Name :: MUHAMMAD\_USAMA

Roll no ::17f\_8195.

Section :: B

Assignment : 4

Task 1:

#include<iostream>

#include<time.h>

#include<string>

using namespace std;

class tic\_tac\_toe

{

private:

char arry[3][3];

int check[3][3];

int count;

int m;

int f;

int win;

int wc;

public:

tic\_tac\_toe()

{

count=0;

f=1;

wc=0;

}

void assing()

{

arry[0][0]='1';

arry[0][1]='2';

arry[0][2]='3';

arry[1][0]='4';

arry[1][1]='5';

arry[1][2]='6';

arry[2][0]='7';

arry[2][1]='8';

arry[2][2]='9';

}

void reprint()

{

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

{

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

{

cout<<arry[i][j]<<" | ";

}

cout<<endl;

}

if(wc==1)

{

system("pause");

return ;

}

}

void user()

{

f=1;

while(f==1)

{

cout<<"please enter your move : ";

cin>>m;

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

{

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

{

count++;

if(count==m)

{

if(arry[i][j]!='U'&&arry[i][j]!='M')

{

if(arry[i][j]=='U')

cout<<"your success "<<endl;

arry[i][j]='U';

cout<<endl;

f=0;

break;

}

else

{

break;

}

}

}

}

if(f==1)

{

cout<<"please enter valid no"<<endl;

count=0;

}

count=0;

}

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

{

win=0;

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

{

if(arry[i][j]=='U')

{

win++;

}

}

if(win==3)

{

cout<<" user won game "<<endl ;

wc=1;

break;

}

}

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

{

win=0;

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

{

if(arry[i][j]=='U')

{

win++;

}

}

if(win==3)

{

cout<<" user won game "<<endl;

wc=1;

break;

}

}

if(arry[1][1]=='U'&&arry[2][2]=='U'&&arry[0][0]=='U')

{

cout<<"game won by user"<<endl;

wc=1;

}

if(arry[0][2]=='U'&&arry[1][1]=='U'&&arry[2][0]=='U')

{

cout<<"game won by user"<<endl;

wc=1;

}

}

void machine()

{

f=1;

while(f==1)

{

srand(time(0));

m=rand()%9+1;

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

{

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

{

count++;

if(count==m)

{

if(arry[i][j]!='M'&&arry[i][j]!='U')

{

cout<<"computer entered his move : "<<m<<endl;

f=0;

arry[i][j]='M';

cout<<endl;

break;

}

else

{

break;

}

}

}

}

count=0;

}

cout<<endl;

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

{

win=0;

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

{

if(arry[i][j]=='M')

{

win++;

}

}

if(win==3)

{

cout<<"computer won game "<<endl;

wc=1;

break;

}

}

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

{

win=0;

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

{

if(arry[i][j]=='M')

{

win++;

}

}

if(win==3)

{

cout<<"computer won game"<<endl;

wc=0;

break;

}

}

if(arry[1][1]=='M'&&arry[2][2]=='M'&&arry[0][0]=='M')

{

cout<<"computer won game"<<endl;

wc=1;

}

if(arry[0][2]=='M'&&arry[1][1]=='M'&&arry[2][0]=='M')

{

cout<<"computer won game"<<endl;

wc=1;

}

}

void mu()

{

t.reprint();

machine();

t.reprint();

user();

t.reprint();

machine();

t.reprint();

user();

t.reprint();

machine();

t.reprint();

user();

t.reprint();

machine();

t.reprint();

user();

t.reprint();

machine();

t.reprint();

}

void um()

{

t.reprint();

user();

t.reprint();

machine();

t.reprint();

user();

t.reprint();

machine();

t.reprint();

user();

t.reprint();

machine();

t.reprint();

user();

t.reprint();

machine();

t.reprint();

user();

t.reprint();

}

}t;

int main()

