Problem Statement 01:

A Mess owner wants to develop a software for its mess members. Everyday mess member meal details will be entered in the software and after the month it will show the bill of the mess member. Mess member deposit at least 1000 tk at beginning of the month. As a student of CSE, write OOP code for the project. All the communications will be held by message.

Source Code:

#include<bits/stdc++.h>

using namespace std;

class Person {

public:

Person() {}

string name, date\_of\_birth;};

class Mess{

public:

string name;

Mess() {}; ///Constructor-1 /Default Constructor

Mess(string name){ ///Constructor-2

this->name = name;}

friend view\_PRINT();};

class Owner{

public:

class Member : public Person { ///Member class inherits Person class;

public:

void getName(string name, string date\_of\_birth){

this->name = name;

this->date\_of\_birth = date\_of\_birth;}

class Meal { ///Meal class;

public:

float deposit, total\_cost, meal\_rate;

int t\_meal;

static int counter,cnt;

string cmeal[10],date[10];

meal\_count(string datee,string mel){ ///function for meal counting

date[counter] = datee;

cmeal[counter] = mel;

if(cnt % 5 == 4)

counter++;

cnt++;}

get\_meal(int total\_meal, float deposit, float meal\_rate){ /// function for Meal information

total\_cost = total\_meal \* meal\_rate;

this->deposit = deposit;

t\_meal = total\_meal;

this->meal\_rate=meal\_rate;}};

view(void){ /// function for viewing Meal information

cout << "Name : " << name <<endl;

cout << "Date of Birth : " << date\_of\_birth <<endl;

cout << "---------------------------" <<endl;

cout << "Total meal --> " << ml.t\_meal << endl;

cout << "Meal Rate --> " << ml.meal\_rate << endl;

cout << "Total Cost --> " << ml.total\_cost << endl;

cout << "Total Deposit --> " << ml.deposit << endl;

cout << "Total Balance --> " << ml.deposit - ml.total\_cost << endl<<endl;

cout << "----Meal Information of " << name << "----\n"<<endl;

for(int j = 0; j<5; j++)

cout << ml.date[j] << " --> " << ml.cmeal[j] <<endl;

cout << endl<<endl;

cout << "---------------------------" <<endl;

cout << "---------------------------" <<endl<<endl<<endl;

} Meal ml;

};

Member members[7]; }; ///Has-A relationship

view\_PRINT(Mess mess){

cout << "\nMess Name : " << mess.name << endl;}

int Owner::Member::Meal :: counter; ///Defination of Static Data Member

int Owner::Member::Meal:: cnt; ///Defination of Static Data Member

int main(){

Mess mess(" AllaUddin\n");

Owner owner;

string name,date\_of\_birth,deposite,date,cmeal[10][10];

int counter=0,all\_total = 0;

int total[10]= {0};

string a,b,c,d,e;

ifstream file,file\_2;

file\_2.open("E:/cpp project/csv files/meal - Copy2.csv"); ///Open a csv file named meal

int test=5,xx=0;

while(test--){

getline(file\_2,date,',');

for(int s=0; s<5; s++){ ///Meal Count

if(s==4) getline(file\_2, cmeal[s][counter], '\n'); ///for person 3

else if(s==3) getline(file\_2, cmeal[s][counter], ',');

else if(s==2) getline(file\_2, cmeal[s][counter], ','); /// for person 1,2

else if(s==1) getline(file\_2, cmeal[s][counter], ','); /// for person 1,2

else if(s==0) getline(file\_2, cmeal[s][counter], ','); /// for person 1,2

if(counter == 0) a = cmeal[s][counter];

if(counter == 1) b = cmeal[s][counter];

if(counter == 2) c = cmeal[s][counter];

if(counter == 3) d = cmeal[s][counter];

if(counter == 4) e = cmeal[s][counter];

int Day1 = 0, Day2 = 0,Day3 = 0,Day4 = 0,Day5 = 0;

int num1 = 0,num2 =0,num3 = 0,num4 = 0,num5 = 0;

if(counter == 0){

for(int j=0; a[j]; j++){

if(a[j]>='0' and a[j]<='9'){

num1 \*= 10;

