

Q1. Define inheritance and enlist its type.

Ans:

- The mechanism of **deriving a class from another** class is known as inheritance.

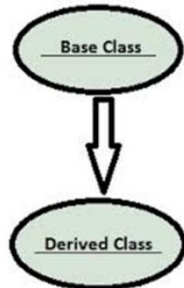
Types of inheritance are as follows:

- Single inheritance.
- Multiple inheritance
- Multilevel inheritance.
- Hierarchical inheritance
- Hybrid inheritance.

Q2. What is inheritance? Why inheritance used in c++?

Ans:

- The mechanism of **deriving a class from another** class is known as inheritance.
- The **class from which** another class is derived is called base class.
- **Class derived** is called as derived class.



•

Inheritance gives reusability of a code. In this new classes are created by using the properties of the existing classes.

It saves the time and money and increases reliability. Hence inheritance is used in C++.

Q3.Explain different visibility modes/modifiers used in inheritance.

Ans:

There are three types of visibility modes in inheritance:

1. private visibility.
2. protected visibility
3. public visibility.

1.Private visibility:

When a base class is **privately inherited** by a derived class, public and protected members of base class become private members of derived class and therefore the public and protected members of base class can be accessed by the member functions of the derived class.

Private members of base class are not inherited in derived class.

2.Public visibility:

when a base class is publicly inherited by a derived class, „public“ members of base class becomes public members of derived class and protected members of

base class becomes protected members of derived class. Private members of base class are not inherited in derived class.

3.protected visibility:

when a base class is inherited in derived class in protected mode, protected and public members of base class becomes protected members of derived class. Private members of base class are not inherited in derived class.

Example:

```
class base
{
    public:
        int x;
    protected:
        int y;
    private:
        int z;
};

class publicDerived: public base
{
    // x is public
    // y is protected
    // z is not accessible from publicDerived
};

class protectedDerived: protected base
{
    // x is protected
    // y is protected
    // z is not accessible from protectedDerived
};

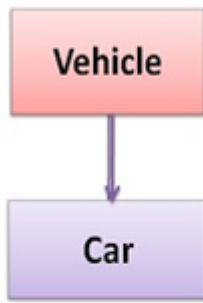
class privateDerived: private base
{
    // x is private
    // y is private
    // z is not accessible from privateDerived
}
```

Q4.state different types of inheritance with diagram.

Ans:

1. Single inheritance:

- In this case **only one class** is derived from another class.



Single Inheritance

2. **Multilevel Inheritance:**

You can derive a class from the base class but you can also derive a class from the derived class.

This form of inheritance is known as multilevel inheritance.

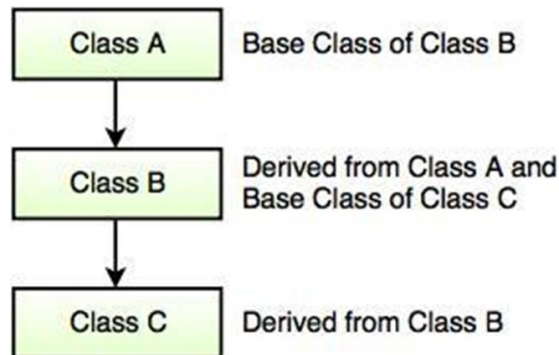


Fig. Multilevel Inheritance

3. **Multiple inheritance:**

When a class is derived from two or more base classes, such inheritance is called **Multiple Inheritance**.

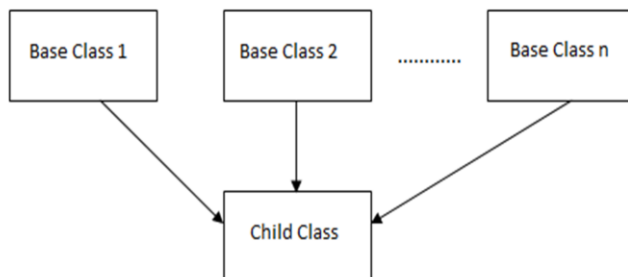
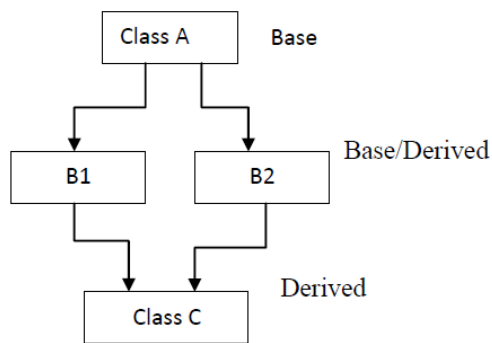