{

char choice;

t.assing();

do

{

cout<<"u : for first user take\nc : for first computer take "<<endl;

cin>>choice;

if(choice=='u'||choice=='U')

{

t.um();

}

else if(choice=='c'||choice=='C')

{

t.mu();

}

else

{

cout<<"you press invalid key"<<endl;

cout<<"a : for again choice exit for any"<<endl;

}

}

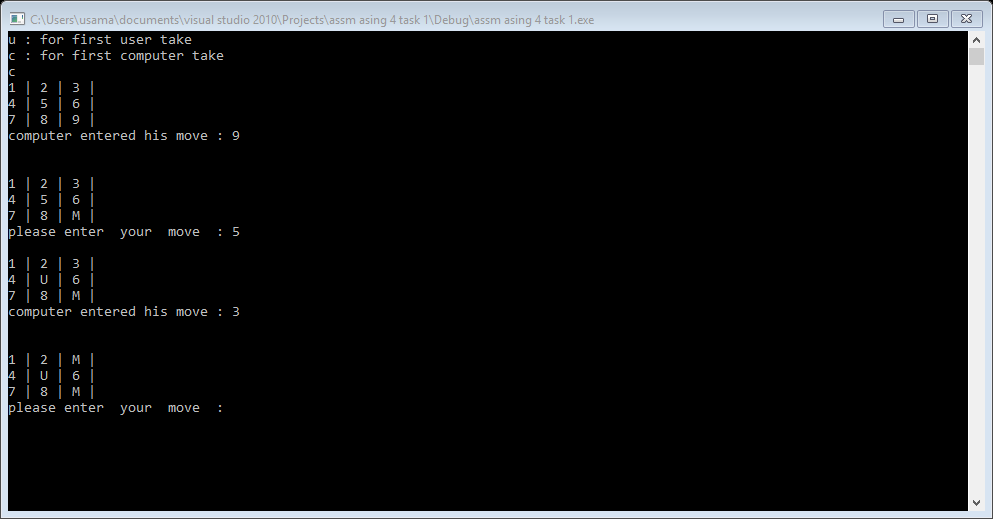
while(choice=='a'||choice=='A');

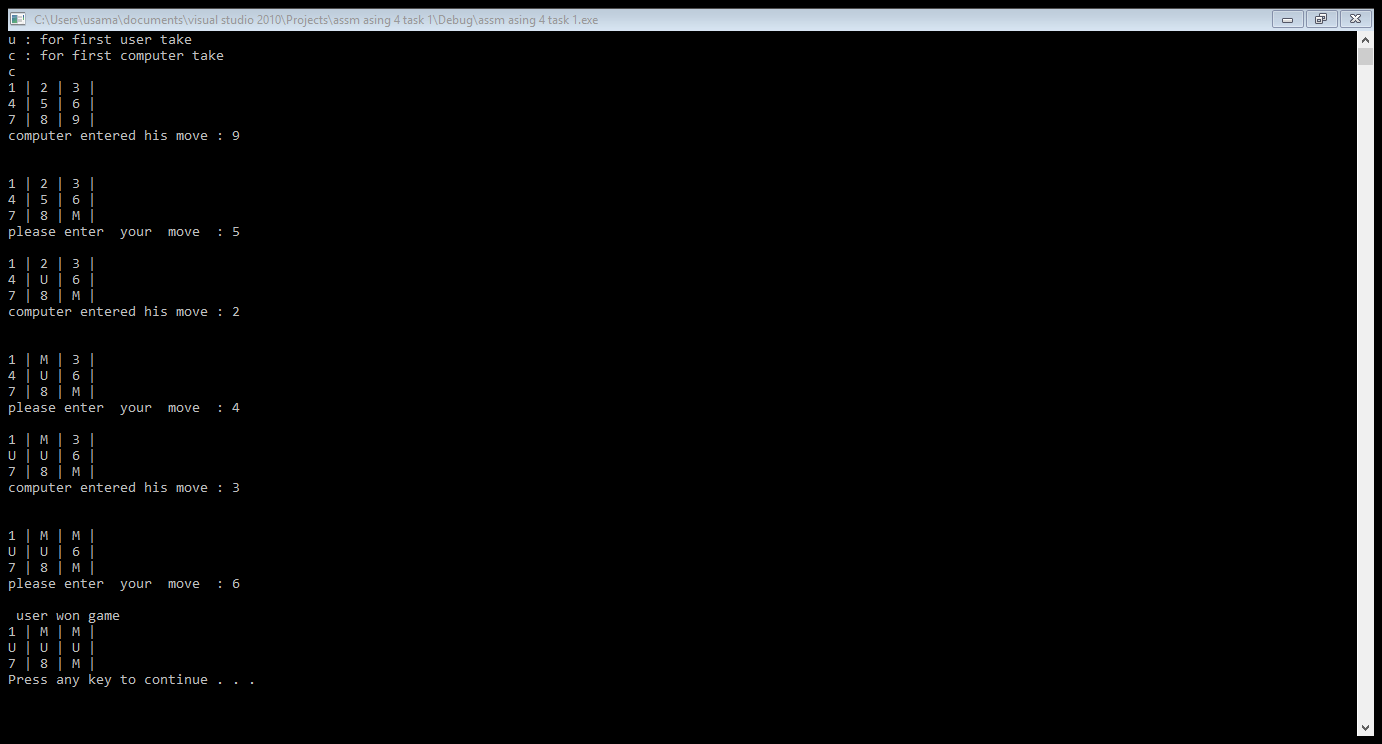
system("pause");

