CHAPTER-6

Inheritance: Extending Classes

VERY SHORT/SHORT ANSWER QUESTIONS

1. What is inheritance? Discuss its various forms.

Ans. Inheritance is the capability of one class to inherit properties from another class.

Forms of Inheritance:

1. Single Inheritance: One base class and one super class.
2. Multiple Inheritance: Multiple base class and one sub class.
3. Hierarchical Inheritance: One base class and multiple sub class.
4. Multi-level Inheritance: A subclass inherits from a class that itself inherits from another class.
5. Hybrid Inheritance: A subclass inherits from multiple base classes and all of its base classes inherit from a single base class.

2. Discuss various reasons that support the concept of inheritance in Object Oriented Languages.

Ans. The reasons that support the existence of inheritance concept in OO Languages are:

1. 'Inheritance' is capable of expressing the inheritance relationship of real-world models.
2. 'Inheritance' facilitates the code reusability.
3. 'Inheritance' is capable of simulating the transitive nature of real-world's inheritance.

3. Differentiate between public and protected visibility in context of Object Oriented Programming giving suitable examples for each.

Ans.

Public visibility	Protected visibility
The public derivation means that the derived class can access the public and protected members of the base class but not the private members of the base class.	The protected derivation means that the derived class can access the public and private members of the base class protectedly.
With publicly derived class, the public members of the base class become the public members of the derived class, and the protected members of the base class become the protected members of the derived class.	With protectedly derived class, the public and protected members of the base class become protected members of the derived class.

Example:
`class super`
`{ private:`
`int x; void get();`
`public:`
`int y; void put();`
`protected:`
`int z; void disp();`
`};`
`class sub:public super`
`{ private:`
`int a; void init();`
`public:`
`int b; void readit();`
`protected:`
`int c; void writeit();`
`};`
	Example:
	`class super`
	`{ private:`
	`int x; void get();`
	`public:`
	`int y; void put();`
	`protected:`
	`int z; void disp();`
	`};`
	`class sub:protected super`
	`{ private:`
	`int a; void init();`
	`public:`
	`int b; void readit();`
	`protected:`
	`int c; void writeit();`
	`};`

4. How does the visibility mode control the access of members in the derived class? Explain with examples.

Ans. Visibility mode controls the access of member in derived class by using access specifier. Below table shows different visibility modes.

Public inheritance			
Base access specifier	**Derived access specifier**	**Derived class access?**	**Public access?**
Public	Public	Yes	Yes
Private	Private	No	No
Protected	Protected	Yes	No

Private inheritance			
Base access specifier	**Derived access specifier**	**Derived class access?**	**Public access?**
Public	Private	Yes	No
Private	Private	No	No
Protected	Private	Yes	No

Example:

```
class A{ private:
           int a; void init();
         public:
           int b; void get();
        protected:
           float c; void put();
      };
class B:public A
{
      private:
            char arr[10];
      public:
            void enter();
};
class C:private A
{
      private:
            float f;
      public:
            void show();
};
```

5. **What should be the structure of a class when it has to be a base class for other classes?**

Ans.

class *classname*
{ private:
//members not intended to be inherited
public:
//members intended to be inherited and available to every function
protected:
//members intended to be inherited but not intended to be public
};

6. **Discuss a situation in which the private derivation will be more appropriate as compared to public derivation.**

Ans. When we want that the derived class has the direct access privilege only to the non-private members of the base class, in that situation the private derivation will be more appropriate as compared to public derivation.

7. **From a base class A, two classes B and C are deriving. B is deriving publicly from A whereas C is deriving privately. The classes B and C, otherwise, are doing the similar tasks. Discuss the similarities and differences between the functioning of B and c.**

Ans.

Similarities:
Private members of the base class A are not inherited in derived class B and class C at all.
Differences:
In class B, the public and protected members of the base class A remain public and protected whereas in class C, the public and protected members of the base class A become private members.

