**Object Oriented Programming**

**Assignment 01**

**Name:** Muhammad Shahbaz Hassan

**Roll Number:** 24K-0884

**Section:** BCS-2C

**Question 1  
  
Code:**

#include<iostream>

#include<string>

using namespace std;

class Mentor;

class Skill{

    int skillID;

    string name;

    string description;

    public:

    Skill(){

*this*->skillID=-1;

*this*->name**=**"NULL";

*this*->description**=**"NULL";

    }

    Skill(int id, string name, string description){

*this*->skillID=id;

*this*->name**=**name;

*this*->description**=**description;

    }

    void view\_Skill(){

        cout**<<**"\nSkill ID: "**<<***this*->skillID**<<**endl;

        cout**<<**"Skill Name: "**<<***this*->name**<<**endl;

        cout**<<**"Skill Description: "**<<***this*->description**<<**endl;

    }

    void updateSKillDescription(string newDescription){

*this*->description**=**newDescription;

    }

    int get\_skillID(){

        return *this*->skillID;

    }

};

class Sport{

    int sportID;

    string name;

    string description;

    Skill requiredSkill[100];

    public:

    Sport(){

        int i;

*this*->sportID=-1;

*this*->name**=**"NULL";

*this*->description**=**"NULL";

        for(i=0;i<100;i++){

            requiredSkill[i]**=**Skill();

        }

    }

    Sport(int id, string name, string description){

*this*->sportID=id;

*this*->name**=**name;

*this*->description**=**description;

    }

    void view\_Sport(){

        cout**<<**"\nSport ID: "**<<***this*->sportID**<<**endl;

        cout**<<**"Sport Name: "**<<***this*->name**<<**endl;

        cout**<<**"Sport Description: "**<<***this*->description**<<**endl;

    }

    void add\_Skill(Skill s){

        int i;

        for(i=0;i<100;i++){

            if(requiredSkill[i].get\_skillID()==-1){

                requiredSkill[i]**=**s;

                break;

            }

        }

    }

    void remove\_Skill(Skill s){

        int i,j;

        for(i=0;i<100;i++){

                if(requiredSkill[i].get\_skillID()==s.get\_skillID()){

                    requiredSkill[i]**=**Skill();

                    for(j=i;requiredSkill[j+1].get\_skillID()!=-1;j++){

                        requiredSkill[j]**=**requiredSkill[j+1];

                        cout**<<**"Hello"**<<**endl;

                        requiredSkill[j+1]**=**Skill();

                    }

                }

        }

    }

    void view\_RequiredSkill(){

        int i;

        bool check=false;

        for(i=0;i<100;i++){

            if(requiredSkill[i].get\_skillID()!=-1){

                requiredSkill[i].view\_Skill();

                check=true;

            }

            else{

                break;

            }

        }

        if(check==false){

            cout**<<**"\nSkills List Empty"**<<**endl;

        }

    }

    Skill get\_requiredSkill(int i){

        return requiredSkill[i];

    }

    int get\_sportID(){

        return *this*->sportID;

    }

};

class Student{

    int studentID;

    string name;

    int age;

    Sport\* sportsInterests[100];

    public:

    Mentor\* mentor;

    Student(){

*this*->studentID=-1;

*this*->name**=**"NULL";

*this*->age=-1;

*this*->mentor=nullptr;

        int i;

        for(i=0;i<100;i++){

            sportsInterests[i]=nullptr;

        }

    }

    Student(int id,string name, int age){

*this*->studentID=id;

*this*->name**=**name;

*this*->age=age;

*this*->mentor=nullptr;

    }

    Sport\* get\_sportsInterest(int i){

        return sportsInterests[i];

    }

    void view\_Student(){

        cout**<<**"\nStudent ID: "**<<***this*->studentID**<<**endl;

        cout**<<**"Student Name: "**<<***this*->name**<<**endl;

        cout**<<**"Student Age: "**<<***this*->age**<<**endl;

*this*->view\_mentorInfo();

    }

    void add\_SportsInterests(Sport s){

        int i;

        for(i=0;i<100;i++){

            if(sportsInterests[i]==nullptr){

                sportsInterests[i]=new Sport(s);

                break;

            }

        }

    }

    void remove\_SportsInterests(Sport s){

        int i,j;

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

            if(sportsInterests[i]->get\_sportID()==s.get\_sportID()){

                sportsInterests[i]=nullptr;

                for(j=i;sportsInterests[j+1]!=nullptr;j++){

                    sportsInterests[j]=sportsInterests[j+1];

                    sportsInterests[j+1]=nullptr;

                }

            }

        }

    }

    void view\_SportsInterests(){

        int i;

        bool check=false;

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

            sportsInterests[i]->view\_Sport();

            check=true;

        }

        if(check==false){

            cout**<<**"\nSports Interest List Empty"**<<**endl;

        }

    }

    int get\_studentID(){

        return *this*->studentID;

    }

    void registerForMentorship(Mentor &m);

    void view\_mentorInfo();

};

class Mentor{

    int mentorID;

    string name;

    int maxLearners;

    Sport\* sportsExpertise[100];

    Student\* assignedLearners[100];

    public:

    Mentor(){

*this*->mentorID=-1;

*this*->name**=**"NULL";

*this*->maxLearners=-1;

        int i;

        for(i=0;i<100;i++){

            sportsExpertise[i]=nullptr;

            assignedLearners[i]=nullptr;

        }

    }

    Mentor(int id, string name, int max){

*this*->mentorID=id;

*this*->name**=**name;

*this*->maxLearners=max;

    }

    int get\_mentorID(){

        return *this*->mentorID;

    }

    int get\_max(){

        return *this*->maxLearners;

    }

    Student\* get\_Learners(int i){

        return assignedLearners[i];

    }

    void provide\_guidance(){

        cout**<<**"\nGuidance Provided"**<<**endl;

    }

    void assignLearners(Student& s){

        int i;

        for(i=0;i<maxLearners;i++){

            if(assignedLearners[i]==nullptr){

                assignedLearners[i]=&s;

                s.mentor=*this*;

                break;

            }

        }

    }

    void removeStudent(Student s){

        int i,j;

        for(i=0;i<maxLearners;i++){

            if(assignedLearners[i]->get\_studentID()==s.get\_studentID()){

                assignedLearners[i]=nullptr;

                for(j=i;assignedLearners[j+1]!=nullptr;j++){

                    assignedLearners[j]=assignedLearners[j+1];

                    assignedLearners[j+1]=nullptr;

                    s.mentor=nullptr;

                }

            }

        }

    }

    void view\_assignedLearners(){

        int i;

        bool check=false;

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

            assignedLearners[i]->view\_Student();

            check=true;

        }

        if(check==false){

            cout**<<**"\nAssigned Learners List Empty"**<<**endl;

        }

    }

    void add\_sportsExpertise(Sport s){

        int i;

        for(i=0;i<100;i++){

            if(sportsExpertise[i]==nullptr){

                sportsExpertise[i]=new Sport(s);

                 break;

            }

        }

    }

    void remove\_sportsExpertise(Sport s){

        int i,j;

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

            if(sportsExpertise[i]->get\_sportID()==s.get\_sportID()){

                sportsExpertise[i]=nullptr;

                for(j=i;sportsExpertise[j+1]!=nullptr;j++){

                    sportsExpertise[j]=sportsExpertise[j+1];

                    sportsExpertise[j+1]=nullptr;

                }

            }

        }

    }

    void view\_sportsExpertise(){

        int i;

        bool check=false;

        for(i=0;i<100;i++){

            if(sportsExpertise[i]!=nullptr){

                sportsExpertise[i]->view\_Sport();

                check=true;

            }

            else{

                break;

            }

        }

        if(check==false){

            cout**<<**"\nSports Expertise List Empty"**<<**endl;

        }

    }

    Sport\* get\_sportsExpertise(int i){

        return sportsExpertise[i];