num1 += (a[j]-'0');}

else{

Day1 += num1;

num1 = 0;}}

Day1 += num1;}

else Day1 = 0;

if(counter == 1){

for(int k=0; b[k]; k++){

if(b[k]>='0' and b[k]<='9'){

num2 \*= 10;

num2 += (b[k]-'0');}

else{

Day2 += num2;

num2 = 0;}}

Day2 += num2;}

else Day2 = 0;

if(counter == 2 ){

for(int j=0; c[j]; j++){

if(c[j]>='0' and c[j]<='9'){

num3 \*= 10;

num3 += (c[j]-'0');}

else{

Day3 += num3;

num3 = 0;}}

Day3 += num3;}

else Day3 = 0;

if(counter == 3){

for(int j=0; d[j]; j++){

if(d[j]>='0' and d[j]<='9'){

num4 \*= 10;

num4 += (d[j]-'0');}

else{

Day4 += num4;

num4 = 0;}}

Day4 += num4;}

else Day4 = 0;

if(counter == 4){

for(int j=0; e[j]; j++){

if(e[j]>='0' and e[j]<='9'){

num5 \*= 10;

num5 += (e[j]-'0');}

else{

Day5 += num5;

num5 = 0;}}

Day5 += num5;}

else Day5 = 0;

if(s == 0 ) total[0] += Day1 +Day2 +Day3 +Day4 +Day5;

if(s == 1 ) total[1] += Day1 +Day2 +Day3 +Day4 +Day5;

if(s == 2 ) total[2] += Day1 +Day2 +Day3 +Day4 +Day5;

if(s == 3 ) total[3] += Day1 +Day2 +Day3 +Day4 +Day5;

if(s == 4 ) total[4] += Day1 +Day2 +Day3 +Day4 +Day5;

owner.members[s].ml.meal\_count(date,cmeal[s][counter]);}

counter++;}

file\_2.close(); ///file closed (meal)

ifstream bz;

string baz;

int bazar=0,m;

bz.open("E:/cpp project/csv files/bazar - Copy.csv"); ///Open a csv file named bazar

int tst = 5;

while(tst--){

getline(bz, baz,'\n');

stringstream gek(baz);

gek >> m;

bazar+=m;}

for(int i=0; i<5; i++)

all\_total+=total[i];

bz.close();

float meal\_rate = bazar / (float)all\_total;

file.open("E:/cpp project/csv files/name - Copy2.csv"); ///Open a csv file named "name"

int i =0;

test =5;

while(test--){

float final\_dep = 0.0;

getline(file,name,',');

getline(file,date\_of\_birth, ',');

getline(file,deposite, '\n');

stringstream geek(deposite); ///converts strings into integer

geek >> final\_dep;

owner.members[i].getName(name,date\_of\_birth);

owner.members[i].ml.get\_meal(total[i],final\_dep,meal\_rate);

i++;}

file.close(); ///file closed (name)

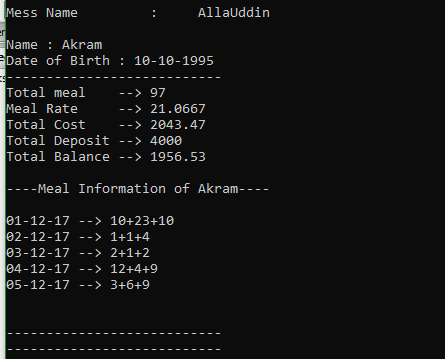
view\_PRINT(mess);

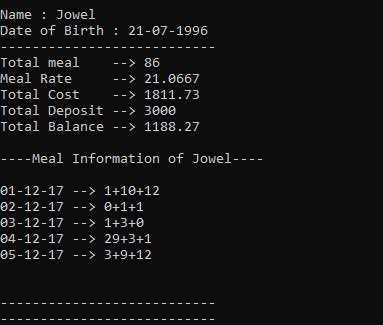
for(int i = 0; i<5; i++) ///Mess name print