8. **Identify the error(s) and the responsible reason(s) in the following code snippet:**

```
class X  {
           public:
              int a;
              float b;
              char c;
         };
```

```
class Y:public X {
                    public:
                       int d;
                    private:
                       X::a;
                  };
```

Ans. In above code X::a is nor allowed as we cannot deny access to certain members of a base class when inheriting publicly. Above error can be solved inheriting the class privately as following:

```
class Y:private X {
                    public:
                       int d;
                        X::a;
                  };
```

9. **Identify the errors in the following code fragment. Discuss the responsible reasons for them.**

```
int glob;
class F{
   int glob;
      public:
        void readit()
        { cin>>glob; }
};
class W:public F {
    public:
       void test()
       { glob--; }
};
```

Ans. The compiler will not allow the class W to access F::glob as well as global glob because F::glob being private to F cannot be accessed directly by the class W, and global glob cannot be accessed in W as it is hidden here because F::glob is visible here but inaccessible.

Above error can be solved by writing following statement in test() method:

```
::glob--;
```

10. **Given the definition of following two classes. Write the constructor definition for both classes.**

```
class Alpha {
               int a;
               float b;
               char c;
            public:
               .........  //constructor definition
                       // has to become here
               :
           };
class Beta:public Alpha {
                          public:
                           ........
                           //constructor definition
                         };
```

Ans.

```
class Alpha {
  int a;
  float b;
  char c;
    public:
         Alpha(int i, float j,char k)
              { a=i; b=j; c=k;  }
};
class Beta:public Alpha {
```

```
      public:
        Beta(int p,float q,char r):Alpha(p,q,r)
        {   cout<<"Beta constructor...";  }
};
```

11. Define constructors for the classes defined below:

```
class Alpha {   int a;
              public:
                 float b;
                 .........  //the constructor definition
                 :
            };
class Beta {   int P;
             public:
                float q;
                .........  //the constructor definition
                :
            };
class Gamma {
                 Alpha A;
                 Beta B;
                 char X;
              public:
                 double y;
                 .........
                    //the constructor definition
                 :
             };
```

Ans.

```
class Alpha {   int a;
              public:
                 float b;
                 Alpha(int x,float y)
                 { a=x; b=y; }
            };
class Beta {   int P;
             public:
                float q;
                Beta(int i,float j)
                 { p=i; q=j; }
            };
class Gamma {
                 //Alpha A;
                 //Beta B;
                 char X;
              public:
                 double y;
                 Gamma(char c,double d)
                 { x=c; y=d; }
             };
```

12. Consider the following code:

```
#include<iostream.h>
class A  { public:
              A() { cout<<"A";}
              ~A() { cout<<"~A"; }
         };
class B  { public:
```

```
            B() { cout<<"B";}
            ~B() { cout<<"~B"; }
        };
class C  { public:
            C() { cout<<"C";}
            ~C() { cout<<"~C"; }
          private:
            B c1;
            A c2;
        };
class D  { public:
            D() { cout<<"D";}
            ~D() { cout<<"~D"; }
        };
class E:public C
{  public:
     E() { cout<<"E";}
     ~E() { cout<<"~E"; }
   private:
     D e1;
     C e2;
};
int main()
{    E e;
     return 0;
}
```

If the program compiles and runs correctly, what dos it prints out?

Ans. Output:
BACDBACE~E~C~A~B~D~C~A~B

13. **How does the invocation of constructor differ in derivation of class and nesting of classes?**

Ans. In derivation of class first the base class constructor is invoked, followed by the derived class constructor, whereas in nested classes constructors of all the member objects are called before the constructors of the object enclosing other objects.

14. **A class One with data members int a and char b inherits from another class Two with data members float f and int x. Write definitions for One and Two for the following situations:**
(i) Objects of both the classes are able to access all the data members of both the classes.
(ii) Only the members of class One can access all the data members of both the classes.

Ans.

```
(i)  class Two
     {
          protected:
            float f;
            int x;

     };
     class One:private Two
     {
                int a;
                char b;
    };
(ii) class Two
     {
          public:
            float f;
            int x;
```

```
};
class One:public Two
{
     public:
        int a;
       char b;
};
```

15. **Assume a class Derv derived from a base class Base. Both classes contain a member function func() that takes no arguments. Write the definition for a member function of Derv that calls both the func()s.**

Ans.

```
class Base
{
    public:
     void func()
     {
          cout<<"base class";
     }

};
class Derv:public Base
{
    public:

     void func()
     {
          cout<<"derived class";
     }
     void callAll()
     {
          Base::func();
          func();
     }
};
```