    }

    void view\_mentor(){

        cout**<<**"\nMentor ID: "**<<***this*->mentorID**<<**endl;

        cout**<<**"Mentor Name: "**<<***this*->name**<<**endl;

        cout**<<**"Max Learners: "**<<***this*->maxLearners**<<**endl;

    }

};

void Student::registerForMentorship(Mentor &m){

    int i,size=0;

    for(i=0;i<100;i++){

        if(m.get\_Learners(i)==nullptr){

            break;

        }

        size++;

    }

    if(size<m.get\_max()){

        mentor=&m;

        m.assignLearners(\**this*);

        cout**<<**"\nMentor added"**<<**endl;

    }

    else{

        cout**<<**"Limit reached | can't apply for mentorship"**<<**endl;

    }

}

void Student::view\_mentorInfo(){

    if(*this*->mentor!=nullptr){

*this*->mentor->view\_mentor();

    }

    else{

        cout**<<**"Mentor Not Chosen"**<<**endl;

    }

}

int main(){

    int a,b,c,i,j,k,x,z,l,q,w,e,r,t,y;

    string name1,name2,str1,str2;

    cout**<<**"Enter the number of Mentors the system will handle: ";

    cin**>>**a;

    cout**<<**"Enter the number of Students the system will handle: ";

    cin**>>**b;

    cout**<<**"Enter the number of Sports the system will handle: ";

    cin**>>**c;

    Sport sport[c];

    Student student[b];

    Mentor mentor[a];

    cout**<<**"\n--- Sport Info ---"**<<**endl;

    for(i=0;i<c;i++){

        cout**<<**"\nSport "**<<**i+1**<<**" Info"**<<**endl;

        bool check=true;

        do{

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

            cin**>>**x;

            if(x<=0){

                cout**<<**"ID Cannot be negative or zero"**<<**endl;

            }

            else{

            bool check2=false;

            for(j=0;j<c;j++){

                if(x==sport[j].get\_sportID()){

                    check2=true;

                    cout**<<**"ID Already Used for different sport"**<<**endl;

                    break;

                }

            }

            if(check2==false) break;

            }

        }while(1);

        cout**<<**"Enter the Name of the sport: ";

        cin**>>**name1;

        cin.ignore();

        cout**<<**"Enter the Description: ";

        getline(cin,name2);

        sport[i]**=** Sport(x,name1,name2);

        cout**<<**"How many Skills are required for this Sport: ";

        cin**>>**l;

        for(j=0;j<l;j++){

                do{

                    cout**<<**"\nEnter the Skill ID for Skill number "**<<**j+1**<<**": ";

                    cin**>>**x;

                    if(x<=0){

                        cout**<<**"ID cannot be negative or zero"**<<**endl;

                    }

                    else{

                        bool check2=false;

                        for(k=0;k<l;k++){

                            if(sport[i].get\_requiredSkill(k).get\_skillID()==x){

                                check2=true;

                                cout**<<**"ID Already in use for another skill of the same sport"**<<**endl;

                                break;

                            }

                        }

                        if(check2==false) break;

                    }

                }while(1);

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

                cin**>>**name1;

                cin.ignore();

                cout**<<**"Enter the Skill Description: ";

                getline(cin,name2);

                Skill s(x,name1,name2);

                sport[i].add\_Skill(s);

        }

    }

    int x1;

    cout**<<**"\n--- Mentor Info ---"**<<**endl;

    for(i=0;i<a;i++){

        cout**<<**"\nMentor "**<<**i+1**<<**" Info: "**<<**endl;

        do{

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

            cin**>>**x;

            if(x<=0){

                cout**<<**"ID cannot be zero or negative"**<<**endl;

            }

            else{

                bool check2=false;

                for(j=0;j<a;j++){

                    if(mentor[j].get\_mentorID()==x){

                        check2=true;

                        cout**<<**"ID Already in use"**<<**endl;

                        break;

                    }

                }

                if(check2==false) break;

            }

        }while(1);

        cout**<<**"Enter the mentor name: ";

        cin**>>**name1;

        cout**<<**"Enter the maximum Learners the mentor can handle: ";

        cin**>>**l;

        Mentor m(x,name1,l);

        mentor[i]**=**m;

        cout**<<**"How many sports the mentor is expert: ";

        cin**>>**x;

        if(x!=0){

            cout**<<**"Here is the list to choose from: "**<<**endl;

        for(j=0;j<c;j++){

            sport[j].view\_Sport();

            cout**<<**"\n";

            }

        }

        j=0;

        do{

            if(j<x){

                cout**<<**"\nEnter the sport ID for Sport number "**<<**j+1**<<**" : ";

            cin**>>**x1;

            bool check=false;

            for(k=0;k<c;k++){

                if(x1==sport[k].get\_sportID()){

                    mentor[i].add\_sportsExpertise(sport[k]);

                    check=true;

                    cout**<<**"\nSport is Added to the Mentor Expertise"**<<**endl;

                    j++;

                }

            }

            if(check==false){

                cout**<<**"\nWrong ID Entered"**<<**endl;

            }

            }

            else{

                break;

            }

        }while(1);

    }

    cout**<<**"\n--- Student Info ---"**<<**endl;

    for(i=0;i<b;i++){

        cout**<<**"\nStudent "**<<**i+1**<<**" Info: "**<<**endl;

        do{

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

        cin**>>**x;

        if(x<=0){

            cout**<<**"ID cannot be zero or negative"**<<**endl;

        }

        else{

            bool check=false;

            for(j=0;j<b;j++){

                if(student[j].get\_studentID()==x){

                    cout**<<**"ID Already in use"**<<**endl;

                    check=true;

                    break;

                }

            }

            if(check==false) break;

        }

        }while(1);

        cout**<<**"Enter the Student name: ";

        cin**>>**name1;

        cout**<<**"Enter the Student age: ";

        cin**>>**l;

        Student s(x,name1,l);

        student[i]**=**s;

        cout**<<**"How many sports the Student has interest: ";

        cin**>>**x;

        if(x!=0){

            cout**<<**"Here is the list to choose from: "**<<**endl;

        for(j=0;j<c;j++){

            sport[j].view\_Sport();

            cout**<<**"\n";

            }

        }

        j=0;

        do{

            if(j<x){

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

                cin**>>**x1;

                bool check=false;

                for(k=0;k<c;k++){

                    if(x1==sport[k].get\_sportID()){

                    student[i].add\_SportsInterests(sport[k]);

                    check=true;

                    cout**<<**"\nSport is Added to Student's Interest"**<<**endl;

                    j++;

                    }

                }

                if(check==false){

                    cout**<<**"\nWrong ID Entered"**<<**endl;

                }

            }

            else{

                break;

            }

        }while(1);

    }

    do{

        cout**<<**"\n--- MAIN Section ---\n1. Mentors\n2. Sport\n3. Student\n4. Exit\nEnter the corresponding number: ";

        cin**>>**q;

       if(q==1){

            do{

                cout**<<**"\nMAIN > MENTORS:\n1. Display Mentors Info\n2. Sports Expertise\n3. Assigned Learners\n4. Provide Guidance \n5. Exit\nEnter the Corresponding Number: ";

                cin**>>**w;

                if(w==1){

                    for(i=0;i<a;i++){

                        mentor[i].view\_mentor();

                    }

                }

                else if(w==2){

                    cout**<<**"\nEnter the Mentor ID: ";

                    cin**>>**e;

                    bool check=false;

                    for(i=0;i<a;i++){

                        if(mentor[i].get\_mentorID()==e){

                            check=true;

                            do{

                                cout**<<**"\nMAIN > MENTORS > SPORTS EXPERTISE:\n1. Display\n2. Add a sport\n3. Remove a Sport\n4. Exit\nEnter the Corresponding number: ";

                                cin**>>**r;

                                if(r==1){

                                   mentor[i].view\_sportsExpertise();

                                }

                                else if(r==2){

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

                                    cin**>>**t;

                                    bool check=false;

                                    for(j=0;j<c;j++){

                                        if(sport[j].get\_sportID()==t){

                                            mentor[i].add\_sportsExpertise(sport[j]);

                                            check=true;

