Lab-7

Title: Object Inheritance and reusability

Objective:

- To be familiar with inheritance and composition
- To understand about how inheritance supports reusability
- To understand about ambiguity in inheritance and virtual base class

Theory:

- **Introduction to inheritance and its types**
- **Ambiguity in inheritance**
- **Virtual base class**
- **Composition**

1. WAP to show that how inheritance supports reusability.

```
#include<iostream>
using namespace std;
class A{
	private:
		int x;
	protected:
		int y;
	public:
		int z;
	void get_XYZ(){
		cout<<"Enter the value of X,Y,Z"<<endl;
		cin>>x>>y>>z;
	}
	void show_XYZ(){
		cout<<"x="<<x<<",y="<<y<<",z="<<z<<endl;
	}
};
class B:public A{
	private:
		int k,sum;
	public:
	void get_k(){
		cout<<"Enter the value of k"<<endl;
		cin>>k;
	}
	void show_k(){
		cout<<"K="<<k<<endl;
	}
	void add(){
		sum=y+z+k;
	}
	void display(){
		cout<<"sum="<<sum;
	}
};
int main(){
	B b1;
	b1.get_XYZ();
	b1.show_XYZ();
	b1.get_k();
	b1.show_k();
	b1.add();
	b1.display();
	return 0;

}
```

2. WAP to enter information of n students and then display is using the concept multiple inheritance.

```
#include<iostream>
using namespace std;
class info{
	protected:
		char name[20];
		int roll;
		char faculty[20];
	public:
		void get_info(){
			cout<<"Enter the name of student"<<endl;
			cin>>name;
			cout<<"Enter the roll number of student"<<endl;
			cin>>roll;
			cout<<"Enter the faculty of student"<<endl;
			cin>>faculty;
			}};
class marks{
	protected:
		int m1,m2,m3;
	public:
	   void mark(){
		cout<<"Enter the obtain marks in 3 subject"<<endl;
		cin>>m1>>m2>>m3;
		}
};
class display:public info,public marks{
	public:
		void show_info(){
			cout<<"Name of student:"<<name<<endl;
			cout<<"Roll number of student:"<<roll<<endl;
			cout<<"Faculty of student:"<<faculty<<endl;
		if(m1>45&&m2>45&&m3>45){
			cout<<"Result is pass"<<endl;
		}
		else{
			cout<<"Result is fail"<<endl;
		}}
};
int main(){
	display d[100];
	int i,n;
	cout<<"Enter the number of student"<<endl;
	cin>>n;
	for(i=0;i<n;i++){
		cout<<"for "<<i+1<<" student"<<endl;
		d[i].get_info();
		d[i].mark();
		d[i].show_info();
	}
	return 0;
}
```

3. WAP to enter information of n students and then display is using the concept multilevel inheritance.

```
#include<iostream>
using namespace std;
class info{
	protected:
		char name[20];
		int roll;
		char faculty[20];
	public:
		void get_info(){
			cout<<"Enter the name of student"<<endl;
			cin>>name;
			cout<<"Enter the roll number of student"<<endl;
			cin>>roll;
			cout<<"Enter the faculty of student"<<endl;
			cin>>faculty;
			}
};
class marks:public info{
	protected:
		int m1,m2,m3;
	public:
	   void mark(){
		cout<<"Enter the obtain marks in 3 subject"<<endl;
		cin>>m1>>m2>>m3;
		}};
class display:public marks{
	public:
		void show_info(){
			cout<<"Name of student:"<<name<<endl;
			cout<<"Roll number of student:"<<roll<<endl;
			cout<<"Faculty of student:"<<faculty<<endl;
		if(m1>45&&m2>45&&m3>45){
			cout<<"Result is pass"<<endl;
		}
		else{
			cout<<"Result is fail"<<endl;
		}}};
int main(){
	display d[100];
	int i,n;
	cout<<"Enter the number of student"<<endl;
	cin>>n;
	for(i=0;i<n;i++){
		cout<<"for "<<i+1<<" student"<<endl;
		d[i].get_info();
		d[i].mark();
		d[i].show_info();
	}
	return 0;
}
```