Fig. Multiple Inheritance

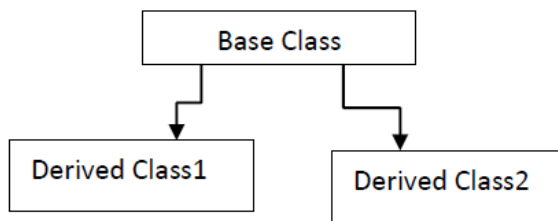
4. **Hybrid Inheritance:**

In this inheritance, it combines single inheritance, multiple inheritance, multi – level inheritance & hierarchical inheritance.



5.Hierarchical inheritance:

In this inheritance, multiple classes can be derived from one single base class. All derived classes inherit properties of single base class.



Q5.state general form of defining derived class.

Ans:

Syntax:

```
class derived-class-name : visibility-mode base-class-name  
{
```

.....

.....

```
};
```

Example :

```
class A
```

```
{
```

.....

.....

```
};
```

```
class B:public A
```

```
{
```

.....

.....

```
};
```

Q6.Define derived class give one example

Ans:

Derived class:

Class derived from another class is known as derived class.

Example:

```
class A
{
.....
.....
};
class B:public A
{
.....
.....
};
```

Here class B is derived class. Which is inheriting properties of class A.

Q7.how protected access specifier is different from private.

Ans:

- 1)The members declared as "protected" cannot be accessed from outside the class, but can be accessed from a derived class. This is used when inheritance is applied to the members of a class.
- 2) The members declared as "private" can be accessed only within the same class and not from outside the class.
- 3) A member declared as a protected is accessible by the member functions within its class and any class immediately derived from it. A member declared as a private is accessible within class only;
- 4) When a protected member is inherited in public mode, it becomes protected in the derived class and therefore it is accessible by the member functions of the derived class.
- 5) If a private member is inherited in public mode, it becomes private in the derived class and therefore it is not accessible by the member functions of the derived class.

Q8.write a program to add two numbers using single inheritance such that the base class function must accept the two numbers from user and derived class function must add these numbers display the sum.

Ans:

```
class A
{
protected:
int a,b;
public:
void accept();
```

```
};  
void A::accept()  
{  
cout<<"enter the two numbers";  
cin>>a>>b;  
}  
class B:public A  
{  
private:  
int sum;  
public:  
void add();  
void display();  
};  
void B::add()  
{  
sum=a+b;  
}  
void B::display()  
{  
cout<<"addition ="<<sum;  
}  
void main()  
{  
clrscr();  
B b1;  
b1.accept();  
b1.add();  
b1.display();  
getch();  
}
```

Q9.write a program to show use of single inheritance.

Ans:

```
class A  
{  
protected:  
int a,b;  
public:  
void accept();  
};
```

```
void A::accept()
{
    cout<<"enter the two numbers";
    cin>>a>>b;
}
class B:public A
{
private:
    int sum;
public:
    void add();
    void display();
};
void B::add()
{
    sum=a+b;
}
void B::display()
{
    cout<<"addition ="<<sum;
}
void main()
{
    clrscr();
    B b1;
    b1.accept();
    b1.add();
    b1.display();
    getch();
}
```

**Q9. Write a program to implement single inheritance from following data.
Accept and display data for one object.**

Data ;
Base class_name = Furniture
Data_mem = material , price
Derived class_name = Table
Data-mem = height, surface-area

Ans:

```
#include<iostream.h>
#include<conio.h>
class furniture
```

```
{
protected:
char material[20];
int price;
};
class table:public furniture
{
int height,surface_area;
public:
void accept();
void display();
};
void table::accept()
{
cout<<"enter maetial type,price,height and surface area";
cin>>material>>price>>height>>surface_area;
}
void table::display()
{
cout<<material<<price<<height<<surface_area;
}
void main()
{
clrscr();
table t1;
t1.accept();
t1.display();
getch();
}
```

Q10. Write a program to implement single inheritance.Declare base class 'Employee' with emp_no and emp_name.Declare derived class 'Fitness' with height and weight.accept and display data for one object.