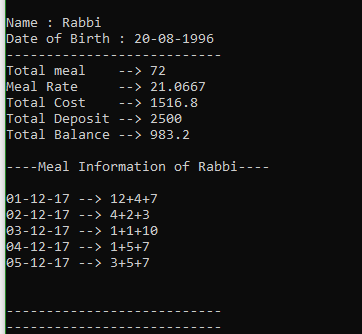
owner.members[i].view();

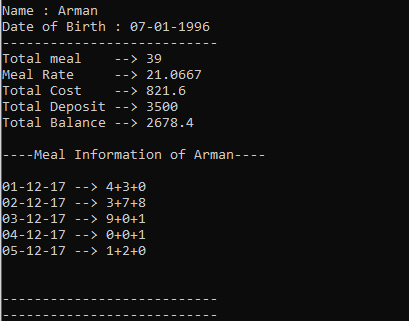
return 0;}

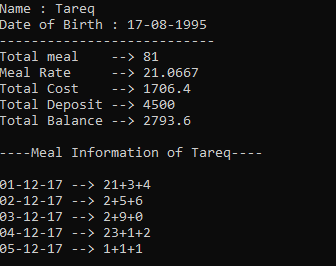
Output:



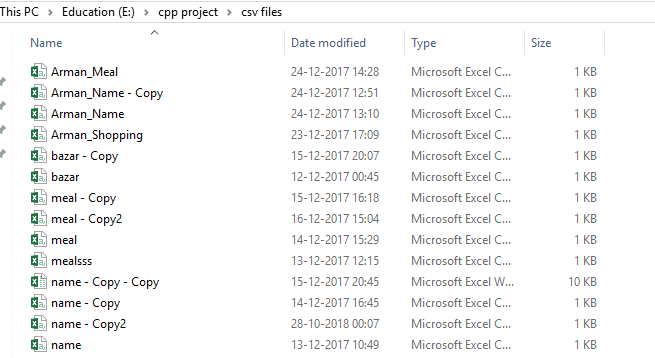








Csv files:



Problem Statement 02:

Write C++ code in object oriented approach for the students. Student can be rewarded from the department if he gets good GPA in a semester and solved atleast 100 ACM problems in the last year. Department will publish the top 5 students name in their honor board. As a student of CSE, write OOP code for the project.

Source Code:

#include<bits/stdc++.h>

using namespace std;

float arr[9];

int i,cnt=0;

class Person{

protected:

string name;

public:

Person() { }

Person(string x){

name=x;}

string acces\_person(){

return name;}};

class Contestant{

protected:

int ACM\_solve;};

class Student : public Contestant,public Person{

protected:

float GPA;

public:

Student() { }

Student(string Name,float y,int z):Person(Name){

GPA=y;

ACM\_solve=z;}

int checkgpa(float res){

if(GPA==res and ACM\_solve>=100)

return 1;

else return 2;}

float accessGPA(){

return GPA;}

void show\_student\_info(){

cout << setw(15) << acces\_person() << setw(19) << GPA << endl << endl;}};

class Teacher : public Person{

protected:

string designation,Username,Password;

public :

Teacher() { }

Teacher (string name,string w,string u, string p):Person(name){

designation=w;

Username=u;

Password=p;}

string access\_username(){ return Username;}

string access\_password(){ return Password;}

void show\_teacher\_info(){

cout << endl

<<"This List has been Prepared by" << endl

<< acces\_person() << endl << designation << endl;}

int Match\_info(string Username,string Password){

if(this->Username==Username and this->Password==Password)

return 1;

else return 2;}

void Bubblesort(){

float temp;

for(int i=0;i<8-1;i++)

for(int j=0;j<8-i-1;j++){

if(arr[j]>arr[j+1]){

temp=arr[j+1];

arr[j+1]=arr[j];

arr[j] = temp ;}}}};

class Department{

protected:

string Department\_name;

string saved\_username[3],saved\_password[3];

public:

Department() { Department\_name="Department of Computer Science and Engineering"; }

void Save\_username\_password(string j,string k,int l){

saved\_username[l]=j;

saved\_password[l]=k;}

int Access\_Request1(string Username){

for(i=0;i<3;i++){

if(saved\_username[i]==Username){

cnt=1;

return i;}}}

int Access\_Request2(string Password,int Index){

if(Password==saved\_password[Index])

return 1;}

void show\_dept\_name(){

cout << Department\_name << endl << endl;}};

class Date{

public :

string Month[14];

void Month\_initializer(){

Month[0]=" ",Month[1]="January",Month[2]="February",Month[3]="March",Month[4]="April",Month[5]="May",Month[6]="June",Month[7]="July",Month[8]="August",Month[9]="September",Month[10]="October",Month[11]="November",Month[12]="December";}

int year,month,date,x;

void Incertpresenttime(tm \*ltm){

year=ltm->tm\_year + 1900 ;

month=ltm->tm\_mon + 1;

date=ltm->tm\_mday ;}

void display(){

cout << "Date of publication " << endl;

cout << date << "th " << Month[month] << " ," << year << endl;}};

int main(){

int n,a,b,j,Index=-1;

Student \*performance[9]={

new Student("Islam",3.27 ,270 ),

new Student("Misty",3.31,180),

new Student("Romjan",3.42,345),

new Student("Raisul",3.76,434),

new Student("Karim",3.11,109),

new Student("Taijul",3.36,310),

new Student("Bappy",3.06,207),

new Student("Rohan",3.56,118)};

Department CSE;

Teacher \*assigned\_teacher[3]={

new Teacher("Kamal Hossain Chowdhury", "Chairman","Kamal\_Hossain","kamal0123" ),

new Teacher("Mahmudul Hasan Raju", "Assistant professor", "Mahmudul\_Hasan","raju0987"),

new Teacher("Faisal Bin Abdul Aziz", "Assistant professor","Faisal\_Aziz","faisal5678")

};

string temp\_username,temp\_password;

for(i=0;i<3;i++){

temp\_username=assigned\_teacher[i]->access\_username();

temp\_password=assigned\_teacher[i]->access\_password();

CSE.Save\_username\_password(temp\_username,temp\_password,i);}

string Username,Password;

cout << "Enter the Username" << endl;

cin >> Username;

Index=CSE.Access\_Request1(Username);

while(cnt!=1){

cout << "Invalid Username. Enter again" << endl;

cin >> Username;

Index=CSE.Access\_Request1(Username);}

cnt=0;

cout << "Enter the Password" << endl;

cin >> Password;

cnt=CSE.Access\_Request2(Password,Index);

while(cnt!=1){

cout << "wrong password. Enter again" << endl;

cin >> Password;

cnt=CSE.Access\_Request2(Password,Index);}

float temp;

int counter=0;

float maximum;

for(i=0;i<8;i++){

arr[i]=performance[i]->accessGPA();}

assigned\_teacher[Index]->Bubblesort();

cout << " TOP 5 students on the honor Board are" << endl << endl;

cout << setw(10) << "Student Name " << setw(10) << "GPA" << endl ;

cout << setw(10) << "\_\_\_\_\_\_\_\_\_\_\_\_\_" << setw(10) << "\_\_\_\_" << endl << endl ;

for(i=7;i>=0;i--){

maximum=arr[i];

for(j=0;j<8;j++){

if(performance[j]->checkgpa(maximum)==1){

counter++;

performance[j]->show\_student\_info();}

if(counter==5) break;}}

assigned\_teacher[Index]->show\_teacher\_info();

CSE.show\_dept\_name();

time\_t now = time(0);

tm \*ltm = localtime(&now);

Date A;

A.Incertpresenttime(ltm);

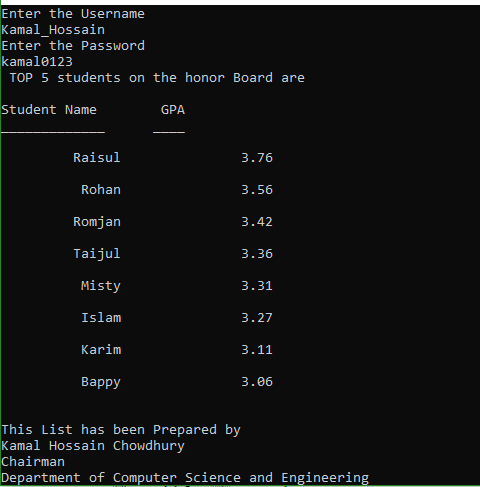
A.Month\_initializer();

A.display();

return 0;

}

Output:



Problem Statement 03:

CSE department wants to manage a Fast-food shop in its premise. Every student can make pre-order of his/her breakfast before 10 PM of the previous day. A sales person can manage the data and sells the preordered item to the students. If a student pre-ordered before but not take his/her breakfast and the sales person can inform it to the department. If he will not be illegible to pre-order the breakfast another time. As a student of CSE, write OOP code for the project. All the communications will be held by message.