4. Develop a complete program for an institution which wishes to maintain a database of its staff. Declare a base class Staff which include staff_id and name.Now develop a records for the following staffs with the given information below.
Lecturer(subject,department)
Administrative staff (Post,department)

```
#include<iostream>
using namespace std;
class staff{
	protected:
		int staff_id;
		char sname[20];
	public:
		void get(){
			cout<<"Enter the staff id:"<<endl;
			cin>>staff_id;
			cout<<"Enter the staff name:"<<endl;
			cin>>sname;
			}
		void display1(){
			cout<<"Staff id:"<<staff_id<<endl;
			cout<<"Staff name:"<<sname<<endl;
		}
};
class lecturer:public staff{
	protected:
		char subject[20];
		char department[20];
	public:
		void get_lecture(){
			cout<<"Enter the subject:"<<endl;
			cin>>subject;
			cout<<"Enter the department:"<<endl;
			cin>>department;
		}
		void displaylecturer(){
		  cout<<"Subject:"<<subject<<endl;
		  cout<<"Department:"<<department<<endl;
		  }
};
class administrative:public staff{
	protected:
		char post[20];
		char department[20];
	public:
		void getadministrative(){
			cout<<"Enter the post:"<<endl;
			cin>>post;
			cout<<"Enter the department:"<<endl;
			cin>>department;
```

```
                }
                void displayadministrative(){
                        cout<<"Post:"<<post<<endl;
                        cout<<"Department"<<department<<endl;
                }
};
int main(){
        lecturer l1;
        l1.get();
        l1.get_lecture();
        l1.display1();
        l1.displaylecturer();
        administrative a1;
        a1.get();
        a1.getadministrative();
        a1.display1();
        a1.displayadministrative();
        return 0;
}
```

5. Create a class student with two data members represent name and age. Use appropriate member function to read and print these data members name and roll. Derive a class marks from student that has additional data member sessional1, sessional2 to store sessional marks. Derive another class result from marks and add the sessional marks. Use appropriate member function to read and display data in the class.

```
#include<iostream>
using namespace std;
class student{
	protected:
		char name[20];
		int roll;
	public:
		void getinfo(){
			cout<<"Enter the name of student:"<<endl;
			cin>>name;
			cout<<"Enter the roll of student:"<<endl;
			cin>>roll;
			}
		void displayinfo(){
			cout<<"Student name:"<<name<<endl;
			cout<<"Student Roll:"<<roll<<endl;
		}
};
class marks:public student{
	protected:
		int sessional1;
		int sessional2;

	public:
		void get_mark(){
			cout<<"Enter the mark of sessional1:"<<endl;
			cin>>sessional1;
			cout<<"Enter the marks of sessional2:"<<endl;
			cin>>sessional2;
		}
	   void displaymark(){
		cout<<"Mark of sessional1:"<<sessional1<<endl;
		cout<<"Mark of sessional2:"<<sessional2<<endl;
		}};
class result:public marks{
	public:
		void displayresult(){
			cout<<"Sum of marks"<<sessional1+sessional2;
		}};
int main(){
	result r1;
	r1.getinfo();
	r1.get_mark();
	r1.displayinfo();
	r1.displaymark();
	r1.displayresult();
	return 0;
}
```

6. Implement the below given in class diagram in C++.Assume necessary function yourself.

```
#include<iostream>
using namespace std;
class student{
	protected:
		char name[20];
		int roll;
	public:
		void get_info(){
		cout<<"Enter the name of student:"<<endl;
		cin>>name;
		cout<<"Enter the roll of student:"<<endl;
		cin>>roll;}
		void display_info(){
			cout<<"Name of student"<<name<<endl;
			cout<<"ROll of student"<<roll<<endl;
		}
};
class test:public student{
	protected:
		int math;
		int eng;
	public:
		void get_marks(){
			cout<<"Enter the marks of math"<<endl;
			cin>>math;
			cout<<"Enter the marks of english"<<endl;
			cin>>eng;
		}
		void display_marks(){
			cout<<"Marks in math:"<<math<<endl;
			cout<<"Marks in English:"<<eng<<endl;
		}
	};
class sport{
	protected:
		int score;
	public:
		void get_score(){
			cout<<"Enter the score"<<endl;
			cin>>score;
		}
		void display_score(){
			cout<<"Score is:"<<score<<endl;
		}
};
class result:public test,public sport{
	protected:
		int total;
	public:
		void total_marks(){
			total=math+eng;
			cout<<"Total marks is "<<total;
		}
```

```
};
int main(){
        result r;
   r.get_info();
   r.get_marks();
   r.get_score();
   r.display_info();
   r.display_marks();
   r.display_score();
   r.total_marks();
   return 0;
}
```