16. **What will be the output of the following program?**

```
#include<iostream.h>
class Student
{
  public:
     Student (char pname[]=" ")
     {
          strcpy(name,pname);
          average=semesterHours=0;
     }
     void addCourse(int hours,float grade)
     {
          average=(semesterHours*average+grade);
          semesterHours+=hours;
          average=semesterHours;
     }
     int hours()
          {  return semesterHours; }
     float gpa()
          { return average; }
  protected:
     char name[40];
     int semesterHours;
```

```
      float average;
};
class GradStudent:public Student
{    public:
            GradStudent(char pname[]=" "):Student(pname)
            {     qual_grade=0;
            }
            int qualifier()
                {   return qual_grade; }
     protected:
            int qual_grade;
};
int main()
{
      Student commerce("Saurabh 7/.");
      GradStudent gs;
      commerce.addCourse(4,2.5);
      gs.addCourse(4,3.0);
      return 0;
}
```

Ans.

Above code will generate various compilation errors few of them are listed below–

i. `strcpy(name,pname);` gives error due to missing string.h file

ii. After adding the required header file code will execute but screen will appear blank due to missing output statements.

17.

```
#include<iostream.h>
class a
{
  public:
      void something()
      {   cout<<"a"; }
};
class b
{
  public;
      void something()
      {   cout<<"b"; }
};
class c:public a,public b {};
int main()
{
      c x;
      x.something();
      return 0;
}
```

(a) a::something() is called
(b) b:: something() is called
(c) Runtime Error
(d) Syntax Error

Ans.

(d) Syntax Error

18.

```
#include<iostream.h>
class basex
{      int x;
 public:
       void setx(int y) { x=y; }
```

	}; **class derived:basex { };** **What is the access level for the member function "setx" in the class "derived" above?** **(a) protected (b) private (c) local (d) public**
Ans.	(b) private
19.	**class professor {};** **class teacher:public virtual professor {};** **class reasearcher:public virtual professor {};** **class myprofessor:public teacher,public reasearcher {};** **Referring to the same code above, if a object of class "myprofrssor" were created, how many instances of professor will it contain?** **(a) 4 (b) 1 (c) 2 (d) 3**
Ans.	(b) 1
20.	**When does ambiguity arise in multiple inheritance? How can one resolve it? What are virtual base classes? What is their significance?**
Ans.	An ambiguity can arise when several paths exist to a class from the same base class. This means that a child class could have duplicate sets of members inherited from a single base class. This can be solved by using a virtual base class. When two or more objects are derived from a common base class, we can prevent multiple copies of the base class being present in an object derived from those objects by declaring the base class as virtual when it is being inherited. Such a base class is known as virtual base class. When a class is made virtual, necessary care is taken so that the duplication is avoided regardless of the number of paths that exist to the child class.
21.	**Answer the questions (i) and (iv) based on the following:** **class Student** **{** **int Rollno;** **char SName[20];** **float Marks1;** **protected:** **void Result();** **public:** **Student();** **void Enroll();** **void Display();** **};** **class Teacher** **{** **long TCode;** **char TName[20];** **protected:** **float Salary;** **public:** **Teacher ();** **void Enter();** **void Show();** **};** **class Course:public Student,private Teacher** **{** **long CCode[10]** **char CourseName[50];** **char StartDate[8],EndDate[8];** **public:** **Course();**

```
    void Commence();
    void CDetail();
};
```

(i) Write the names of member functions, which are accessible from objects of class Course.

(ii) Write the names of all data members, which is/are accessible from member function Commence of class Course.

(iii) Write the names of all the members, which are accessible from objects of class Teacher.

(iv) Which type of inheritance is illustrated in the above C++ code?

Ans.

(i) void Commence( ), void CDetail( ), void Enroll ( ), void Display ( );

(ii) CCode, CourseName, StartDate, EndDate, Salary

(iii) void Enter ( ), void Show ( );

(iv) Multiple inheritance

22. **Answer the questions (i) to (iv) based on the following:**

```
class Ball
{     char Btype[10];
   protected:
      float Rate;
      void CalcRate(float);
   public:
      Ball();
      void BInput();
      void BShow();
      void TEntry();
      void TDisplay();
};
class SoftToys:public Toys
{     char STName[20];
      float Weight;
   public:
      SofToys();
      void STEntry();
      void STDisplay();
};
class ElectronicToys:public Toys
{     char ETName[20];
      char No_of_Batteries;
   public:
      ElectronicToys();
      void ETEntry();
      void ETDisplay();
};
```

(i) Which type of Inheritance is shown in the above example?