Source Code:

#include<bits/stdc++.h>

using namespace std;

class Date{

public:

string date;

Date() {};

Date(string x){

date=x;}};

class Person{

public:

string name;

Date date;

Person() {};

Person(string xx){

name=xx;}

Person(string y,string yy){

name=y;

date.date=yy;};};

class Department{

public:

string dpt\_name;

Department() {};

Department(string name){

dpt\_name=name;}};

class Student : public Person{

public:

string id;

Department department;

Student() {};

Student(string name,string date,Department dpt\_name,string id\_no) : Person(name,date){

id=id\_no;

department=dpt\_name;}

view\_student\_info(){

cout<<"Student Name: "<<name<<endl;

cout<<"Department: "<<department.dpt\_name<<endl;

cout<<"Student ID: "<<id<<endl;

cout<<"Students Birthday: "<<this->date.date<<endl;}

friend orderfood();

friend blacklist();};

class Sellsman : public Person{

public:

int seller\_no;

Sellsman() {};

Sellsman(string name,string date,int seller\_no): Person(name,date){

this->seller\_no=seller\_no;};

view\_sellsman(){

cout<<"Sells person Name: "<<name<<endl;

cout<<"Birthday: "<<this->date.date<<endl;}

friend orderfood();};

class Login{

string user\_name;

string password;

public:

Login() {};

Login(string zz,string pp){

user\_name=zz;

password=pp;}

friend blacklist();};

class Faculty : public Person{

public:

string designation;

Department department;

Login login;

Faculty() {};

Faculty(string d,string name,Department dd,Login \*login2):Person(name){

designation=d;

department=dd;

login=\*login2;}

view\_faculty(){

cout<<"Name: "<<name<<endl;

cout<<"Department: "<<department.dpt\_name<<endl;

cout<<"Designation: "<<designation<<endl;}

Blacklist(int order\_data[],string deliver\_data[],Student \*student[],string blacklist[]){

cout<<"Blacklisted names are: \n";

for(int n=0; n<4; n++){

if(order\_data[n]==1)

if(deliver\_data[n]=="Didn't"){

blacklist[n]=student[n]->name;

cout<<"Name: "<<student[n]->name<<"\nID: "<<student[n]->id<<"\nYou have been blacklisted, Contact with Department"<<endl;}}};};

int view\_system\_time(){

time\_t k = time(0);

struct tm \*t = localtime(&k);

cout<< t->tm\_hour << ":" << t->tm\_min << endl;

return t->tm\_hour;}

int orderfood(Student \*student){

int tm=view\_system\_time();

if(tm>22){

cout<<"Sorry, You cannot Place the pre-order.\n";

return -1;}

else{

cout<<"Name: "<<student->name<<"\nID: "<<student->id<<"\n Your Order Has been placed.\n";

return 1;}}

int main(){

int number\_of\_students=100;

int temp=number\_of\_students;

Department department("CSE");

Student \*students[number\_of\_students]={

new Student("Afsana","9-9-1998",department,"11708001"),

new Student("Hira","2-09-1997",department,"11708002"),

new Student("Johan","22-11-1990",department,"11708003"),

new Student("Taifu","2-7-1992",department,"11708005"),

new Student("Utpol","2-7-1994",department,"11708006"),

new Student("Resel","2-7-1989",department,"11708007"),

new Student("Tofail","2-7-1996",department,"11708008")};

Sellsman sells\_person("Ratul","1-2-1975",1);

sells\_person.view\_sellsman();

Login \*login\_data[2]={

new Login("Jeshan","11708010"),

new Login("Karim","11708011")};