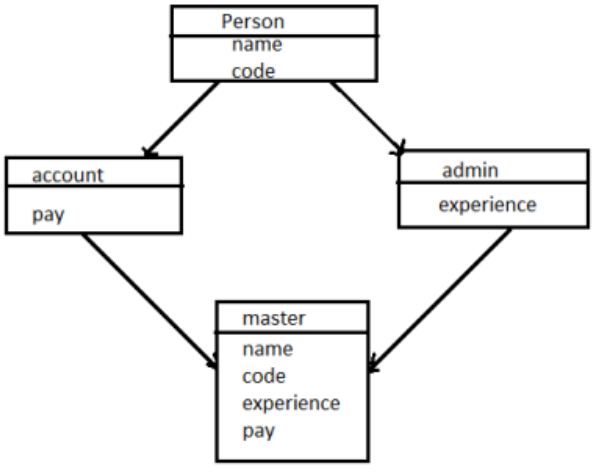
7. An educational institution wishes to maintain a database of its employees. The database is divided into a number of classes whose hierarchical relationship are shown below. The figure also shows minimum information requires for each class. Specify all the classes and define functions to create database and retrieve individual information when required.

```
#include<iostream>
#include<string>
using namespace std;
class Staff {
protected:
   char name[20];
   int code;
public:
   void get_staffinfo() {
      cout<<"Enter the name of staff:";
      cin>>name;
      cout<<"Enter the code of staff:";
      cin>>code;
   }
   void show_staffinfo() {
      cout<<"Name of the staff:"<<name<<endl;
      cout<<"Code of the staff:"<<code<<endl;
   }
};

class Teacher:public Staff {
   char sub[20];
   char pub[20];
public:
   void get_teacherinfo() {
      get_staffinfo();
      cout<<"Enter subject:";
      cin>>sub;
      cout<<"Enter publication:";
      cin>>pub;
   }
   void show_teacherinfo() {
      show_staffinfo();
      cout<<"Subject:"<<sub<<endl;
      cout<<"Publication:"<<pub<<endl;
   }
};
class Officer:public Staff {
   char grade[20];
public:
   void get_officerinfo() {
      get_staffinfo();
      cout<<"Enter grade:";
      cin>>grade;
   }
   void show_officerinfo() {
      show_staffinfo();
      cout<<"Grade:"<<grade<<endl;
```

```
    }
};
class Typist :public Staff {
protected:
    float speed;
public:
    void get_typistinfo() {
        get_staffinfo();
        cout<<"Enter typing speed:";
        cin>>speed;
    }
    void show_typistinfo() {
        show_staffinfo();
        cout<<"Typing Speed:"<<speed<<"wpm"<<endl;
    }
};

class Regular :public Typist {
    float salary;
public:
    void get_regularinfo() {
        get_typistinfo();
        cout<<"Enter salary:";
        cin>>salary;
    }
    void show_regularinfo() {
        show_typistinfo();
        cout<<"Salary:"<<salary<<endl;
    }
};
class Casual  :public Typist {
    float dailyWages;
public:
    void get_casualinfo() {
        get_typistinfo();
        cout<<"Enter daily wages:";
        cin>>dailyWages;
    }
    void show_casualinfo() {
        show_typistinfo();
        cout<<"Daily Wages:"<<dailyWages<<endl;
    }
};
int main() {
    Teacher t;
    Officer o;
    Regular r;
    Casual c;
    t.get_teacherinfo();
    o.get_officerinfo();
    r.get_regularinfo();
    c.get_casualinfo();
    t.show_teacherinfo();
    o.show_officerinfo();
```

```
    r.show_regularinfo();
    c.show_casualinfo();
            return 0;
}
```