                                            cout**<<**"\nSport Added to the Expertise"**<<**endl;

                                        }

                                    }

                                    if(check==false){

                                        cout**<<**"\nNo Sport of this ID exist"**<<**endl;

                                    }

                                }

                                else if(r==3){

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

                                    cin**>>**t;

                                    bool check=false;

                                    for(j=0;j<c;j++){

                                        if(sport[j].get\_sportID()==t){

                                            mentor[i].remove\_sportsExpertise(sport[j]);

                                            check=true;

                                            cout**<<**"\nSport Removed"**<<**endl;

                                        }

                                    }

                                    if(check==false){

                                        cout**<<**"\nNo Sport of this ID exist"**<<**endl;

                                    }

                                }

                                else if(r==4){

                                    break;

                                }

                            }while(1);

                        }

                    }

                    if(check==false){

                        cout**<<**"\nNo Mentor of this ID exist"**<<**endl;

                    }

                }

                else if(w==3){

                    cout**<<**"\nEnter the Mentor ID: ";

                    cin**>>**e;

                    bool check=false;

                    for(i=0;i<a;i++){

                        if(mentor[i].get\_mentorID()==e){

                            check=true;

                            do{

                                cout**<<**"\nMAIN > MENTORS > ASSIGNED LEARNERS:\n1. Display\n2. Add A learner\n3. Remove a learner\n4. Exit\nEnter the Corresponding number: ";

                                cin**>>**r;

                                if(r==1){

                                    mentor[i].view\_assignedLearners();

                                }

                                else if(r==2){

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

                                    cin**>>**t;

                                    int size=0;

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

                                        if(mentor[i].get\_Learners(j)!=nullptr){

                                            size++;

                                        }

                                    }

                                    if(size<mentor[i].get\_max()){

                                    bool check=false;

                                    for(j=0;j<b;j++){

                                        if(student[j].get\_studentID()==t){

                                            mentor[i].assignLearners(student[j]);

                                            check=true;

                                            cout**<<**"\nStudent Added"**<<**endl;

                                        }

                                    }

                                    if(check=false){

                                        cout**<<**"\nNo Student with this ID Exist"**<<**endl;

                                    }

                                    }

                                    else{

                                        cout**<<**"\nMaximum Capacity"**<<**endl;

                                    }

                                }

                                else if(r==3){

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

                                    cin**>>**t;

                                    for(j=0;j<b;j++){

                                        if(student[j].get\_studentID()==t){

                                            mentor[i].removeStudent(student[j]);

                                        }

                                    }

                                }

                                else if(r==4){

                                    break;

                                }

                            }while(1);

                        }

                    }

                    if(check==false){

                        cout**<<**"\nMentor with such ID does not exist"**<<**endl;

                    }

                }

                else if(w==4){

                    cout**<<**"\nEnter the Mentor ID: ";

                    cin**>>**e;

                    bool check=false;

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

                           if(mentor[i].get\_mentorID()==e){

                                mentor[i].provide\_guidance();

                                check=true;

                           }

                       }

                    if(check==false){

                        cout**<<**"\nNo Mentor of this ID Exist"**<<**endl;

                    }

                }

                else if(w==5){

                    break;

                }

            }while(1);

       }

       else if(q==2){

        do{

            cout**<<**"\nMAIN > SPORTS:\n1. Sports Info\n2. Required Skills\n3. Exit\nEnter the Corresponding number: ";

            cin**>>**w;

            if(w==1){

                for(i=0;i<c;i++){

                    sport[i].view\_Sport();

                }

            }

            else if(w==2){

                cout**<<**"\nEnter the Sport ID: ";

                cin**>>**e;

                bool check=false;

                for(i=0;i<c;i++){

                    if(e==sport[i].get\_sportID()){

                        check=true;

                        do{

                            cout**<<**"\nMAIN > SPORTS > REQUIRED SKILLS:\n1. Display\n2. Add a Skill\n3. Remove a Skill\n4. Exit\nEnter the Corresponding number: ";

                            cin**>>**r;

                            if(r==1){

                                sport[i].view\_RequiredSkill();

                            }

                            else if(r==2){

                                cout**<<**"Enter the Skill ID to add: ";

                                cin**>>**t;

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

                                cin**>>**str1;

                                cout**<<**"Enter the Skill Description: ";

                                cin**>>**str2;

                                Skill s(t,str1,str2);

                                sport[i].add\_Skill(s);

                                cout**<<**"\nSkill Added"**<<**endl;

                            }

                            else if(r==3){

                                cout**<<**"Enter the Skill ID to remove: ";

                                cin**>>**t;

                                bool check2=false;

                                for(j=0;sport[i].get\_requiredSkill(j).get\_skillID()!=-1;j++){

                                    if(sport[i].get\_requiredSkill(j).get\_skillID()==t){

                                        sport[i].remove\_Skill(sport[i].get\_requiredSkill(j));

                                        check2=true;

                                        cout**<<**"\nSkill Removed"**<<**endl;

                                    }

                                }

                                if(check2==false){

                                    cout**<<**"\nSkill with this ID does not Exist"**<<**endl;

                                }

                            }

                            else if(r==4){

                                break;

                            }

                        }while(1);

                    }

                }

                if(check==false){

                    cout**<<**"\nSport ID does not Exist"**<<**endl;

                }

            }

            else if(w==3){

                break;

            }

        }while(1);

       }

       else if(q==3){

        do{

            cout**<<**"\nMAIN > STUDENT: \n1. Display Student Info\n2. Sports Interest\n3. Register\n4. Exit\nEnter the Corresponding number: ";

            cin**>>**w;

            if(w==1){

                for(i=0;i<b;i++){

                    student[i].view\_Student();

                }

            }

            else if(w==2){

                cout**<<**"\nEnter the Student ID: ";

                cin**>>**e;

                bool check=false;

                for(i=0;student[i].get\_studentID()!=-1;i++){

                    if(student[i].get\_studentID()==e){

                        check=true;

                        do{

                            cout**<<**"\nMAIN > STUDENT > SPORTS INTEREST:\n1. Display\n2. Add Interest\n3. Remove Interest\n4. Exit\nEnter the Corresponding number: ";

                            cin**>>**r;

                            if(r==1){

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

                                    student[i].view\_SportsInterests();

                                }

                            }

                            else if(r==2){

                                cout**<<**"Enter the Sport ID to add: ";

                                cin**>>**t;

                                bool check=false;

                                for(j=0;j<c;j++){

                                    if(sport[j].get\_sportID()==t){

                                        student[i].add\_SportsInterests(sport[j]);

                                        check=true;

                                        cout**<<**"\nSport Added"**<<**endl;

                                    }

                                }

                                if(check==false){

                                    cout**<<**"\nNo Sport with this ID exist"**<<**endl;

                                }

                            }

                            else if(r==3){

                                cout**<<**"Enter the Sport ID to remove: ";

                                cin**>>**t;

                                bool check=false;

                                for(j=0;j<b;j++){

                                    if(student[i].get\_sportsInterest(j)->get\_sportID()==t){

                                        check=true;

                                        student[i].remove\_SportsInterests(\*student[i].get\_sportsInterest(j));

                                        cout**<<**"\nSport Removed"**<<**endl;

                                    }

                                }

                                if(check==false){

                                    cout**<<**"\nWrong ID Entered"**<<**endl;

                                }

                            }

                            else if(r==4){

                                break;

                            }

                        }while(1);

                    }

                }

                if(check=false){

                    cout**<<**"\nNo Student with this ID exist"**<<**endl;

                }

            }

            else if(w==3){

                cout**<<**"\nEnter the student ID: ";

                cin**>>**e;

                for(i=0;i<b;i++){

                    if(student[i].get\_studentID()==e){

                        cout**<<**"Enter the Mentor's ID: ";

                        cin**>>**r;

                        bool check=false;

                        for(j=0;j<a;j++){

                            if(mentor[j].get\_mentorID()==r){

                                student[i].registerForMentorship(mentor[j]);

                                check=true;

