1-

/* Write C++ programs that illustrate a)Single inheritance b) Multiple inheritance c)Multi level inheritance d)Hierarchical inheritance */

```
//a- Single inheritance
# include <iostream>
 using namespace std;
 class base
 {
   public:
   base()
   {
       cout<<"Base class called.\n";
   }
};
Class derived: public base
 {
   public:
     derived()
     {
        cout<<"Derived class called.";
     }
 };
int main()
{
   derived obj;
}
```

<u>output</u>:

```
Base class called.
Derived class called.
```

```
/* b. Multiple inheritance */
# include< iostream>
 using namespace std;
 Class Base 1
 {
   public:
     Base1()
     {
        cout<<"Base1 called.\n";
     }
};
Class  Base 2
{
  public:
     Base2()
      {
       cout<<"Base2 called.\n";
      }
};
class Derived: public Base1, public Base 2
{
 public:
   Derived()
   {
      cout<<"Derived called.\n";
   }
};
int main()
{  Derived obj;
   return(0);
}
```

output:

```
Base1   called
Base2   called
Derived called
```

```
/* c. Multi level inheritance */
#include<iostream>
 using namespace std;
 class Students
{
  protected:
  int roll;
  Char name[20];
  public:
  Void get()
  {
    cout<<"Enter name of student:";
    cin>>name;
    cout<<"Enter roll:";
    cin>>roll;
  }
};
Class marks:public Students
{
  protected:
   int m1,m2;
  public:
   void getmarks()
   {
     cout<<"Enter mark of English:";
     cin>>m1;
     cout<<"Enter mark of Math:";
     cin>>m2;
   }
};
class result: public marks
{
  protected:
    int t;
     Public:
     void total()
     {
       t=m1+m2;
     }
     void show()
     {
       cout<<"\nName="<<name<<"\nRoll No.="<<roll;
       cout<<"\nTotal marks ="<<t;
     }
};
```

```
int main()
{
  result obj;
  obj.get();
  Obj.getmarks();
  obj.total();
  obj.show();
return(0);
}
```

Output:

```
Enter name of student : Arun
Enter roll:5
Enter mark of English:66
Enter mark of Math ;44

Name=Arun
Roll No. =5
Total marks=110
```

```
/* d. Hierarchical inheritance */
# inclue<iostream>
using namespace std;
class base
{
  protected:
  int a=3, b=4;
};
class derived 1: public base
{
  protected:
    int x:
    public:
      void add()
      {
        x=a+b;
        cout<< "a+b= "<< x;
      }
};
class derived 2: public base
{
```

```
public:
  void sub( )
  {
     int y=a-b;
     Cout<<"\na-b="<<y;
  }
};
int main()
{
   derived1 obj1;
   Obj1-add();

   derived2 obj2;
   obj2-sub();
   return(0);
}
```

Output:

```
a+b=7
a-b=1
```

```
/* WAP to illustrate a diamond problem with its solution:*/
# include <iostream>
using namespace std;
class base
{
  public:
    Void show( )
      {
        cout<<"Base class called.";
      }
};
class derived 1: virtual public base
 { };
Class derived 2: virtual public base
 { };
 class derived 3: public derived1, public derived 2
 { };
 int main()
 {
    derived3 obj;
    obj.show();
 }
```

Output:

```
Base class called.
```