(ii) How many bytes will be required by an object of the class SoftToys?

(iii) Write name of all the data members accessible from member functions of the class SoftToys.

(iv) Write name of all the member functions accessible from an object of the class ElectronicToys.

Ans.

In above code Ball class is not mentioned in inheritance process, it should be Toys class

(i) Hierarchical Inheritance

(ii) 38 Bytes

(iii) Rate, STName, Weight

(iv) ETEntery, ETDisplay, BInput, BShow, TEntery, TDisplay

23. **Answer the question (i) to (iv) based on the following code:**

```
class Trainer
{
      char TNo[5], TName[20], Specialisation[10];
```

```
        int Days;
    protected:
        float Remuneration;
        void AssignRem(float);
    public:
        Trainer();
        void TEntery();
        void TDisplay();
};
class Learner
{
        char Regno[10], LName[20],Prpgram[10];
    protected:
        int Attendeance, Grade;
    public:
        Learner();
        void LEntery();
        void LDisplay();
};
class Institute:public Learner, public Trainer
{
        char ICode[10],IName[20];
    public:
        Institute();
        vod IEntry();
        void IDisplay();
};
```

(i) Which type of Inheritance is depicted by the above example?
(ii) Identify the member function(s) that cannot be called directly from the objects of class Institute from the following:
TEntry()
LDisplay()
IEntry()
(iii) Write name of all the member(s) accessible from member functions of class Institute.
(iv) If class Institute was derived privately from class Learner and privately from class Trainer, then, name the member function(s) that could be accessed through Objects of class Institute.

Ans.

(i) Multiple Inheritance
(ii) None (Since all of these functions can be called from object of class Institute).
(iii) Data Members: Remuneration, Attendance, Grade, ICode, IName
Member Functions: AssignRem(), TEntry(), TDisplay(), LEntry(), LDisplay(), IEntry(), IDisplay()
(iv) IEntry(), IDisplay

24. **Consider the following and answer the questions give below:**

```
class MNC
{     char Cname[25];      //Compay name
    protected:
      char Hoffice[25];    //Head office
    public:
      MNC();
      char Country[25];
      void EnterData();
      void DisplayData();
};
class Branch:public MNC
{     long NOE   //Number of employees
      char Ctry[25];   //Country
```

```
    protected:
      void Association();
    public:
      Branch();
      void Add();
      void Show();
};
class Outlet:public Branch
{     char State[25];
    public:
      Outlet();
      void Enter();
      void Output();
};
```

(i) Which class' constructor will be called first at the time of declaration of an object of class Outlet?
(ii) How many bytes does a object belonging to class Outlet require?
(iii) Name the member function(s), which are accessed from the object(s) of class Outlet.
(iv) Name the data member(s), which are accessible from the object(s) of class Branch.

Ans.

(i) class MNC
(ii) 129 Bytes
(iii) Enter(), Output(), Add(), Show(), EnterData(), DisplayData()
(iv) Country[25]

25. **Answer the questions (i) and (iv) based on the following:**

```
class Director
{
    long DID;    //Director identification number
    char Name[20];
  protected:
     char Description[40];
     void Allocate();
  public:
     Director();
     void Assign();
     void Show();
};
class Factory:public Director
{
     int FID;            //Factory ID
     char Address[20];
  protected:
     int NOE             // No Of Employees
  public:
     Factory();
     void Input();
     void Output();
};
class ShowRoom:private Factory
{
   int SID;               //ShowRoom ID
   char City[20];
 public:
   ShowRoom();
   void Enter();
   void Display();
};
```

	(i) Which type of inheritance out of the following is illustrated in the above C++ code? **a) Single level inheritance** **b) Multi level inheritance** **c) Multiple inheritance** **(ii) Write the names of data members, which are accessible by objects of class type ShowRoom.** **(iii) Write the names of all member functions which are accessible by objects of class type ShowRoom.** **(iv) Write the names of all members, which are accessible from member functions of class Factory.**
Ans.	(i)Multi level inheritance (ii) None (iii) Enter(), Display(), Enter(), Display() (iv) Data Members: FID, Address, NOE Member Functions: Input(), Output()

26.	**Answer the questions (i) to (iv) based on the following:**
	`class FaceToFace`
	`{`
	`  char CenterCode[10];`
	`public:`
	`  void Input( );void Output( );`
	`};`
	`class Online`
	`{`
	`  char website[50];`
	`public:`
	`  void SiteIn( );`
	`  void SiteOut( );`
	`};`
	`class Training: public FaceToFace, private online`
	`{`
	`  long Tcode;`
	`  float charge;`
	`  int period;`
	`public:`
	`  void Register( );`
	`  void show( );`
	`};`
	(i) Which type of inheritance is shown in the above example?
	(ii) Write names of all the member functions accessible from Show( ) function of class Training.
	(iii) Write name of all the member accessible through an object of class Training.
	(iv) Is the function Output( ) accessible inside the function SiteOut( )? Justify your answer?

Ans.	(i) Multiple Inheritance (ii) Register( ) Siteln( ), SiteOut( ), Input( ), Output( ) (iii) Register( ), Show( ), Input( ), Output( ). (iv) No, function Output( ) is not directly accessible inside the function SiteOut( ), because Output( ) is a member function of class FaceToFace and SiteOut( ) is a member function of class Online, and the classes FaceToFace and Online are two independent classes.

LONG ANSWER QUESTIONS

1.	**Imagine a publishing company that markets both books and audio-cassette versions of its works. Create a class publication that stores the title (a string) ad price (type float) of a publication. From this class derive two classes: book, which adds a page count (type int); and tape, which adds a playing time in minutes (type float). Each of these three classes should have a getdata() function to get its data from the user at the keyboard, and a putdata()**

function to display its data.
Write a main() program to test the book and tape classes by creating instances of them, asking the user to fill in their data with getdata(), and then displaying the data with putdata().

Ans.

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
class publication
{
      char title[20];
      float price;
   public:
      void getdata()
      {    cout<<"Enter title: ";
           gets(title);
           cout<<"Enter price: ";
           cin>>price;
      }
      void putdata()
      {     cout<<"Title: "<<title<<endl;
           cout<<"Price: "<<price<<endl;
      }
};
class book:public publication
{
      int page_count;
     public:
      void getdata()
      {    publication::getdata();
           cout<<"Enter page count: ";
           cin>>page_count;
      }
      void putdata()
      {    publication::putdata();
           cout<<"Page count: "<<page_count<<endl;
      }
};
class tape:public publication
{
      float play_time;
     public:
      void getdata()
      {     publication::getdata();
           cout<<"Enter Play time: ";
           cin>>play_time;
      }
      void putdata()
      {     publication::putdata();
           cout<<"Play time: "<<play_time<<endl;
      }
};
void main()
{    clrscr();
      book b;
      tape t;
      b.getdata();
```