return 0;

}

Result:





Task : 2

#include<iostream>

#include<string>

using namespace std;

class network

{

private:

int id;

string name;

int age;

int \*flist;

public:

network():id(0),name(""),age(0)

{

flist=new int[5];

}

void setid(int id)

{

this->id=id;

}

void setname(string n)

{

name=n;

}

void setage(int a)

{

age=a;

}

void setflist(int \*fl)

{

fl=new int[10];

for(int i=0; i<10; i++)

{

flist[i]=fl[i];

}

}

int getid()

{

return id;

}

string getname()

{

return name;

}

int getage()

{

return age;

}

int\* getflist()

{

return flist;

}

~network()

{

delete []flist;

flist=NULL;

}

}w[15];

void idsnamesages();

void friendsa();

void open();

void have();

void mfriends(string n,string m);

int main()

{

int choice=0;

string n;

string m;

idsnamesages();

friendsa();

do

{

cout<<"1 : for open your id "<<endl;

cout<<"2 : see for thoes persons have ids"<<endl;

cout<<"3 : for search mutual friends "<<endl;

cout<<"4 : for terminate the programe"<<endl;

cin>>choice;

if(choice==1)

{

open();

}

else if(choice==2)

{

have();

}

else if(choice==3)

{

cout<<"eneter first name : ";

cin>>n;

cout<<"eneter 2nd name : ";

cin>>m;

mfriends(n,m);

}

else

{

cout<<"you press invalid "<<endl;

cout<<"1 : for taking again input "<<endl;

cout<<"2 : for exit"<<endl;

cin>>choice;

}

cout<<"if you see this press 1 for more option"<<endl;

cin>>choice;

system("cls");

cout<<"1 : for taking again input "<<endl;

cout<<"2 : for exit"<<endl;

cin>>choice;

}

while(choice==1);

//cout<<w[0].getname();

system("pause");

return 0;

}

void idsnamesages()

{

int ids=8000;

//network w[15];

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

{

w[i].setid(ids);

ids++;

}

w[0].setname("usama");

w[1].setname("zain");

w[2].setname("usman");

w[3].setname("talha");

w[4].setname("ali");

w[0].setage(19);

w[1].setage(22);

w[2].setage(17);

w[3].setage(20);

w[4].setage(18);

}

void friendsa()

{

int friends[3];

int id=8000;

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

{

friends[i]=id;

id++;

}

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

{

w[i].setflist(friends);

}

id--;

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

{

friends[i]=id;

id++;

}

for(int i=3; i<5; i++)

{

w[i].setflist(friends);

}

}

void open()

{

int sid=0;

cout<<"enter your id to open your acount : ";

cin>>sid;

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

{

if(w[i].getid()==sid)

{

cout<<"you searched this id :"<<w[i].getid()<<endl;

cout<<"your name is : "<<w[i].getname()<<endl;

cout<<"your age : "<<w[i].getage()<<endl;

cout<<"your friends is : "<<\*(w[i].getflist())<<endl;

}

}

}

void have()

{

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

{

cout<<w[i].getname()<<endl;

}

}

void mfriends(string n,string m)

{

int i=0;

int j=0;

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

{

if(w[i].getname()==n)

{

break;

}

}

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

{

if(w[j].getname()==m)

{

break;

}

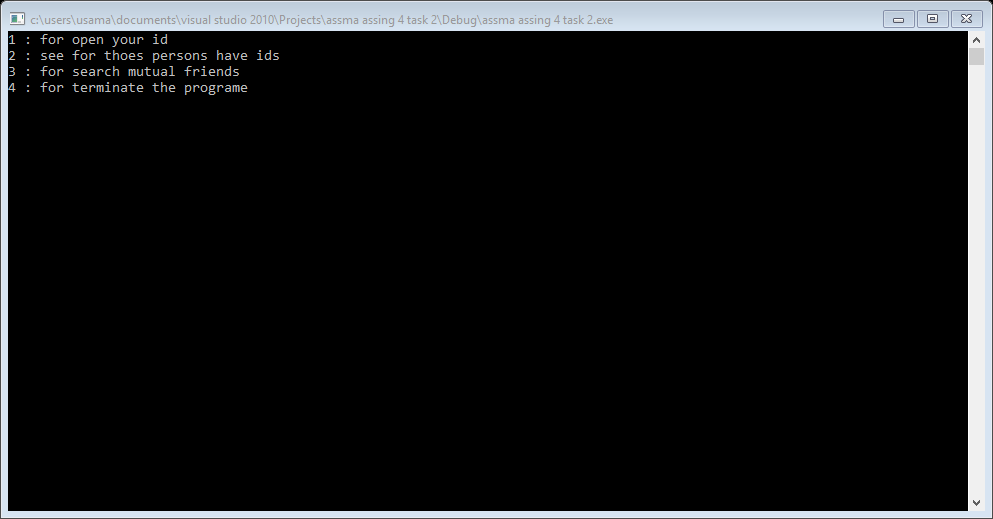
}

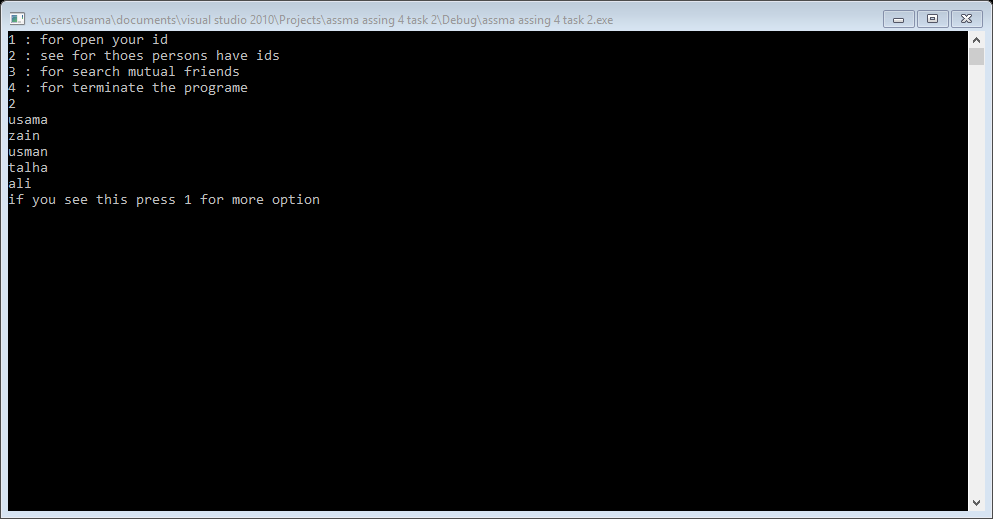
cout<<w[i].getname()<<" friends are : "<<w[i].getflist()<<endl;

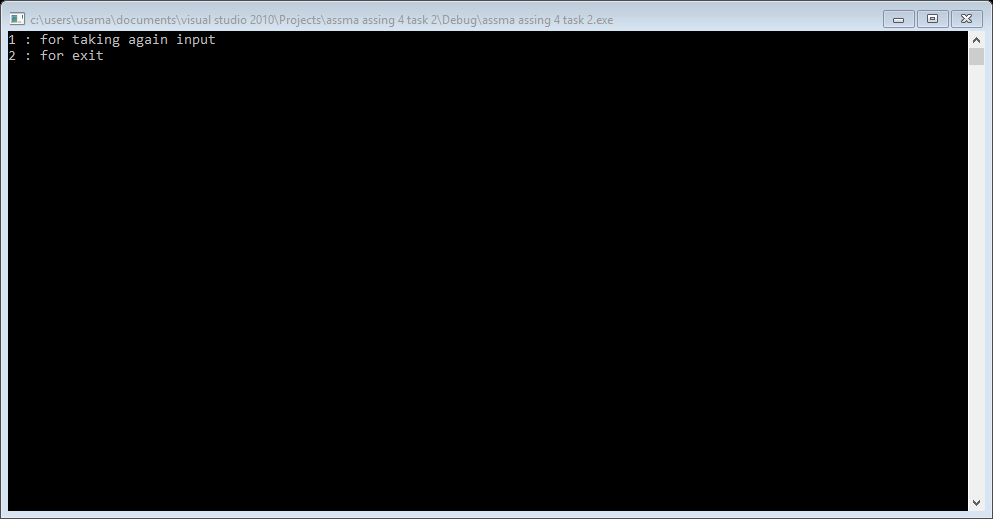
cout<<w[j].getname()<<" friends are : "<<w[i].getflist()<<endl;

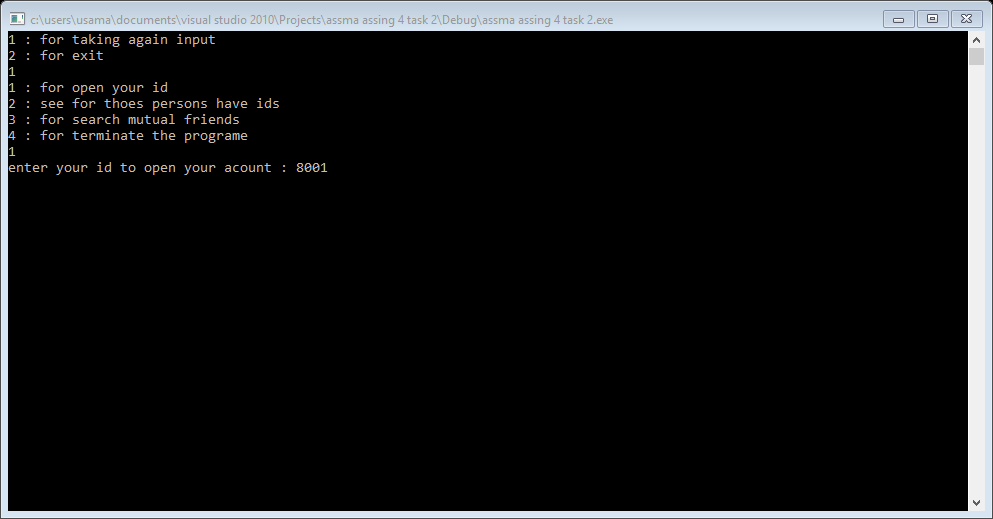
}

Result:









Task 4:

#include<iostream>

using namespace std;

class angle

{

private:

int degree;

float mints;

char direction;

int degreel;

float mintsl;

char directionl;

public:

void l(int d, int m, char dir)

{

degree=d;

mints=m;

direction=dir;

}

void a(int d, int m, char dir)

{

degreel=d;

mintsl=m;

directionl=dir;

}

void input()

{

int d;

float m;

char dir;

int dl;

float ml;

char dirl;

cout<<"enter degree for logitude : ";

cin>>d;

cout<<"enter mints for logitude : ";

cin>>m;

cout<<"enter direction for logitude : ";

cin>>dir;

l(d,m,dir);

cout<<endl;

cout<<"enter degree for latitude : ";

cin>>dl;

cout<<"enter mints for latitude : ";

cin>>ml;

cout<<"enter direction for latitude : ";

cin>>dirl;

a(dl,ml,dirl);

}

void showl()

{

cout<<degree<<"\xf8"<<mints<<"'"<<direction<<endl;

}

void showa()

{

cout<<degreel<<"\xf8"<<mintsl<<"'"<<directionl<<endl;

}

}a;

int main()

{

a.input();

a.showl();

a.showa();

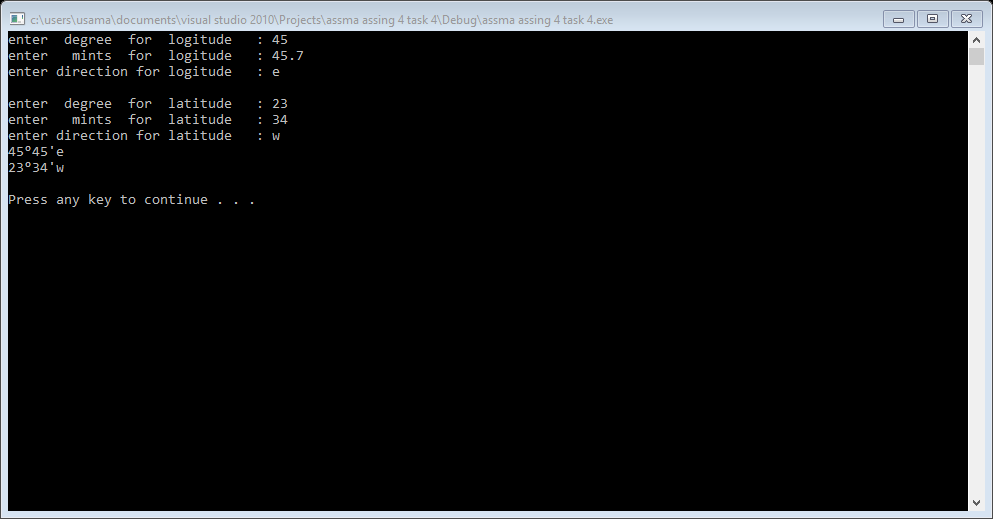
cout<<endl;

system("pause");

return 0;

}

Result:



Task 5:

#include<iostream>

using namespace std;

class matrix

{

private:

int row;

int column;

int \*\*mat;

public:

matrix(int r, int c)

{

row=r;

column=c;

mat[row][column];

}

void putel(int r, int c,int t)

{

row=r;

column=c;

}

int getelr(int r , int c)

{

return r;

}

int getelc(int r , int c)

{

return c;

}

matrix()

{

}

~matrix()

{

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

{

delete []mat[j];

}

delete []mat;

}

};

int main()

{

int temp=3433;

matrix m1(4,3);

m1.putel(7,4,temp);

temp=m1.getelr(7,4);

cout<<temp<<endl;

temp=m1.getelc(7,4);

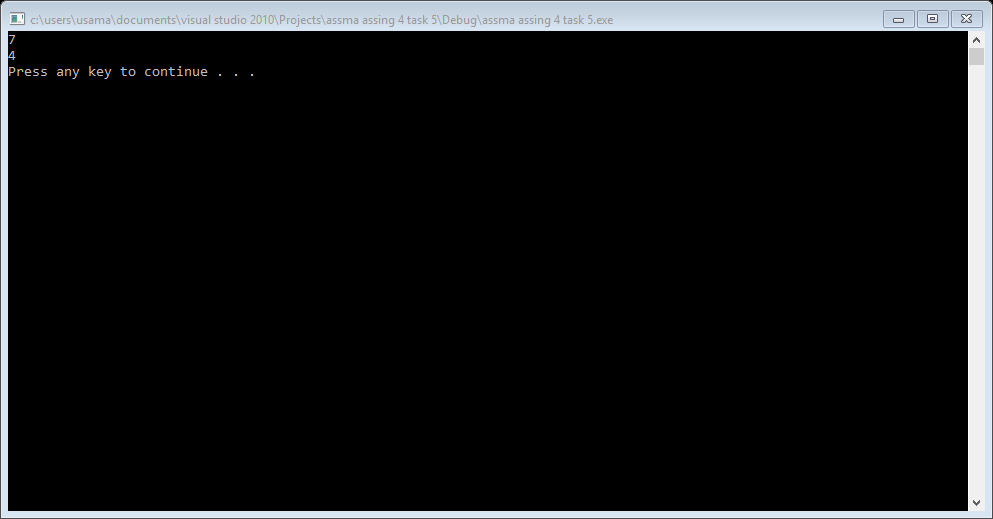
cout<<temp<<endl;

system("pause");

return 0;

}

Result:



Task 6:

#include<iostream>

using namespace std;

class student

{

private:

int rollno;

double cnic;

public:

string name;

student( int r, double c) //parametrized constructor

{

rollno=r;

cnic=c;

}

student() //default constructor

{}

int getrollno()

{

return rollno;

}

double getcnic()

{

return cnic;

}

};

class undergraduate :public student

{

private:

int semester\_no;

int cgpa;

int student\_id;

public:

undergraduate(int n, int c, int sid) //parametrized constructor

{

semester\_no=n;

cgpa=c;

student\_id=sid;

}

undergraduate() //default constructor

{ }

};

class freshmen :public undergraduate

{

public:

freshmen(string n)

{

name=n;

}

freshmen()

{}

};

class sophomore :private undergraduate // all data member of undergraduate is private her

// if we derive more classes from sophomore they cannot

{ //access undergraduate class data member.

public: //all public data of undergraduate behaive like private

sophomore(string n) //data member of sophomore class.

{

name=n;

}

sophomore()

{}

};

class junior :private undergraduate // all data member of undergraduate is private her

{ // if we derive more classes from junior they cannot

public: //access undergraduate class data member.

junior(string n)

{

name=n;

}

junior()

{}

};

class senior :public undergraduate

{

public:

senior(string n)

{

name=n;

}

senior()

{}

};

int main()

{

int r=8195;

double cnic=135527;

student s(r,cnic);

undergraduate u;

cout<<"cnic of student of base class :"<<u.getcnic()<<endl; // here we access base class private variable by using punlic

// public function of base class and use drive class object

cout<<"roll no of base class student : "<<u.getrollno()<<endl;

junior j;

//j.getcnic(); // it is inaccessibel because it private inherited from base class.

// so, in junior class all public and protected data of base class

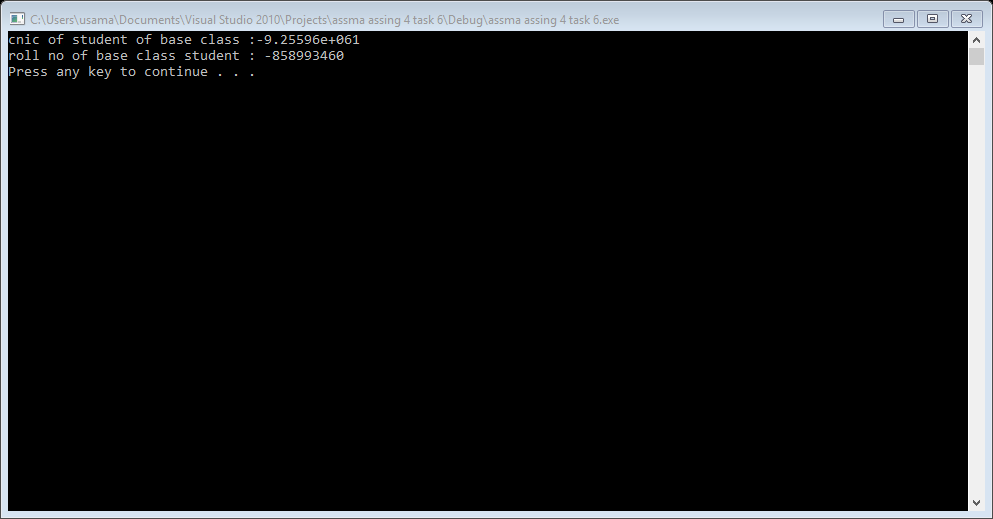
//behave like private. so private data member not accessibal in main().

system("pause");

return 0;

}

Result



Task 7:

#include<iostream>

#include<string>

using namespace std;

class employee

{

private:

string name;

string dob;

string doj;

int cnic;

public:

employee(string n, string db, string dj):name(n),dob(db),doj(dj)

{}

employee()

{ }

void setname(string n)

{

name=n;

}

void setdob(string db)

{

dob=db;

}

void setdoj(string dj)

{

doj=dj;

}

void setcnic(int c)

{

cnic=c;

}

string getname()

{

return name;

}

string getdob()

{

return dob;

}

string getdoj()

{

return doj;

}

int getcnic()

{

return cnic;

}

~employee()

{

cout<<"i am destructor of employee"<<endl;

}

};

class admin

{

private:

int experience;

int grade;

public:

admin(int e, int g):experience(e),grade(g)

{ }

admin()

{}

void setexperience(int ex)

{

experience=ex;

}

void setgrade(int g)

{

grade=g;

}

int getexperices()

{

return experience;

}

int getgrade()

{

return grade;

}

~admin()

{

cout<<"i am destructure of admin"<<endl;

}

};

class accounts

{

private:

int salary;

int office\_no;

public:

accounts(int s, int o):salary(s),office\_no(o)

{ }

accounts()

{}

void setsalary(int s)

{

salary=s;

}

void setoffice\_no(int no)

{

office\_no=no;

}

int getsalary()

{

return salary;

}

int getoffice\_no()

{

return office\_no;

}

~accounts()

{

cout<<"i am destructure of accounts class"<<endl;

}

};

class faculty

{

private:

string degree;

int no\_p\_papers;

string specialization;

public:

faculty(string d, int no, string sp):degree(d),no\_p\_papers(no),specialization(sp)

{}

void setdegree(string d)

{

degree=d;

}

void setno\_P\_papers(int n)

{

no\_p\_papers=n;

}

void setspecialization(string sp)

{

specialization=sp;

}

string getdegree()

{

return degree;

}

int getno\_p\_papers()

{

return no\_p\_papers;

}

string getspecialization()

{

return specialization;

}

};

int main()

{

employee s;

admin a;

system("pause");

return 0;

}