LAB 5

1. /*Create a person class with attributes id and name and member function to set the member variables and to display those variables. Then create class student with member variables grade and addresses which inherits person class publically.*/

```
#include<iostream>
using namespace std;
class person
{
	private:
		int id;
		char name[20];

	public:
		void getdata()
		{
			cout<<"Enter the name of the person : ";
			cin>>name;
			cout<<"Enter the id of the person : ";
			cin>>id;
		}
		void readdata()
		{
			cout<<endl<<"Name : "<<name;
			cout<<endl<<"Id : "<<id;
		}
```

```
};
class student: public person
{
	private:
		char grade , address[20];

	public:
		void input()
		{
			getdata();
			cout<<"Enter the grade of the person : ";
			cin>>grade;
			cout<<"Enter the address of the person : ";
			cin>>address;
		}
		void showdata()
		{
			readdata();
			cout<<endl<<"Grade : "<<grade;
			cout<<endl<<"Address : "<<address;
		}
};
int main()
{
	student s;
```

```
        s.input();
        cout<<endl<<"----------Personal Details----------";
        s.showdata();
}
```

2. /*Create class polygon based on this class, derive two base classes Rectangle and Triangle to calculate area with necessary data members and member function. (note: area of rectangle=L*b and area of triangle= (l*b/2)).*/

```
#include<iostream>
using namespace std;
class polygon
{
        public:
                int l , b;
                void getdata()
                {
                        cout<<"Enter the length and breadth of the polygon : "<<endl;
                        cin>>l>>b;
                }
};

class rectangle: private polygon
{
        public:
                void areaofrect()
```

```
        {
            getdata();
            cout<<endl<<"The area of rectangle is : "<<l*b;
        }
};

class triangle: private polygon
{
    public:
        void areatri()
        {
            getdata();
            cout<<endl<<"The area of triangle is : "<<(l*b)/2;
        }
};
int main()
{
    cout<<"For Rectangle"<<endl;
    rectangle r1;
    r1.areaofrect();
    cout<<endl<<"-----------------------------------"<<endl;
    cout<<"For Triangle"<<endl;
    triangle t1;
    t1.areatri();
}
```

3. /*Write a program to perform basic mathematical (+, -, / , %) operation using the concept of Hierarchical inheritance*/

```
#include<iostream>
using namespace std;
class operation
{

	public:
		int a,b;

		void getop()
		{
			cout<<"Enter the operands : "<<endl;
			cin>>a>>b;
		}
};

class add : public operation
{

	public:
		void sum()
		{
			cout<<"For addition"<<endl;
			getop();
```

```
                cout<<"The sum of the operands is "<<a+b;
        }
};
class sub : public operation
{

    public:
        void diff()
        {
            cout<<"For subtraction"<<endl;
            getop();

            cout<<"The difference of the operands is "<<a-b;
        }
};

class divide : public operation
{

    public:
        void div()
        {
            cout<<"For division"<<endl;
            getop();

            cout<<"The division of the operands is "<<a/b;
```

```
        }
};
class modulo : public operation
{

    public:
        void mod()
        {
            cout<<"For remainder"<<endl;
            getop();

            cout<<"The remainder of the operands is "<<a%b;
        }
};
int main()
{
    add a1;
    a1.sum();
    cout<<endl<<endl;

    sub s1;
    s1.diff();
    cout<<endl<<endl;
    divide d1;
    d1.div();
    cout<<endl<<endl;
```

```
        modulo m1;
        m1.mod();
        cout<<endl<<endl;
}
```

4. /*Write a to illustrate the use of constructors in multiple inheritance*/

```
#include<iostream>
using namespace std;
class A
{
        protected:
                int x;

        public:
                A(int a)
                {
                        x=a;
                }
};
class B
{
        protected:
                int y;
```

```
        public:
                B(int b)
                {
                        y=b;
                }
};

class C: public A, public B
{
        int z;

        public:
                C(int a , int b , int c):A(a), B(b)
                {
                        z=c;
                }

                void show()
                {
                        cout<<"X = "<<x<<endl;
                        cout<<"Y = "<<y<<endl;
                        cout<<"Z = "<<z<<endl;
                }
};
int main()
{
```

```
        C c1(1,2,3);


        c1.show();
}
```

5. /*Write a to illustrate the use of constructors in multilevel inheritance*/

```
#include<iostream>
using namespace std;
class A
{
        protected:
                int ad;
        public:
                A(int a)
                {
                        ad=a;
                }
};
class B : public A
{
        protected:
                int bd;
        public:
                B(int b , int a):A(a)
                {
```

```
                bd=b;
        }
};
class C : public B
{
    private:
        int cd;
    public:
        C(int a , int b , int c):B(b,a)
        {
            cd=c;
        }
        void show()
        {
            cout<<"ad = "<<ad<<endl;
            cout<<"bd = "<<bd<<endl;
            cout<<"cd = "<<cd<<endl;
        }
};
int main()
{
    C c(1,2,3);

    c.show();
}
```

6. /*Write a program to illustrate use of destructors in multiple inheritance*/

```
#include<iostream>
using namespace std;
class A
{
	public:
		~A()
		{
			cout<<"Class A Destructor"<<endl;
		}
};
class B
{
	public:
		~B()
		{
			cout<<"Class B Destructor"<<endl;
		}
};

class C : public A , public B
{
	public:
		~C()
		{
```

```
                cout<<"Class C Destructor"<<endl;
        }
};
int main()
{
    C x;
}
```

7. /*Write a program to illustrate use of destructors in multilevel inheritance*/

```
#include<iostream>
using namespace std;
class A
{
    public:
        ~A()
        {
            cout<<"Class A Destructor"<<endl;
        }
};
class B : public A
{
    public:
        ~B()
        {
            cout<<"Class B Destructor"<<endl;
```

```
        }
};
class C : public B
{
    public:
        ~C()
        {
            cout<<"Class C Destructor"<<endl;
        }
};
int main()
{
    C x;
}
```

8. /*Write a program to illustrate the concept of aggregation.*/

```
#include<iostream>
using namespace std;
class employee
{
    private:
        int eid;

    public:
        void getdata()
```

```
        {
            cout<<"Enter eid of employee : ";
            cin>>eid;
        }
        void display()
        {
            cout<<"Employee id : "<<eid<<endl;
        }
};
class company
{
    private:
        char name[20];
        employee e;
    public:
        void input()
        {
            e.getdata();
            cout<<"Enter the name of company : ";
            cin>>name;
        }
        void display()
        {
            e.display();
            cout<<"Company Name : "<<name<<endl;
        }
```

```
};
int main()
{
	company c;

	c.input();
	cout<<"-----------------------------------"<<endl;
	c.display();
}
```