                            }

                        }

                        if(check==false){

                            cout**<<**"\nNo Mentor with this ID Exist"**<<**endl;

                        }

                        else{

                        cout**<<**"Mentor: ";

                        student[i].view\_mentorInfo();

                        }

                    }

                    else{

                        cout**<<**"\nNo Student with this ID Exist"**<<**endl;

                    }

                }

            }

            else if(w==4){

                break;

            }

        }while(1);

       }

       else if(q==4){

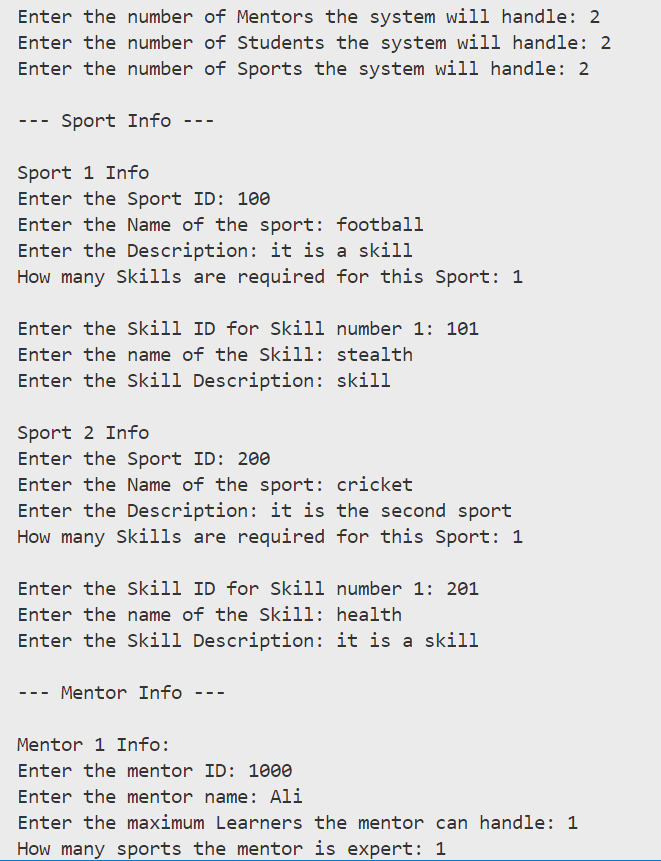
        break;

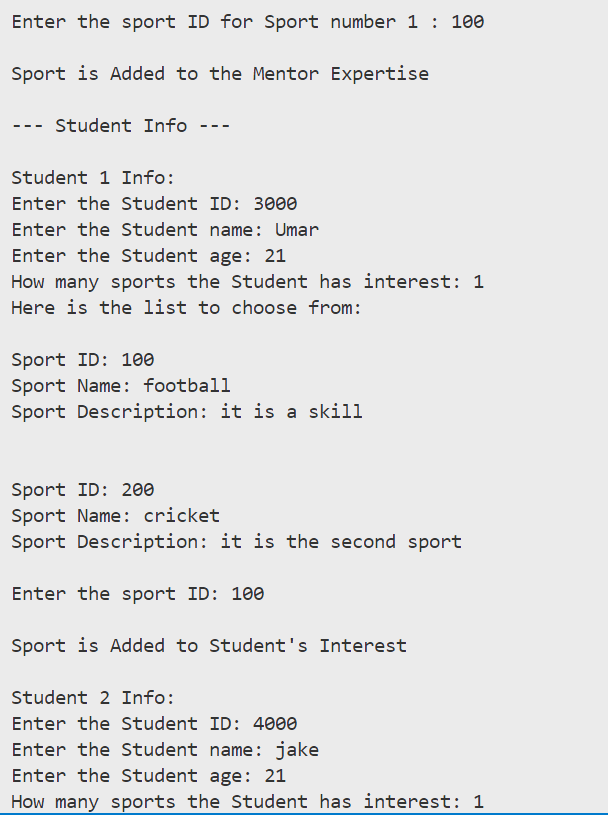
       }

    }while(1);

}

**Output:**

  
A screenshot of a computer

AI-generated content may be incorrect.  
  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.

**Question 2  
  
Code:**

#include<iostream>

using namespace std;

class Robot{

    string name;

    int hits=0;

    public:

    Robot(){

        this->name="NULL";

    }

    Robot(string name){

        this->name=name;

    }

    int get\_hits(){

        return this->hits;

    }

    void Hitball(int &ballX,int &ballY, string direction){

        if(direction=="u"){

            ballX--;

        }

        else if(direction=="d"){

            ballX++;

        }

        else if(direction=="l"){

            ballY--;

        }

        else if(direction=="r"){

            ballY++;

        }

       hits++;

    }

};

class Ball{

    int x;

    int y;

    Robot\* robot=nullptr;

    public:

    Ball(){

        this->x=0;

        this->y=0;

        this->robot=nullptr;

    }

    Ball(int x, int y){

        this->x=x;

        this->y=y;

    }

    int getX(){

        return this->x;

    }

    int getY(){

        return this->y;

    }

    void move(int dx, int dy){

        x=x+dx;

        y=y+dy;

    }

    void getPosition(){

        cout<<"("<<this->x<<","<<this->y<<")\n"<<endl;

    }

};

class Goal{

    int x;

    int y;

    public:

    Goal(){

        this->x=0;

        this->y=0;

    }

    Goal(int x, int y){

        this->x=x;

        this->y=y;

    }

    bool isGoalReached(int ballX, int ballY){

        if(ballX==this->x&&ballY==this->y){

            return true;

        }

        else{

            return false;

        }

    }

};

class Team{

    string teamName;

    Robot\* robot;

    public:

    Team(){

        this->teamName="NULL";

        this->robot=nullptr;

    }

    Team(string name, Robot\* robo){

        this->teamName=name;

        this->robot=robo;

    }

    string get\_name(){

        return this->teamName;

    }

    Robot\* get\_robot(){

        return this->robot;

    }

};

class Game{

    Team teamOne;

    Team teamTwo;

    Ball ball;

    Goal goal;

    int initiial\_ball\_x;

    int initial\_ball\_y;

    public:

    Game(Team one, Team two, Ball ball, Goal goal, int ball\_x, int ball\_y){

        this->teamOne=one;

        this->teamTwo=two;

        this->goal=goal;

        this->initiial\_ball\_x=ball\_x;

        this->initial\_ball\_y=ball\_y;

    }

    void play(Team\* team){

        string direc;

        int x,y;

        this->ball=Ball(initiial\_ball\_x, initial\_ball\_y);

        cout<<"\nTeam "<<team->get\_name()<<" turn"<<endl;

        do{

            cout<<"Enter the Direction: ";

            cin>>direc;

            x=ball.getX();

            y=ball.getY();

            team->get\_robot()->Hitball(x,y,direc);

            if(direc=="Up"){

                ball.move(-1,0);

            }

            else if(direc=="Down"){

                ball.move(1,0);

            }

            else if(direc=="Left"){

                ball.move(0,-1);

            }

            else if(direc=="Right"){

                ball.move(0,1);

            }

            else{

                cout<<"\nEnter only 4 directions : Up, Down, Left, Right"<<endl;

            }

            ball.getPosition();

            if(goal.isGoalReached(ball.getX(),ball.getY())==true){

                cout<<"Hits by Team "<<team->get\_name()<<": "<<  team->get\_robot()->get\_hits()<<endl;

                break;

            }

        }while(1);

    }

    void declareWinner(){

        cout<<"\n--- LeaderBoard ---"<<endl;

        cout<<"\nHits by team "<<teamOne.get\_name()<<" is: "<<teamOne.get\_robot()->get\_hits()<<endl;

        cout<<"\nHits by team "<<teamTwo.get\_name()<<" is: "<<teamTwo.get\_robot()->get\_hits()<<endl;

        if(teamOne.get\_robot()->get\_hits()>teamTwo.get\_robot()->get\_hits()){

            cout<<"\nTeam "<<teamTwo.get\_name()<<" Wins"<<endl;

        }

        else if(teamOne.get\_robot()->get\_hits()<teamTwo.get\_robot()->get\_hits()){

            cout<<"\nTeam "<<teamOne.get\_name()<<" Wins"<<endl;

        }

        else if(teamOne.get\_robot()->get\_hits()==teamTwo.get\_robot()->get\_hits()){

            cout<<"\nTie!"<<endl;

        }

    }

    void startGame(){

        cout<<"\n--- Football Simulation ---"<<endl;

        play(&teamOne);

        play(&teamTwo);

        declareWinner();

    }

};

int main(){

    string team1, team2, r1, r2;

    cout<<"\nEnter the name of Team 1: ";

    cin>>team1;

    cout<<"Enter the name of the Robot given of Team "<<team1<<": ";

    cin>>r1;

    cout<<"\nEnter the name of Team 2: ";

    cin>>team2;

    cout<<"Enter the name of the Robot given of Team "<<team2<<": ";

    cin>>r2;

    Robot\* robot1=new Robot(r1);

    Robot\* robot2=new Robot(r2);

    Team Team1(team1,robot1);

    Team Team2(team2,robot2);

    Ball ball(0,0);

    Goal goal(3,3);

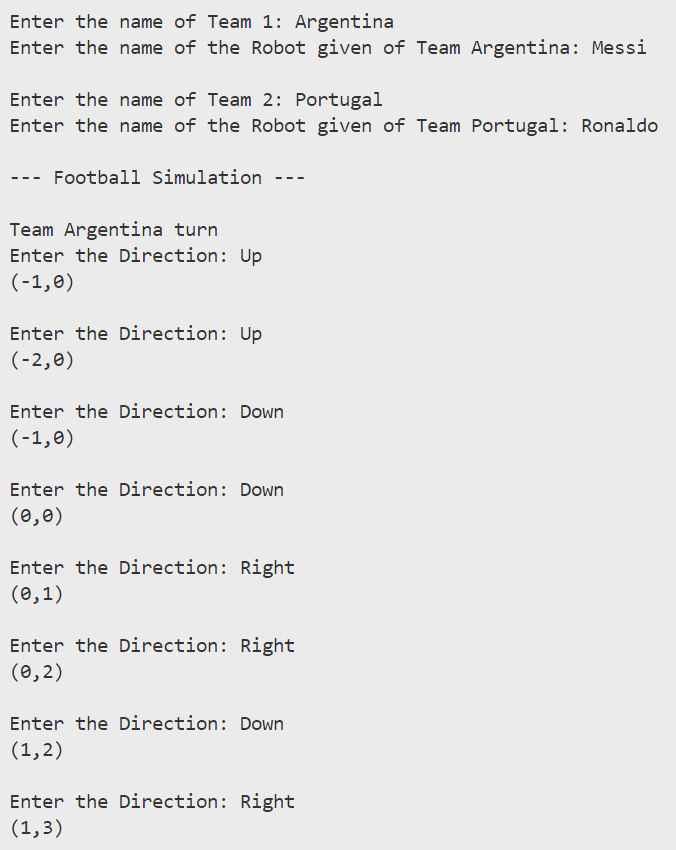
    Game game(Team1,Team2,ball,goal,ball.getX(),ball.getY());

    game.startGame();

    game.declareWinner();

}

**Output:**

  
A screenshot of a computer program

AI-generated content may be incorrect.  
A screenshot of a computer screen

AI-generated content may be incorrect.

**Question 3  
  
Code:**

#include<iostream>

using namespace std;

class Vehicle{

    int ID;

    string model;

    float rental\_Price;

    string eligibility;

    bool availibility;

    public:

    Vehicle(){

        this->ID=0;

        this->model="NULL";

        this->rental\_Price=0.0;

        this->eligibility="NULL";

        this->availibility=true;

    }

    Vehicle(string a, int b, string c, int d){

        this->model=a;

        this->rental\_Price=b;

        this->eligibility=c;

        this->ID=d;

        this->availibility=true;

    }

    void update\_model(string a){

        this->model=a;

    }

    void update\_rental(float a){

        this->rental\_Price=a;

    }

    void update\_eligiblity(string c){

        this->eligibility=c;

    }

    int get\_carID(){

        return this->ID;

    }

    string get\_model(){

        return this->model;

    }

    int get\_price(){

        return this->rental\_Price;

    }

    bool get\_availibility(){

        return this->availibility;

    }

    string get\_license(){

        return this->eligibility;

    }

    void update\_availibility(int a){

        if(a==0) this->availibility=false;

        else this->availibility=true;

    }

    void update\_ID(int ID){

        this->ID=ID;

    }

    void view\_Vehicle(){

        cout<<"\nVehicle ID: "<<this->get\_carID()<<endl;

        cout<<"Model: "<<this->get\_model()<<endl;

        cout<<"Rental Price per day: "<<this->get\_price()<<endl;

        cout<<"License Eligibility: "<<this->get\_license()<<endl;

        if(this->get\_availibility()==true){

            cout<<"Availibility Status: Available"<<endl;

        }

        else{

            cout<<"Availibility Status: Not Available"<<endl;

        }

    }

};

class Users{

    int user\_ID;

    int age;

    string license\_Type;

    int contact\_Info;

    Vehicle\* v;

    public:

    Users(){

        this->user\_ID=0;

        this->age=0;

        this->license\_Type="NULL";

        this->contact\_Info=0;

        this->v=nullptr;

    }

    Users(int a, string b, int c){

        this->age=a;

        this->license\_Type=b;

        this->contact\_Info=c;

    }

    void update\_ID(int ID){

        this->user\_ID=ID;

    }

    void update\_age(int a){

        this->age=a;

    }

    void update\_license(string b){

        this->license\_Type=b;

    }

    void update\_contact(int c){

        this->contact\_Info=c;

    }

    int get\_userID(){

        return this->user\_ID;

    }

    string get\_license(){

        return this->license\_Type;

    }

    int get\_age(){

        return this->age;

    }

    int get\_info(){

        return this->contact\_Info;

    }

    void rent\_Vehicle(Vehicle& b){

        if(b.get\_availibility()==1){

            bool available=false;

            if(this->get\_license()==b.get\_license()&&this->get\_license()=="Learner"){

                available=true;

            }

            else if(this->get\_license()=="Intermediate" && (b.get\_license()=="Intermediate" || b.get\_license()=="Learner")){

                available=true;

            }

            else if(this->get\_license()=="Full"){

                available=true;

            }

            else{

                cout<<"\nYour License is the problem"<<endl;

                available=false;

            }

            if(available==true){

                v=&b;

                cout<<"\nYour License type matched so you got the Car"<<endl;

                b.update\_availibility(false);

                v->update\_availibility(false);

            }

        }

        else{

            cout<<"\nVehicle is not available right now"<<endl;

        }

    }

    void view\_Rented(){

            if(this->v==nullptr){

                cout<<"No Vehicle Rented Yet"<<endl;

            }

            else{

                this->v->view\_Vehicle();

            }

    }

    bool view\_suitablecars(Vehicle\* b){

        if(b->get\_availibility()==true){

            if(this->get\_license()=="Learner"&&b->get\_license()=="Learner"){

                return true;

                }

                else if(this->get\_license()=="Intermediate"&&(b->get\_license()=="Learner"||b->get\_license()=="Intermediate")){

                    return true;

                }

                else if(this->get\_license()=="Full"){

                    return true;

                }

                else{

                    return false;

                }

            }

            else{

            return false;

        }

    }

    void view\_User(){

        cout<<"\nUser ID: "<<this->get\_userID()<<endl;

        cout<<"User Age: "<<this->get\_age()<<endl;

        cout<<"License Type: "<<this->get\_license()<<endl;

        cout<<"Contact Info: "<<this->get\_info()<<endl;

        if(this->v==nullptr){

            cout<<"No Vehicle Rented Yet"<<endl;

        }

        else{

            cout<<"Vehicle Rented: "<<endl;

            this->view\_Rented();