```
      b.putdata();
      t.getdata();
      t.putdata();
      getch();
}
```

2. **Assume that a bank maintains two kinds of accounts for customers, one called as savings account and the other as current account. The saving account provides compound interest ad withdrawal facilities but not cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintains a minimum balance and if the balance falls below this level, a service charge is imposed.**
Create a class Account that stores customer name, account number and opening balance.
From this derive the classes Current and Saving to make them more specific to their requirements. Include necessary member functions in order to achieve the following tasks:
(i) deposit an amount for a customer and update the balance
(ii) display the account details
(iii) compute and deposit interest
(iv) withdraw amount for a customer after checking the balance and update the balance.
(v) check for the minimum balance (for current account holders), impose penalty, if necessary, and update the balance.
Implement these without using any constructor.

Ans.

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<process.h>
int const min=500;
class Account
{     char name[20];
      long ano;

     public:

      void getdata()
      {     cout<<"Enter customer name: ";
            gets(name);
            cout<<"Enter account no.: ";
            cin>>ano;
            cout<<"Enter opening balace: ";
            cin>>balance;
      }
      void display()
      {     cout<<"Customer name: "<<name<<endl;
            cout<<"Account no: "<<ano<<endl;
            cout<<"Balance :"<<balance;
      }
};
class Current:public Account
{     float depo,with,pen;
     public:
      void deposit()
      {     cout<<endl<<"Enter money to deposit: ";
            cin>>depo;
            display();
            balance=balance+depo;
            cout<<endl<<"After deposit main balance is: "<<balance<<endl;
      }
      void withdraw()
```

```
      {     cout<<endl<<"Enter money to withdraw: ";
            cin>>with;
            if(with<balance)
            {     display();
            balance=balance-with;
            cout<<endl<<"After withdraw main balance is: "<<balance<<endl;
            }
            else
            {   cout<<endl<<"You cannot withdraw money....."<<endl;
             }
      }
      void check_bal()
      {     if(balance<min)
            {     cout<<"Opening balance should not be less than 500...."<<endl;
                  balance=balance-150;
                  cout<<endl<<"After penalty main balance is: "<<balance<<endl;
            }
      }
};
class Savings:public Account
{     float depo,with,intr;
     public:
      void deposit()
      {     cout<<endl<<"Enter money to deposit: ";
            cin>>depo;
            display();
            balance=balance+depo;
            cout<<endl<<"After deposit main balance is: "<<balance<<endl;
      }
      void withdraw()
      {     cout<<endl<<"Enter money to withdraw: ";
            cin>>with;
            if(with<balance)
            {     display();
            balance=balance-with;
            cout<<endl<<"After withdraw main balance is: "<<balance<<endl;
            }
            else
            {   cout<<"You cannot withdraw money....."<<endl;
             }
      }
      void cal_intr()
      {     intr=(balance*2)/100;
            balance=balance+intr;
            cout<<endl<<"After calculating interest balance is: "<<balance;
      }
};
void main()
{     clrscr();
      Current c;
      Savings s;
      char ch;
      int choice,ch2;
      cout<<"Enter 'S' for saving and 'C' for current: ";
      cin>>ch;
      if(ch=='C'||ch=='c')
```

```
      {     c.getdata();
            c.check_bal();
            l2:cout<<"\n 1. Display \n 2.Deposit \n 3.Withdraw \n 4. Exit \n";
            cout<<"Enter your choice: ";
            cin>>choice;
            switch(choice)
            {
            case 1: c.display();
                  goto l2;
                  break;
            case 2: c.deposit();
                  goto l2;
                  break;
            case 3: c.withdraw();
                  goto l2;
                  break;
            case 4: exit(0);
            }
      }
      else if(ch=='S'||ch=='s')
      {     s.getdata();
            l1:cout<<"\n 1. Display \n 2.Deposit \n 3.Withdraw \n 4.Calculate
iterest \n 5. Exit \n";
            cout<<"Enter your choice: ";
            cin>>ch2;
            switch(ch2)
            {
            case 1: s.display();
                  goto l1;
                  break;
            case 2: s.deposit();
                  goto l1;
                  break;
            case 3: s.withdraw();
                  goto l1;
                  break;
            case 4: s.cal_intr();
                  goto l1;
                  break;
            case 5: exit(0);
            }
      }
      else
            cout<<"Wrong choice.........."<<endl;
      getch();
}
```

3. **Modify the program 2 of Type C to include constructors for the three classes.**

Ans.

```
class Account
{     public Account()
      {     strcpy(name,"NULL"); ano=0;  balance=0.0;  }
      // same as above
};
class Current:public Account
{     public Current()
      {     depo=0.0;with=0.0;pen=0.0; }
      // same as above
```

```
};
class Savings:public Account
{
      public Savings()
      {     depo=0.0;with=0.0;intr=0.0; }
      // same as above
};
void main()
{     // same as above
}
```

4. **Write a declaration for a class Person which has the following:**

- **data members name, phone**
- **set and get functions for every data member**
- **a display function**
- **a destructor**

(i) For the Person class above, write each of the constructor, the assignment operator, and the getName member function. Use member initialization lists as often as possible.

(ii) Given the Person class above, write the declaration for a class Spouse that inherits from Person and does the following:

- **has an extra data member spouseName**
- **redefines the display member function.**

(iii) For the Spouse class above, write each of the constructors and the display member function. Use member initialization lists as often as possible.

Ans.

```
class Person
{
      char name[20];
      long phone;
    public:
      void set()
      {  strcpy(name,"NULL");
         phone=7878963522;
      }
      void get()
      {  cout<<"Enter name: ";
         gets(name);
         cout<<"Enter phone: ";
         cin>>phone;
      }
      void display()
      {  cout<<"Name: "<<name<<endl;
         cout<<"Phone: "<<phone<<endl;
      }
      Person()
      {   strcpy(name,"Rahul");
          phone=9965869922;
      }
      Person(char na[20],long ph)
      {    name=na;
           phone=ph;
      }
      void getName()
      {
            cout<<"Enter name:";
            gets(name);
      }
```

```
};
class Spouse:public Person
{
      char spouseName[20];
     public:
      void getName()
      {
            cout<<"Enter name:";
            gets(spousename);
      }
      void display()
      {  cout<<"Name: "<<name<<endl;
         cout<<"Phone: "<<phone<<endl;
         cout<<"spouse name: "<<spousename<<endl;
      }
      Spouse()
      {   strcpy(spouseName,"NULL");
      }
      Spouse(char sn[20])
      {    spouseName=sn;
      }
};
```