Faculty \*faculty\_member[2]={

new Faculty("Professor","Dipu Ahmed",department,login\_data[0]),

new Faculty("Lecturer","Dipa Bonik",department,login\_data[1])};

for(int i=0; i<3; i++)

students[i]->view\_student\_info();

for(int i=0; i<2; i++)

faculty\_member[i]->view\_faculty();

int order\_data[number\_of\_students],g=0;

for(int i=0; i<number\_of\_students; i++){

g=orderfood(students[i]);

order\_data[i]=g;}

string delivery\_info[number\_of\_students]= {"ordered","Didn't","ordered","Didn't"};

string blacklisted\_members[number\_of\_students];

faculty\_member[0]->Blacklist(order\_data,delivery\_info,students,blacklisted\_members);

ofstream students\_file;

students\_file.open("Students.csv");

students\_file<<"Students are: \n";

students\_file<<"Roll,Name,Department\n";

for(int i=0; i<number\_of\_students; i++){

students\_file<<students[i]->id<<","<<students[i]->name<<","<<students[i]->department.dpt\_name<<endl;}

ofstream blacklist\_file;

blacklist\_file.open("Blacklist.csv");

int counter=1;

blacklist\_file<<"Blacklisted names are:\n";

blacklist\_file<<"Roll,Name,Department\n";

string blacklisted\_id[number\_of\_students];

for(int i=0; i<number\_of\_students; i++){

if(blacklisted\_members[i]!="\0"){

blacklist\_file<<students[i]->id<<","<<blacklisted\_members[i]<<","<<students[i]->department.dpt\_name<<endl;

counter++;

blacklisted\_id[i]=students[i]->id;}}

blacklist\_file.close();

cout<<"Blacklisted person included in file.\n";

cout<<"New to order service?\nPress 1 to register or 0 to exit\n"<<endl;

int new\_order;

cin>>new\_order;

while(new\_order!=0){

string name,department\_name,birthday,roll;

cout<<"Enter Name: ";

cin>>name;

cout<<"\nEnter ID:";

cin>>roll;

cout<<"\nEnter Department: ";

cin>>department\_name;

cout<<"\nEnter birthday:";

cin>>birthday;

students[number\_of\_students ]= {new Student(name,birthday,department\_name,roll)};

number\_of\_students+=1;

int order\_data1;

order\_data1=orderfood(students[number\_of\_students-1]);

cout<<"New to order service?\nPress 1 to register or 0 to exit"<<endl;

cin>>new\_order;

if(new\_order!=1)

break;}

string delivery\_data2[number\_of\_students]= {"ordered","Didn't","ordered","Didn't","ordered"};

for(int i=temp; i<number\_of\_students; i++){

students\_file<<students[i]->id<<","<<students[i]->name<<","<<students[i]->department.dpt\_name<<endl;}

for(int i=0; i<number\_of\_students; i++){

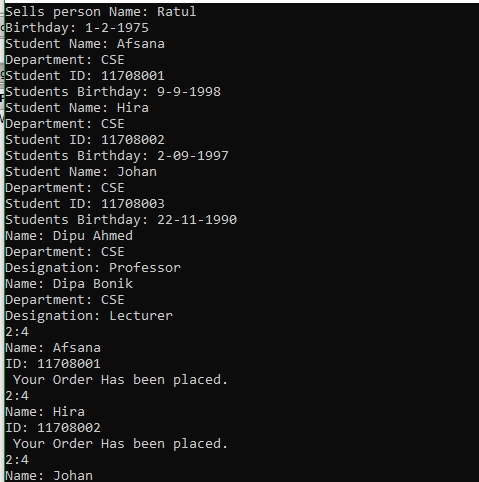
if(blacklisted\_id[i]!="\0" && students[i]->id==blacklisted\_id[i] )