        }

    }

};

int main(){

    int s,v,i,n,j;

    cout<<"Enter the number of Max Students the system will manage: ";

    cin>>s;

    cout<<"Enter the number of Max Vehicles the System will manage: ";

    cin>>v;

    Users\* user=new Users[s];

    Vehicle\*\* vehicles=new Vehicle\*[v];

    for(i=0;i<s;i++){

        user[i]=Users();

    }

    for(i=0;i<v;i++){

        vehicles[i]=new Vehicle();

    }

    do{

        cout<<"\n1. Register a User\n2. Add a Vehicle\n3. Display Users\n4. Display all Vehicles\n5. Rent a Vehicle\n6. Update User Info\n7. Exit\nEnter the Corresponding Number: ";

        cin>>n;

        if(n==1){

            int age,info;

            string type;

            bool check=false;

            bool suitable=true;

            cout<<"\n--- User Registration ---"<<endl;

            for(i=0;i<s;i++){

                if(user[i].get\_userID()==0){

                    check=true;

                }

            }

            if(check==false){

                cout<<"\nSorry Max Space Reached No user can be registered"<<endl;

            }

            else{

                cout<<"--- User ID will be pre Assigned by the system ---"<<endl;

                cout<<"Enter the User age: ";

                cin>>age;

                do{

                    cout<<"Enter the User License Type: ";

                    cin>>type;

                    if(type=="Learner"||type=="Intermediate"||type=="Full"){

                        break;

                    }

                    else{

                        cout<<"\nLicense can only be Learner, Intermediate, Full"<<endl;

                    }

                }while(1);

                cout<<"Enter the User's Contact Info: ";

                cin>>info;

                for(i=0;i<s;i++){

                    if(user[i].get\_userID()==0){

                        user[i]=Users(age,type,info);

                        if(i==0){

                            user[i].update\_ID(1000);

                        }

                        else{

                            user[i].update\_ID(user[i-1].get\_userID()+1000);

                        }

                        break;

                    }

                }

                cout<<"Want to Rent a Car Now(Y or N)?[Note: You can also do it later]: ";

                cin>>type;

                if(type=="Y"){

                    for(j=0;j<v;j++){

                        if(vehicles[j]->get\_carID()!=0){

                            suitable=false;

                        }

                    }

                    if(suitable==false){

                        int k=0;

                       for(j=0;j<v;j++){

                           if(user[i].view\_suitablecars(vehicles[j])==1){

                            k++;

                        }

                       }

                       if(k==0){

                           cout<<"\nSorry No vehicle of your type available right now"<<endl;

                       }

                       else{

                           cout<<"\nThe Suitable Cars according to your license type are:  ";

                            for(j=0;j<v;j++){

                                if(user[i].view\_suitablecars(vehicles[j])==true){

                                    vehicles[j]->view\_Vehicle();

                                }

                            }

                       }

                    }

                    else{

                        cout<<"\nVehicles List Empty | No Data Entered Yet"<<endl;

                    }

                }

                cout<<"\nUser Registered. Here is the Registered Info"<<endl;

                user[i].view\_User();

            }

        }

        else if(n==2){

            int id,price;

            string model,eligible;

            bool check=false;

            cout<<"\n--- Vehicle Registration ---"<<endl;

            for(i=0;i<v;i++){

                if(vehicles[i]->get\_carID()==0){

                    check=true;

                }

            }

            if(check==false){

                cout<<"\nSorry Max Space Reached | No Vehicle can be added"<<endl;

            }

            else{

                cout<<"Enter the Vehicle model: ";

                cin>>model;

                cout<<"Enter the Rental Price of Vehicle per day: ";

                cin>>price;

                do{

                    cout<<"Enter the License Type: ";

                    cin>>eligible;

                    if(eligible=="Learner"||eligible=="Intermediate"||eligible=="Full"){

                        break;

                    }

                    else{

                        cout<<"\nLicense can only be Learner, Intermediate, Full"<<endl;

                    }

                }while(1);

                for(i=0;i<v;i++){

                    if(vehicles[i]->get\_carID()==0){

                        if(i==0){

                            vehicles[i]->update\_ID(1000);

                        }

                        else{

                            vehicles[i]->update\_ID(vehicles[i-1]->get\_carID()+1000);

                        }

                        \*vehicles[i]=Vehicle(model,price,eligible,vehicles[i]->get\_carID());

                        break;

                    }

                }

                cout<<"\nVehicle Registered Successfully\nHere is the Info of the registered Vehicle "<<endl;

                vehicles[i]->view\_Vehicle();

            }

        }

        else if(n==3){

            bool present=false;

            for(i=0;i<s;i++){

                if(user[i].get\_userID()!=0){

                    present=true;

                    break;

                }

            }

            if(present==true){

                cout<<"\n--- Users Data ---"<<endl;

                for(i=0;user[i].get\_userID()!=0&&i<s;i++){

                    user[i].view\_User();

                }

            }

            else{

                cout<<"\nList Empty"<<endl;

            }

        }

        else if(n==4){

            bool present=false;

            for(i=0;i<v;i++){

                if(vehicles[i]->get\_carID()!=0){

                    present=true;

                    break;

                }

            }

            if(present==true){

                cout<<"\n--- Vehicles Data ---"<<endl;

                for(i=0;i<v;i++){

                    if(vehicles[i]->get\_carID()!=0){

                        vehicles[i]->view\_Vehicle();

                    }

                }

            }

            else{

                cout<<"\nList Empty"<<endl;

            }

        }

        else if(n==5){

            int id,age,info;

            string type;

            bool exist=false;

            cout<<"\n--- Login User ---"<<endl;

            cout<<"Enter the User ID: ";

            cin>>id;

            for(i=0;i<s;i++){

                if(user[i].get\_userID()==id){

                    exist=true;

                    break;

                }

            }

            if(exist==false){

                cout<<"No User of this id exist"<<endl;

            }

            else{

                bool suitable=true;

                for(j=0;j<v;j++){

                    if(vehicles[j]->get\_carID()!=0){

                        suitable=false;

                    }

                }

                if(suitable==false){

                    int k=0;

                   for(j=0;j<v;j++){

                       if(user[i].view\_suitablecars(vehicles[j])==true){

                        k++;

                    }

                   }

                   if(k==0){

                       cout<<"\nSorry No vehicle of your type available right now"<<endl;

                   }

                   else{

                       cout<<"\nThe Suitable Cars according to your license type are:  ";

                        for(j=0;j<v;j++){

                            if(user[i].view\_suitablecars(vehicles[j])==1){

                                vehicles[j]->view\_Vehicle();

                            }

                        }

                        cout<<"\nEnter the ID of Vehicle you want to rent: ";

                        cin>>id;

                        bool present=false;

                        for(j=0;j<v;j++){

                            if(vehicles[j]->get\_carID()==id){

                                present=true;

                                break;

                            }

                        }

                        if(present==true){

                            user[i].rent\_Vehicle(\*vehicles[j]);

                        }

                        else{

                            cout<<"\nSorry this type of vehicle is not in our system"<<endl;

                        }

                   }

                }

                else{

                    cout<<"\nVehicles List Empty | No Data Entered Yet"<<endl;

                }

            }

        }

        else if(n==6){

            int i,j,a,b,c;

            string l;

            bool empty=false;

            for(i=0;i<s;i++){

                if(user[i].get\_userID()!=-1){

                    empty=true;

                    break;

                }

            }

            if(empty==false){

                cout<<"\nUser List Empty | Register Users First"<<endl;

            }

            else{

                cout<<"\nEnter the ID of the User whose data you want to update: ";

                cin>>a;

                bool check=false;

                for(i=0;i<s;i++){

                    if(user[i].get\_userID()==a){

                        check=true;

                        cout<<"\n --- User "<<user[i].get\_userID()<<" Info Update ---"<<endl;

                        do{

                            cout<<"\n1. User Age\n2. License Type\n3. Contact Info\n4. All\n5. Exit\nEnter the corresponding Number: ";

                            cin>>b;