5. **Modify the above program so that Clerk and Officer classes contain object of another class called Education that holds two pieces of educational information, namely qualification and experience. Incorporate the functions for handling this additional information.**

Ans. Question referring the class are not mentioned in any of the above question.

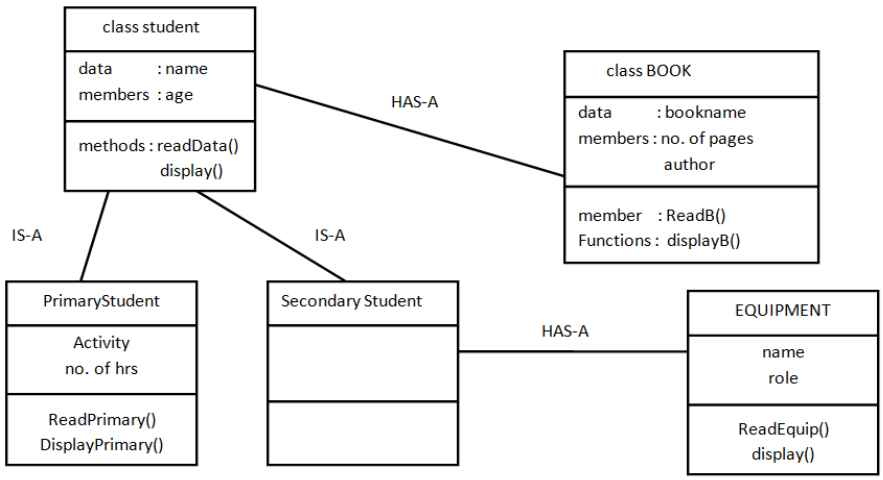
6. **Write a C++ to read and display information about employees and managers. Employee is a class that contains employee number, name, address ad department. Manager class contains all information of the employee class and a list of employees working under a manager.**

Ans.

```
#include<iostream.h>
#include<conio.h>
class employee
{
  public:
    int num,house ;
    char city[20], state[20], name[20],depart[20];
  public:
    void input()
    {
      cout<<"Enter the employe's name";
      cin>>name;
      cout<<"Enter the employe number";
      cin>>num;
      cout<<"Enter the address including house number ,city ,state";
      cin>>house>>city>>state;
      cout<<"enter the department";
      cin>>depart;
    }
    void output()
    {
      cout<<"\nemploye's infomation:";
      cout<<"\n"<<name<<"\n"<<num<<"\n"<<"address -:" <<"\n"<<house<<"
"<<city<<"\n"<<state;
      cout<<"\n"<<depart;
    }
```

```
};
class manager: public employee
{
    char name[20];
    int n ,i;
    public:
     void getdata()
     {
        cout<<"Enter the manager's name";
        cin>>name;
        cout<<"enter the total number of employe's working under him";
        cin>>n;
     }
     void info();
};
void manager::info()
{
   getdata();
   for(i=1;i<=n;i++)
   {
      input();
   }
   cout<<name;
   cout<<"\nemploye's are-:n" ;
   for(i=1;i<=n;i++)
   {
      cout<<i<<" employe-:" ;
      output();
   }
}
void main()
{
   class manager M;
   clrscr();
   M.info();
   getch();
}
```

7. **Create the following class hierarchy in C++.**



Ans.

```
class student
{     char name[20];
      int age;
```

```
   public:
      void readData();
      void display();
};
class Book
{     student Enrollno;
      char bookname[20],author[20];
      int no_of_pages;
   public:
      void ReadB();
      void displayB();
};
class PrimaryStudet:public student
{     char Activity[20];
      int no_of_hrs;
   public:
      void ReadPrimary();
      void DisplayPrimary();
};
class SecondaryStudet:public student
{};
class EQUIPMENT
{     char name[20];
      int role;
   public:
      void ReadEquip();
      void Display();
};
```