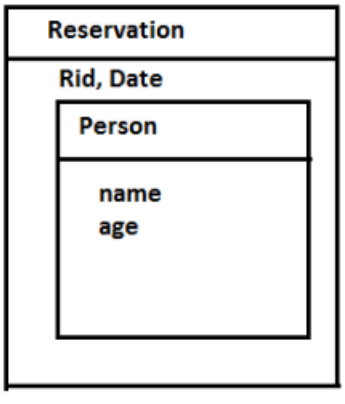
9. The following figure shows minimum information required for each class.
Write a Program to realize the above program with necessary member functions to create the database and retrieve individual information




```
#include <iostream>
#include <string.h>
using namespace std;
class Person {
protected:
   char name[20];
   int code;
public:
   void get_person(){
            cout<<"Enter the name of person"<<endl;
            cin>>name;
            cout<<"Enter the code"<<endl;
            cin>>code;
            }
   void show_person(){
      cout<<"Name:"<<name<<"\nCode:"<<code<<"\n";
   }
};
class Account : virtual public Person {
protected:
   float pay;
public:
   void get_account(){
            cout<<"Enter pay"<<endl;
            cin>>pay;
            }
            void show_account(){
                     cout<<"Pay:"<<pay<<endl;
            }
};
```

```
class Admin : virtual public Person {
protected:
   int experience;
public:
   void get_admin(){
          cout<<"Enter the experience"<<endl;
          cin>>experience;
          }
          void show_admin(){
           cout<<"Experience:"<<experience<<"years"<<endl;
           }
};
class Master : public Account, public Admin {
public:
          void display(){
                    show_person();
                    show_account();
                    show_admin();
          }
};

int main() {
   Master m1;
   m1.Person::get_person();
   m1.Account::get_account();
   m1.Admin::get_admin();
          cout<<"Stored Record";
   m1.display();
   return 0;
}
```

10. Write a program that allow you to book a ticket for person and use two classes Person, Reservation. Class Reservation is composite class/ container class.




```
#include <iostream>
#include <string.h>
using namespace std;
class Person{
          protected:
                    char name[20];
```

```
		int age;
	public:
		void get_person(){
			cout<<"Enter your name"<<endl;
			cin>>name;
			cout<<"Enter your age"<<endl;
			cin>>age;
		}
		void show_person(){

			cout<<"Name:"<<name<<endl;
			cout<<"Age:"<<age<<endl;
		}
	};
class Reservation{
	protected:
		int rid;
		char date[20];
		Person p;
	public:
		void get_reservation(){
			p.get_person();
			cout<<"Enter the reservation id"<<endl;
			cin>>rid;
			cout<<"Enter the reservation date (e.g 2025-04-22)"<<endl;
			cin>>date;
		}
		void show_reservation(){
			p.show_person();
			cout<<"Reservation id:"<<rid<<endl;
			cout<<"Reservation date:"<<date<<endl;
		}
};
int main(){
	Reservation r;
	r.get_reservation();
	r.show_reservation();
	return 0;
}
```

11.Write a program to input two vector coordinates from the base class named “Base”. Class “Derived” inherits all the properties of class “Base”. Class “Derived” must contain a function named add_vector() that add the two vectors input from the base class and finally display the result from the function display() that is friend to the base class.

```
#include <iostream>
using namespace std;

class Base {
protected:
   int x1, y1;
   int x2, y2;
   int rx, ry;
public:
```

```
    void input_vectors() {
        cout << "Enter coordinates of first vector (x1 y1): ";
        cin >> x1 >> y1;
        cout << "Enter coordinates of second vector (x2 y2): ";
        cin >> x2 >> y2;
    }

    friend void display(const Base& b);
};

class Derived : public Base {
public:
    void add_vector() {
        rx = x1 + x2;
        ry = y1 + y2;
    }
};
void display(const Base& b) {
    cout << "Resultant vector: (" << b.rx << ", " << b.ry << ")" << endl;
}
int main() {
    Derived d;
    d.input_vectors();
    d.add_vector();
    display(d);
    return 0;
}
```