                            if(b==1){

                                cout<<"\nEnter the User new age: ";

                                cin>>c;

                                user[i].update\_age(c);

                                cout<<"\nUpdated User Info: "<<endl;

                                user[i].view\_User();

                            }

                            else if(b==2){

                                cout<<"\nEnter the new License Type: ";

                                cin>>l;

                                user[i].update\_license(l);

                                cout<<"\nUpdated User Info: "<<endl;

                                user[i].view\_User();

                            }

                            else if(b==3){

                                cout<<"\nEnter the new Contact Info: ";

                                cin>>c;

                                user[i].update\_contact(c);

                                cout<<"\nUpdated User Info: "<<endl;

                                user[i].view\_User();

                            }

                            else if(b==4){

                                cout<<"\nEnter the User new age: ";

                                cin>>c;

                                user[i].update\_age(c);

                                cout<<"\nEnter the new License Type: ";

                                cin>>l;

                                user[i].update\_license(l);

                                cout<<"\nEnter the new Contact Info: ";

                                cin>>c;

                                user[i].update\_contact(c);

                                cout<<"\nUpdated User Info: "<<endl;

                                user[i].view\_User();

                            }

                            else if(b==5){

                                break;

                            }

                        }while(1);

                    }

                }

                if(check==false){

                    cout<<"\nWrong ID Entered"<<endl;

                }

            }

        }

        else if(n==7){

            break;

        }

        else{

            cout<<"\nEnter a Valid Option"<<endl;

        }

    }while(1);

};

**Output:  
A screenshot of a computer program

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer screen

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.**

**Question 4  
  
Code:**

#include<iostream>

using namespace std;

class Bus{

    int id;

    int n;

    string\* routes;

    public:

    Bus(){

        this->id=0;

        this->n=0;

        this->routes=nullptr;

    }

    Bus(int i){

        int j;

        this->n=i;

        this->id=0;

        routes=new string[i];

        for(j=0;j<i;j++){

            routes[j]="NULL";

        }

    }

    int get\_size(){

        return this->n;

    }

    void set\_id(int i){

        this->id=i;

    }

    int get\_id(){

       return this->id;

    }

    string get\_route(int i){

        return this->routes[i];

    }

    void update\_route(int i,string route){

        this->routes[i]=route;

    }

    void set\_routes(){

        int i;

        for(i=0;i<this->n;i++){

            if(i==n-1){

                cout<<"Last Stop will be: FAST University"<<endl;

                this->routes[i]="FAST University";

            }

            else{

                cout<<"Enter the "<<i+1<<" Stop: ";

                cin>>this->routes[i];

            }

        }

        cout<<"\nRemember: These routes are one way. For Dropoff, same buses will follow same route just opposite"<<endl;

    }

    void view\_info(){

        int i;

        cout<<"\nID: "<<this->id<<endl;

        cout<<"Routes: "<<endl;

        for(i=0;i<this->n;i++){

            if(i==n-1){

                cout<<this->routes[n-1]<<endl;

            }

            else{

                cout<<this->routes[i]<<" ---> ";

            }

        }

        cout<<"\nSame but opposite route will be followed in afternoon for dropping off"<<endl;

    }

};

class Student{

    int id;

    string name;

    int age;

    bool fee;

    string pickup;

    string dropoff;

    bool attendance;

    bool transportation\_card;

    Bus bus;

    public:

    Student(){

        this->age=0;

        this->id=0;

        this->fee=false;

        this->name="NULL";

        this->pickup="NULL";

        this->dropoff="NULL";

        this->attendance=false;

        this->transportation\_card=false;

        this->bus=Bus();

    }

    Student(string name, int age){

        this->name=name;

        this->age=age;

        this->id=0;

        this->fee=false;

        this->pickup=pickup;

        this->dropoff=dropoff;

        this->attendance=false;

        this->transportation\_card=false;

        this->bus=Bus();

    }

    void set\_bus(Bus b){

        this->bus=b;

    }

    void update\_id(int id){

        this->id=id;

    }

    void update\_pickup(string p){

        this->pickup=p;

    }

    void update\_dropoff(string d){

        this->dropoff=d;

    }

    void pay\_fee(){

        this->fee=true;

        this->transportation\_card=true;

    }

    bool get\_card(){

        return this->transportation\_card;

    }

    bool fee\_payed(){

    return this->fee;

    }

    int get\_id(){

        return this->id;

    }

    bool get\_attendance(){

        return this->attendance;

    }

    void mark\_attendance(bool card){

        if(card==true){

            this->attendance=true;

            cout<<"Attendance Marked"<<endl;

        }

        else{

            cout<<"Card not Active Yet"<<endl;

        }

    }

    void view\_student(){

        cout<<"\nID: "<<this->id<<endl;

        cout<<"Name: "<<this->name<<endl;

        cout<<"Age: "<<this->age<<endl;

        cout<<"Pickup: "<<this->pickup<<endl;

        cout<<"Drop off: "<<this->dropoff<<endl;

        if(this->fee==true){

            cout<<"Fee Paid: Yes"<<endl;

        }

        else{

            cout<<"Fee Paid: No"<<endl;

        }

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

            cout<<"Card Active: Yes"<<endl;

        }

        else{

            cout<<"Card Active: No"<<endl;

        }

        if(this->attendance==true){

            cout<<"Attendance: Yes"<<endl;

        }

        else{

            cout<<"Attendance: No"<<endl;

        }

        cout<<"Bus Info: "<<endl;

        if(this->bus.get\_id()==0){

            cout<<"Not Registered Yet"<<endl;

        }

        else{

            this->bus.view\_info();

        }

    }

};

int main(){

    int s,i,n,b;

    cout<<"Enter the number of Students the system will handle: ";

    cin>>s;

    cout<<"Enter the number of buses the university operates: ";

    cin>>b;

    Student\* student=new Student[s];

    Bus\* buses=new Bus[b];

    for(i=0;i<s;i++){

        student[i]=Student();

    }

    for(i=0;i<b;i++){

        buses[i]=Bus();

    }

    do{

        cout<<"\n1. Register Student\n2. Display Students\n3. Add a Bus\n4. View all buses\n5. Register for a Bus\n6. Pay Fee\n7. Mark Attendance\n8. Exit\nEnter the Corresponding number: ";

        cin>>n;

        if(n==1){

            int age;

            string name;

            bool check=false;

            for(i=0;i<s;i++){

                if(student[i].get\_id()==0){

                    check=true;

                    break;

                }

            }

            if(check==false){

                cout<<"\nSorry Maximum Space Reached | Cannot Register"<<endl;

            }

            else{

                cout<<"\n--- Student "<<i+1<<" Info ---"<<endl;

                cout<<"User ID is System Generated"<<endl;

                cout<<"Enter the name: ";

                cin>>name;

                cout<<"Enter the age: ";

                cin>>age;

                student[i]=Student(name,age);

                if(i==0){

                    student[i].update\_id(1000);

                }

                else{

                    student[i].update\_id((i+1)\*1000);

                }

                cout<<"Student Registered in the system| Remember the Student need to choose the routes separately"<<endl;

                cout<<"\nRegistered Student Info"<<endl;

                student[i].view\_student();

            }

        }

        else if(n==2){

            bool check=false;

            for(i=0;i<s;i++){

                if(student[i].get\_id()!=0){

                    check=true;

                }

            }

            if(check==false){

                cout<<"\nList Empty"<<endl;

            }

            else if(check==true){

                cout<<"\n--- Users Info ---"<<endl;

                for(i=0;i<s;i++){

                    if(student[i].get\_id()!=0){

                        student[i].view\_student();

                    }

                }

            }

        }

        else if(n==3){

        int n,i;

        bool check=false;

        for(i=0;i<b;i++){

            if(buses[i].get\_id()==0){

                    check=true;

                    break;

                }

            }

            if(check==false){

                cout<<"\nSorry Max Space Reached"<<endl;

            }

            else{

                cout<<"Enter the Number of routes the bus will have Including the Uni: ";

                cin>>n;

                buses[i]=Bus(n);

                if(i==0){

                    buses[i].set\_id(1);

