**ASSIGNMENT NUMBER 1**

**STATEMENT**: Develop an object oriented program in C++ to create a database of student information system containing the following information: Name, Roll number, Class, division, Date of Birth, Blood group, Contact address, telephone number, driving license no. etc Construct the database with suitable member functions viz, static member functions, friend class/ friend function, this pointer, inline code and dynamic memory allocation operators-new and delete. Implement all the keywords as mentioned in the problem statement.

**AIM**:To develop a student database system using OOP concepts.

**DESCRIPTION**:Create a class ‘student’ with friend functions to read and display student information.

Source Code:

#include<iostream>

#include<string>

#include<cstring>

using namespace std;

class person

{

private:

char name[40];

char dob[15];

char bdg[15];

char add[70];

char mob[15];

char lic\_no[10];

public:

static int count; //Static as it will be common for all instance

friend class student; //friend class to access it private members

person()

{

char \*name=new char[40];

char \*dob=new char[80];

char \*bdg=new char[15];

strcpy(add,"");

strcpy(mob,"");

strcpy(lic\_no,"");

}

static void recordcount()

{

cout<<"\n Total no of records : "<<count<<endl;

}

};

class student{

private:

char cls[70],div[15];

public:

student(){

strcpy(cls,"");

strcpy(div,"");

}

void getdata(person \*obj);

void displaydata(person \*obj);

friend class person;

};

int person::count=0;

void student::getdata(person \*obj)

{

cout<<"\n Enter Name : ";

cin>>obj->name;

cout<<"\n Enter DOB (dd/mm/YYYY): ";

cin>>obj->dob;

cout<<"\n Enter Blood Group : ";

cin>>obj->bdg;

cout<<"\n Enter Branch : ";

cin>>this->cls;

cout<<"\n Enter Division: ";

cin>>this->div;

cout<<"\n Enter Mobile no : ";

cin>>obj->mob;

cout<<"\n Enter Address : ";

cin>>obj->add;

cout<<"\n Enter License number: ";

cin>>obj->lic\_no;

obj->count++;

}

inline void student::displaydata(person \*obj)

{

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

cout<<"\t\tSTUDENT DATABASE";

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

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

cout<<"DOB : "<<obj->dob<<endl;

cout<<"Blood Group : "<<obj->bdg<<endl;

cout<<"Branch : "<<this->cls<<endl;

cout<<"Division : "<<this->div<<endl;

cout<<"Mobile no : "<<obj->mob<<endl;

cout<<"Address : "<<obj->add<<endl;

cout<<"License Number : "<<obj->lic\_no<<endl;

}

int main()

{

student \*p1[30];

person \*p2[30];

int n=0, c, i;

do

{

cout<<"\n\t Menu";

cout<<"\n 1.Enter Details of Student";

cout<<"\n 2.Display Student Details";

cout<<"\n 3.Exit";

cout<<"\n Enter your choice: ";

cin>>c;

switch(c)

{

case 1:

cout<<"\n Enter Details";

cout<<"\n";

p1[n]=new student;

p2[n]=new person;

p1[n]->getdata(p2[n]);

n++;

person::recordcount();

break;

case 2:

for(i=0;i<n;i++)

{

p1[i]->displaydata(p2[i]); //inline function will be copied here by the compiler

}

person::recordcount();

break;

case 3:

break;

default:

cout<<"\nENTER RIGHT OPTION!!\n";

break;

}

}

while(c!=3);

delete \*p1; //dynamic memory de-allocation

delete \*p2;

return 0;

}

**OOP CONCEPT USED**:

1. **Friend function**-:A friend function of a class is defined outside that class' scope but it has the right to access all private and protected members of the class.
2. **This pointer**-:In C++, this pointer is used to represent the address of an object inside a member function.
3. **Inline function**-:C++ provides an inline functions to reduce the function call overhead. Inline function is a function that is expanded in line when it is called.
4. **Dynamic memory allocation**-:Dynamic memory allocation in C/C++ refers to performing memory allocation manually by programmer.

C++ supports these functions and also has 2 operators ‘new’ and ‘delete’ that perform the task of allocating and freeing the memory in a better and easier way.

5. **Static Member Function** -:A static member function is a special member function, which is used to access only static data members, any other normal data member cannot be accessed through static member function..

**CONCLUSION**: In this following assignment, we learned the use of friend functions and this pointer. We also learned the use of dynamic memory allocation.