Ans:

```
#include<iostream.h>
#include<conio.h>
class employee
{
protected:
char emp_name[20];
int emp_no;
};
class fitness:public employee
```



```
{
int height,weight;
public:
void accept();
void display();
};
void fitness::accept()
{
cout<<"enter emp no and emp name height and weight";
cin>>emp_no>>emp_name>>height>>weight;
}
void fitness::display()
{
cout<<emp_no<<emp_name<<height<<weight;
}
void main()
{
clrscr();
fitness f1;
f1.accept();
f1.display();
getch();
}
```

Q11. Write a program to demonstrate constructor in derived class with respect to order of calling constructor and passing parameters to base class constructor.

Ans:

- **Base class** constructors are always called in the **derived class** constructors.
- Whenever you create **derived class object**, first the base class default constructor is executed and then the derived class's constructor finishes execution.

```
class A
{
protected:
int a,b;
public:
A()
{
cout<<"enter two numbers";
cin>>a>>b;
}
};
class B:public A
```

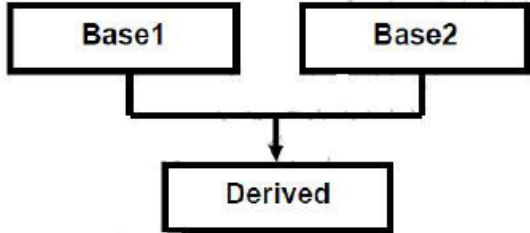
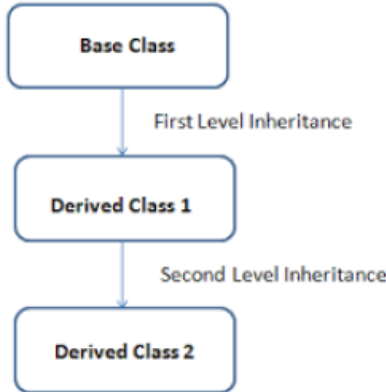
```
{
protected:
int sum;
public:
void add();
void display();
};
void B::add()
{
sum=a+b;
}
void B::display()
{
cout<<"addition ="<<sum;
}
void main()
{
clrscr();
B b1;
b1.add();
b1.display();
getch();
}
```

In above example when derived class object get created by default base class constructor get executed and values get initialized.

Q12. Differentiate between multiple inheritance and multilevel inheritance.

Ans:

Ch3:Inheritance

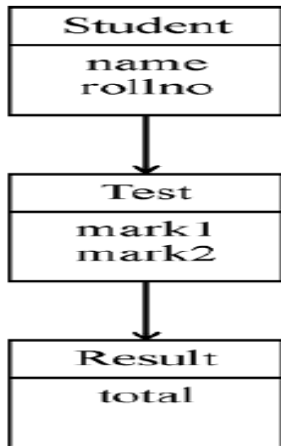
Multiple inheritance:	Multi-level
Multiple inheritance refer to a class being derived from two or more classes.	Multilevel inheritance refers to a class inheriting from a parent class which is itself derived from another class.
A Class derived from at least more than one base class.	A class Extends or Derived from exactly one class & derived class can act as base class for another class
Multiple inheritances are supported by C++, but not by Java and C#.	Multilevel inheritance is supported by all OOPs languages.
Syntax in C++. class base1 { }; class base2 { }; class derived : public base1, public base2 { }; In this example, class derived is derived from two base classes base1 & base2	Syntax in C++. class A { }; class B : public A { }; class C : public B { }; In this example, class B is derived from class A and class C is derived from derived class B.
	

Q13.write a program to calculate the percentage of a student using multilevel inheritance. The base class function will accept the marks in three subjects from user. A class will be derived from above mentioned class that will have a function to find total marks obtained and another class derived from this will have function to calculate and display the percentage scored.

Ans:

Q14.identify the type of inheritance shown in fig implement it by using suitable member function.