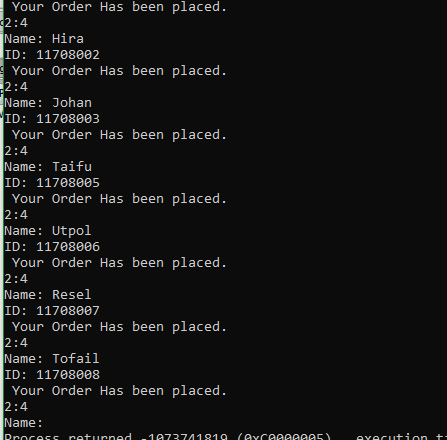
cout<<students[i]->name<<"\nYou cannot order,You have been blacklisted.\nContact with Department"<<endl;

else

orderfood(students[i]);}}

Output:





Problem Statement 04:

CSE department wants to select Programming Coach for its students. Any student of the department can be a coach. He need to have high profile at least three ACM regional contest participation and number of problem solutions of ACM need to more 300. Students need to apply in the department, if anyone’s performance is below the requirement he will discard automatically. As a student of CSE, write OOP code for the project. All the communications will be held by message.

Source Code:

#include<bits/stdc++.h>

using namespace std;

class Date{

public:

int day,month,year;

Date() {};

Date(int d,int m,int y){

day=d;

month=m;

year=y;}};

class Person{

public:

string name;

Date date;

Person() {};

Person(string x){

name=x;}

Person(string n,Date d1){

name=n;

date=d1;}};

class Department{

public:

string dept\_name;

Department() {};

Department(string n){

dept\_name=n;}};

class Student : public Person{

public:

int id,ACM,prob\_solve;

Department department;

Student() {};

Student(string name,Date \*date,Department \*department,int y,int a,int s) : Person(name,\*date){

id=y;

this->department=\*department;

ACM=a;

prob\_solve=s;}

View\_Student\_Info(){

cout<<"Student name: "<<name<<endl;

cout<<"Department: "<<department.dept\_name<<endl;

cout<<"Students Birthday: "<<this->date.day<<"-"<<this->date.month<<"-"<<this->date.year<<endl;}};

class Faculty : public Person{

public:

Student student;

string designation;

Department department;

Faculty() {};

Faculty(string d,string n,Department \*department):Person(d){

designation=n;

this->department=\*department;}

View\_Faculty(){

cout<<"Name: "<<name<<endl;

cout<<"Department: "<<department.dept\_name<<endl;

cout<<"Designation: "<<designation<<endl;}

int select\_coach(Student \*student){

if(student->ACM>=3&&student->prob\_solve>300)

return 1;

else

return 0;}};

int main(){

Department \*department[5]={

new Department("CSE"),

new Department("CSE"),

new Department("CSE"),

new Department("CSE"),

new Department("CSE"),};

Date \*date[4]={

new Date(20,8,1996),

new Date(15,05,1995),

new Date(13,11,1993),

new Date(18,02,1983)};

Student \*student[3]={

new Student("Rabbi",date[0],department[0],110011,4,600),

new Student("Saiful",date[1],department[1],110021,4,900),

new Student("Tuhin",date[2],department[2],110034,2,800)};

Faculty \*faculty[2]={

new Faculty("Kamal Hossain Chowdhury","Assistant Professor",department[3]),

new Faculty("Mahmudul Hasan","Assistant Professor",department[4])};

cout<<"THE DETAILS OF FACULTY MEMBER: "<<endl<<endl;

for(int i=0; i<2; i++){

faculty[i]->View\_Faculty();

cout<<endl;}

cout<<endl<<endl;

cout<<"THE DETAILS OF STUDENTS: "<<endl<<endl;

for(int i=0; i<3; i++){

student[i]->View\_Student\_Info();

cout<<endl;}

int x;

cout<<endl<<endl;

for(int i=0; i<3; i++){

x=faculty[i]->select\_coach(student[i]);

if(x==1){

cout<<"THE STUDENT CAN BE SELECTED AS COACH.."<<endl;

cout<<"THE DETAILS OF THE STUDENT IS: "<<endl;

student[i]->View\_Student\_Info();

cout<<endl<<endl;}

else{

cout<<"THE STUDENT IS NOT SELECTED AS COACH.."<<endl;

cout<<"THE DETAILS OF THE STUDENT IS: "<<endl;

student[i]->View\_Student\_Info();

cout<<endl<<endl;}}

return 0;}

Output:

