **Student**[4];

    users[0]**=Student**(101,"Ali","Ali@gmail.com","AliPass");

    users[1]**=Student**(102,"Ahmed","Ahmed@gmail.com","AhmedPass");

    users[2]**=Student**(103,"Akbar","Akbar@gmail.com","AkbarPass");

    users[3]**=Student**(104,"Jake","Jake@gmail.com","JakePass");

    for(int i=0;i<4;i++){

        for(int j=0;j<3;j++){

            users[i].**give\_assignment**();

        }

    }

*// users[0].submit\_assignment();*

    for(int i=0;i<4;i++){

        users[i].**Display**();

    }

**TA**\* ta1=new **TA**(201,"John","John@gmail.com","JohnPass");

    ta1->**add\_permissions**("Assigning");

    ta1->**assign\_Student**(users[0]);

    ta1->**new\_project**("Project 1 CS");

**Professor**\* p1=new **Professor**(301,"Dr Adil","Adil@gmail.com","AdilPass");

    p1->**add\_permissions**("Assigning TA");

    p1->**assign\_TA**(\*ta1,"Project 2 EE");

    p1->**assign\_TA**(\*ta1,"Project 3 EE");

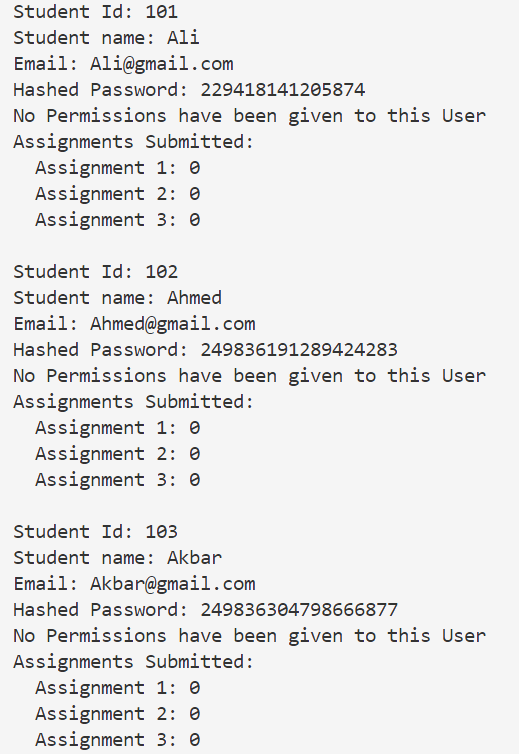
    p1->**Display**();

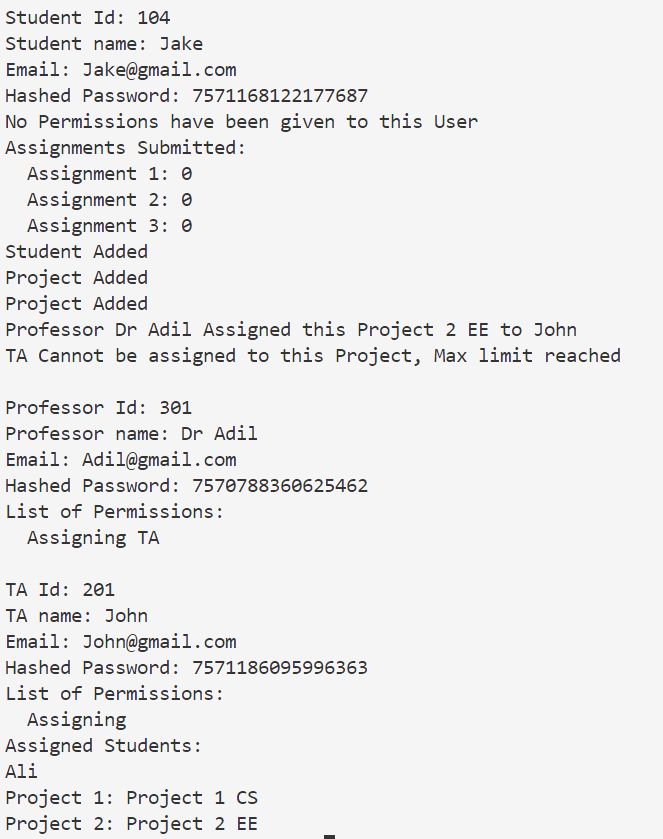
    ta1->**Display**();

    ta1->**view\_projects**();

}

**Output Screenshot**

****

****