Ch3:Inheritance



Ans: above diagram is of type multilevel inheritance.

```
#include<iostream.h>
```

```
#include<conio.h>
```

```
class student
```

```
{
```

```
protected:
```

```
char name[20];
```

```
int roll_no;
```

```
};
```

```
class test:public student
```

```
{
```

```
protected:
```

```
int sub1,sub2;
```

```
};
```

```
class result:public test
```

```
{
```

```
int total;
```

```
public:
```

```
void accept();
```

```
void cal_total();
```

```
};
```

```
void result::accept()
```

```
{
```

```
cout<<"enter student roll no, name sub1 and sub2 marks";
```

```
cin>>roll_no>>name>>sub1>>sub2;
```

```
}
```

```
void result::cal_total()
```

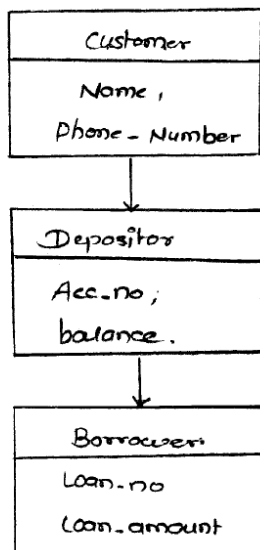
```
{
```

```
total=sub1+sub2;
```

```
cout<<"total="<<total;
```

```
}  
void main()  
{  
  
clrscr();  
result r1;  
r1.accept();  
r1.cal_total();  
getch();  
}
```

Q15. Q21. Identify the type of inheritance and implement it by writing a program for the following Figure. Assume suitable member functions.



Ans:

Q16. Describe multiple inheritance with suitable example.

Ans:

- When a class is derived from two or more base classes, such inheritance is called **Multiple Inheritance**.
- It allow us to combine the features of several existing classes into a single class.
- Example:
- A child has character of both his/her father and mother, etc

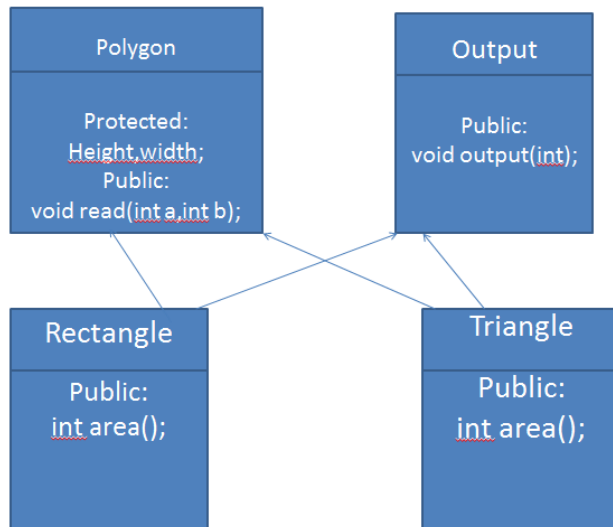
Example:

```
class A  
{  
protected:  
int a;  
public:
```

```
void accept_A();
};
void A::accept_A()
{
cout<<"enter value of a";
cin>>a;
}
class B
{
protected:
int b;
public:
void accept_B();
};
void B::accept_B()
{
cout<<"enter value of b";
cin>>b;
}
class C:public A,public B
{
int mul;
public:
void multiplication();
};
void C::multiplication()
{
mul=a*b;
cout<<"multiplication of two number="<<mul;
}
void main()
{
clrscr();
C c1;
c1.accept_A();
c1.accept_B();
c1.multiplication();
getch();
}
```

Q17. Write a program to define the following relationship using multiple inheritance.

Ch3:Inheritance



Ans:

```
class polygon
```

```
{
```

```
protected:
```

```
int height,width;
```

```
public:
```

```
void read(int a,int b);
```

```
};
```

```
void polygon::read(int a,int b)
```

```
{
```

```
height=a;
```

```
width=b;
```

```
}
```

```
class Output
```

```
{
```

```
public:
```

```
void output(int o);
```

```
};
```

```
void Output::output(int o)
```

```
{
```

```
cout<<"area is="<<o;
```

```
}
```

```
class Rectangle:public polygon,public Output
```

```
{
```

```
public:
```

```
int area();
```

```
};
```

```
int Rectangle::area()
```

```
{
```

```
return(height*width);
}
class Triangle:public polygon,public Output
{
public:
int area();
};
int Triangle::area()
{
return(height*width/2);
}
void main()
{
clrscr();
int height,width,a;
cout<<"enter height and width";
cin>>height>>width;
Rectangle r;
r.read(height,width);
a=r.area();
r.output(a);
Triangle t;
t.read(height,width);
a=t.area();
t.output(a);
getch();
}
```

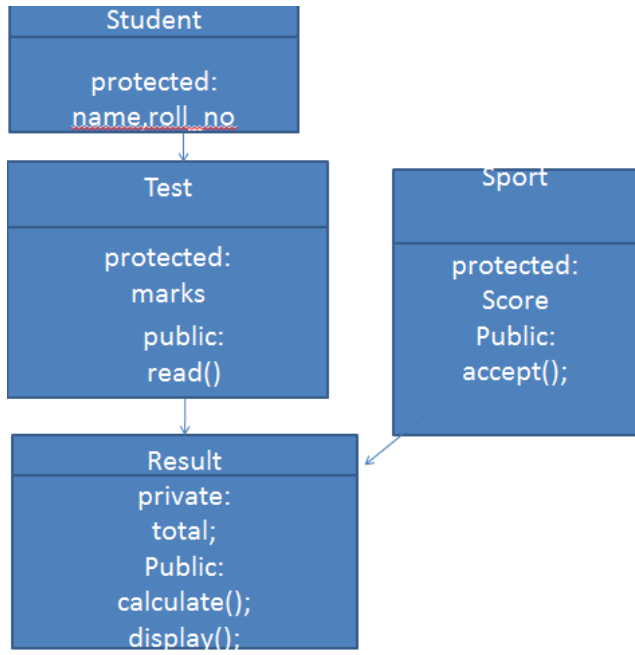
Q18.what is hybrid inheritance? Give one example.

Ans:

In this inheritance, it combines single inheritance, multiple inheritance, multi – level inheritance & hierarchical inheritance.

Example:

Ch3:Inheritance



class Student

```
{
protected:
char name[20];
int rollno;
};
```

class Test:public Student

```
{
protected:
int marks;
public:
void read();
};
void Test::read()
{
cout<<"enter roll no,name and marks";
cin>>rollno>>name>>marks;
}
```

class Sport

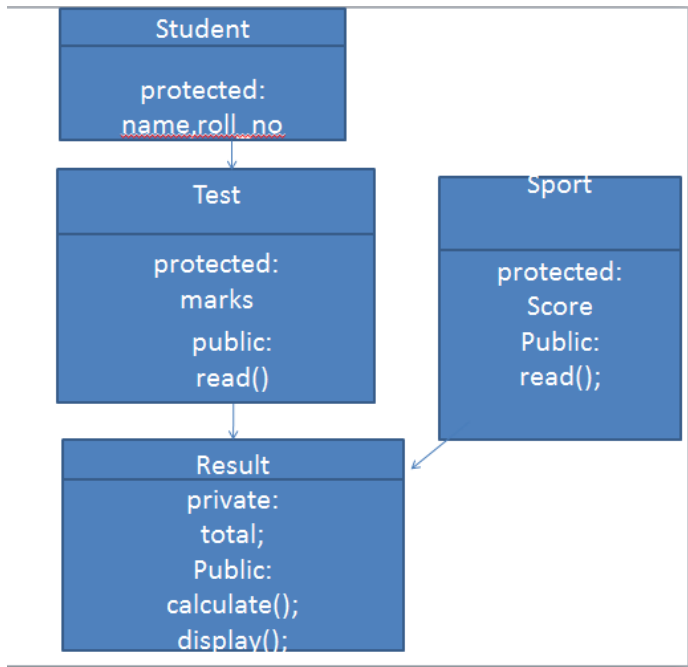
```
{
protected:
int score;
public:
void accept();
};
void Sport::accept()
```

Ch3:Inheritance

```
{
cout<<"1: student has won in national sport event"<<"\n2:student has not won in any
national sport"<<"\n enter score";
cin>>score;
}
class Result:public Sport,public Test
{
private:
int total;
public:
void calculate();
void display();
};
void Result::calculate()
{
if(score==1)
{
total=marks+15;
}
else
{
total=marks;
}
}
void Result::display()
{
cout<<"total marks="<<total;
}
void main()
{
clrscr();
Result r;
r.read();
r.accept();
r.calculate();
r.display();
getch();
}
```

Q19.write a program to define following relationship using hybrid inheritance.