                }

                else{

                    buses[i].set\_id(buses[i-1].get\_id()+1);

                }

                buses[i].set\_routes();

                cout<<"\nBus Info: "<<endl;

                buses[i].view\_info();

            }

        }

        else if(n==4){

            bool check=false;

            for(i=0;i<b;i++){

                if(buses[i].get\_id()!=0){

                    check=true;

                }

            }

            if(check==false){

                cout<<"\nList Empty"<<endl;

            }

            else{

                cout<<"\n--- Bus Info ---"<<endl;

                for(i=0;i<b;i++){

                    if(buses[i].get\_id()!=0){

                        buses[i].view\_info();

                    }

                }

            }

        }

        else if(n==5){

            int n,i,j,k,z;

            bool check=false,check2=false;

            string p1,p2;

            cout<<"Enter the Student ID: ";

            cin>>n;

            for(i=0;i<s;i++){

                if(student[i].get\_id()==n){

                    check=true;

                    if(student[i].get\_card()==false && student[i].fee\_payed()==false){

                        cout<<"\nYour Transportation is not active. First Pay the Semester Fee to active the card and then you will be eligible to register for the courses"<<endl;

                    }

                    else{

                        int arr[b];

                        cout<<"Enter your Pick up point: ";

                        cin>>p1;

                        cout<<"Enter the Dropoff point: ";

                        cin>>p2;

                        student[i].update\_pickup(p1);

                        student[i].update\_dropoff(p2);

                        for(j=0;j<b;j++){

                            arr[j]=-1;

                        }

                        for(j=0,z=0;j<b;j++){

                            bool pick=false;

                            bool drop=false;

                            for(k=0;k<buses[j].get\_size();k++){

                                if(buses[j].get\_route(k)==p1){

                                    pick=true;

                                }

                                if(buses[j].get\_route(k)==p2){

                                    drop=true;

                                }

                            }

                            if(pick==true && drop==true){

                                arr[z]=i;

                                z++;

                            }

                        }

                        if(arr[0]!=-1){

                            cout<<"\n--- Suitable Buses ---"<<endl;

                            for(j=0;arr[j]!=-1;j++){

                                buses[arr[j]].view\_info();

                            }

                        }

                        cout<<"\nEnter the ID of the bus: ";

                        cin>>n;

                        for(j=0;arr[j]!=-1;j++){

                            if(buses[arr[j]].get\_id()==n){

                                student[i].set\_bus(buses[arr[j]]);

                                check2=true;

                            }

                        }

                        if(check2==false){

                            cout<<"\nEnter the Correct ID"<<endl;

                        }

                    }

                }

            }

            if(check==false){

                cout<<"\nSorry this id doesn't exist"<<endl;

            }

        }

        else if(n==6){

            int n,i;

            bool check=false;

            cout<<"Enter the Student ID: ";

            cin>>n;

            for(i=0;i<s;i++){

                if(student[i].get\_id()==n){

                    check=true;

                    if(student[i].get\_card()==false){

                        student[i].pay\_fee();

                        cout<<"Fee Payed | Card Activated"<<endl;

                    }

                    else{

                        cout<<"Fee Already Paid"<<endl;

                    }

                }

            }

            if(check==false){

                cout<<"Student with such ID does not Exist"<<endl;

            }

        }

        else if(n==7){

            int n,i;

            bool check=false;

            cout<<"Enter the Student ID: ";

            cin>>n;

            for(i=0;i<s;i++){

                if(student[i].get\_id()==n){

                    check=true;

                   if(student[i].get\_attendance()==false){

                    student[i].mark\_attendance(student[i].get\_card());

                   }

                }

            }

            if(check==false){

                cout<<"Student with such ID does not Exist"<<endl;

            }

        }

        else if(n==8){

            break;

            }

    }while(1);

}

**Output:**

A screenshot of a computer program

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer program

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer screen

AI-generated content may be incorrect.  
A screenshot of a computer program

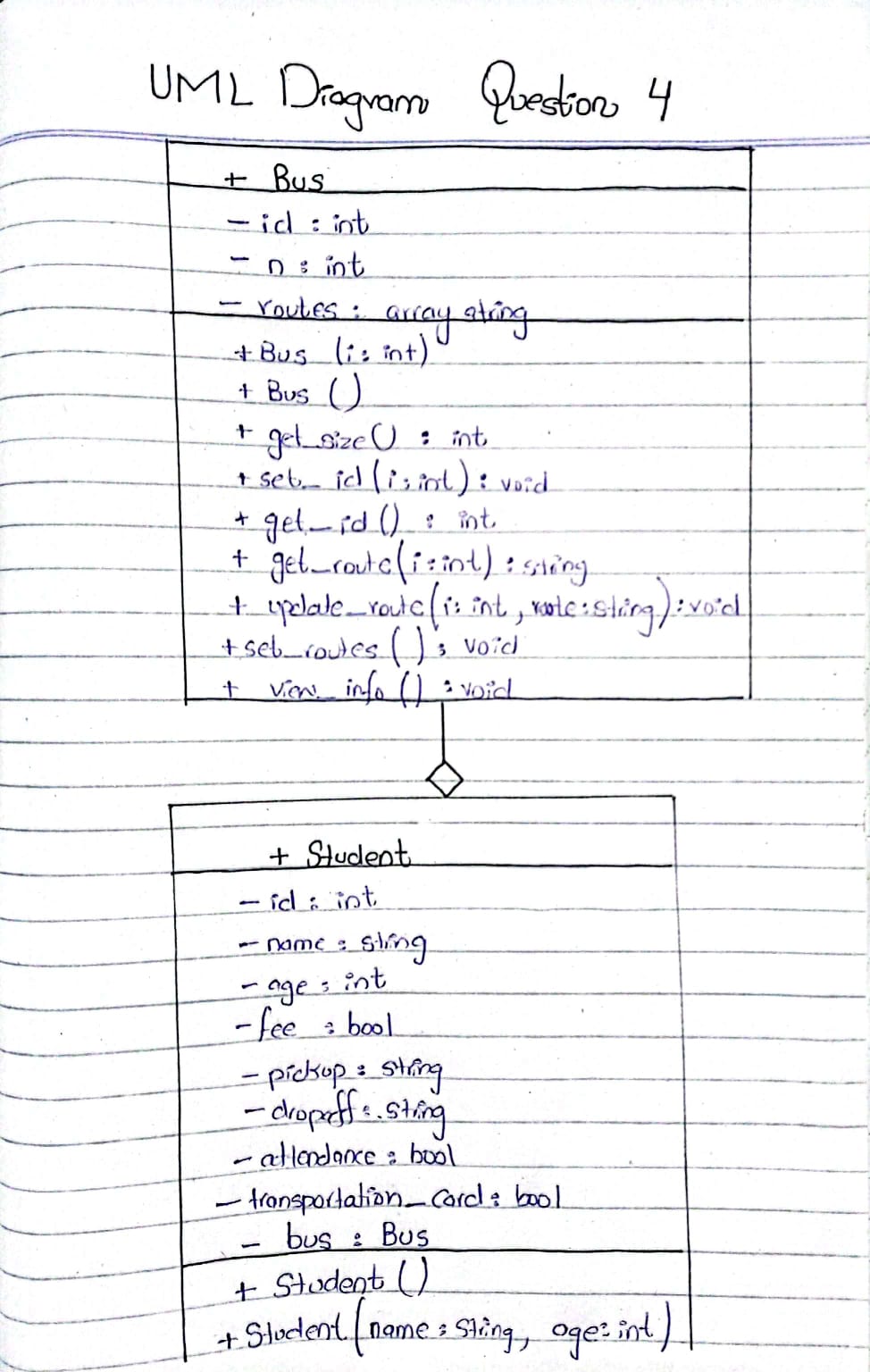
AI-generated content may be incorrect.  
A screenshot of a computer program

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer program

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.

**UML Diagram:  
  
A close-up of a list of software

AI-generated content may be incorrect.**