Ch3:Inheritance



Ans:

class Student

```
{
protected:
char name[20];
int rollno;
};
```

class Test:public Student

```
{
protected:
int marks;
public:
void read();
};
void Test::read()
{
cout<<"enter roll no,name and marks";
cin>>rollno>>name>>marks;
}
```

class Sport

```
{
protected:
int score;
public:
void read();
};
```

```
void Sport::read()
{
cout<<"1: student has won in national sport event"<<"\n2:student has not won in any
national sport"<<"\n enter score";
cin>>score;
}
class Result:public Sport,public Test
{
private:
int total;
public:
void calculate();
void display();
};
void Result::calculate()
{
if(score==1)
{
total=marks+15;
}
else
{
total=marks;
}
}
void Result::display()
{
cout<<"total marks="<<total;
}
void main()
{
clrscr();
Result r;
r.Test::read();
r.Sport::read();
r.calculate();
r.display();
getch();
}
```

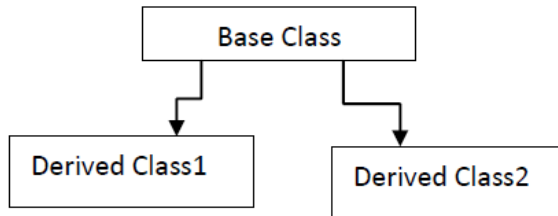
Q20.how hierarchical inheritance is achieved, explain with example.

Ans:

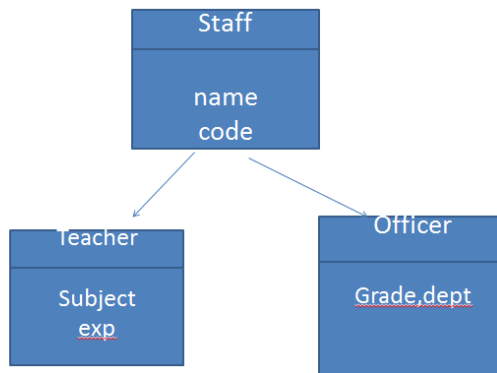
- One base class with many derived classes is called Hierarchical Inheritance.

Ch3:Inheritance

- **Multiple classes are derived from a class** and further more classes are derived from these derived classes.



- Example:



class staff

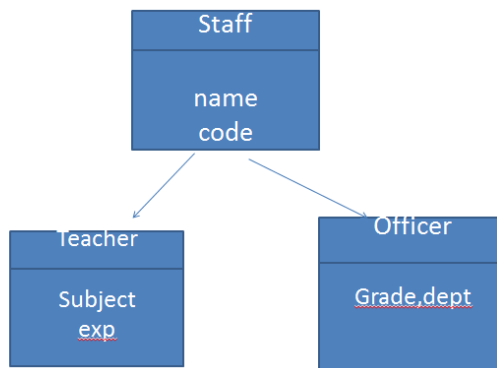
```
{
protected:
char name[20];
int code;
};
class Teacher:public staff
{
char subject[20];
int exp;
public:
void read();
void display();
};
void Teacher::read()
{
cout<<"enter name code,subject and experience of teacher"<<"\n";
cin>>name>>code>>subject>>exp;
}
void Teacher::display()
{
cout<<name<<"\n"<<code<<"\n"<<subject<<"\n"<<exp<<"\n";
```

```
}  
class Officer:public staff  
{  
char dept[20];  
int grade;  
public:  
void read();  
void display();  
};  
void Officer::read()  
{  
cout<<"enter name code,department and grade of officer ";  
cin>>name>>code>>dept>>grade;  
}  
void Officer::display()  
{  
cout<<name<<"\n"<<code<<"\n"<<dept<<"\n"<<grade<<"\n";  
}  
void main()  
{  
clrscr();
```

```
Teacher t;  
t.read();  
t.display();
```

```
Officer o;  
o.read();  
o.display();  
getch();  
}
```

Q21.write a program to implement inheritance as shown in fig. assumes suitable member function.



```
class staff
{
protected:
char name[20];
int code;
};
class Teacher:public staff
{
char subject[20];
int exp;
public:
void read();
void display();
};
void Teacher::read()
{
cout<<"enter name code,subject and experience of teacher"<<"\n";
cin>>name>>code>>subject>>exp;
}
void Teacher::display()
{
cout<<name<<"\n"<<code<<"\n"<<subject<<"\n"<<exp<<"\n";
}
class Officer:public staff
{
char dept[20];
int grade;
public:
void read();
void display();
};
void Officer::read()
{
cout<<"enter name code,department and grade of officer ";
cin>>name>>code>>dept>>grade;
}
void Officer::display()
{
cout<<name<<"\n"<<code<<"\n"<<dept<<"\n"<<grade<<"\n";
}
void main()
{
```

```
clrscr();  
Teacher t;  
t.read();  
t.display();  
Officer o;  
o.read();  
o.display();  
getch();  
}
```