#include<iostream>

using namespace std;

class ClassA

{

public:

void aClass()

{

cout<<"Iam a ClassA\n";

}

};

class ClassB

{

public:

void bClass()

{

cout<<"Iam a ClassB\n";

}

};

//Multiple Inheritance

class ClassC: public ClassA,public ClassB

{

public:

void cClass()

{

cout<<"Iam a ClassC\n";

}

};

//Simple Inheritance

class ClassD: public ClassA

{

public:

void dClass()

{

cout<<"Iam a ClassD\n";

}

};

//Hybrid Inheritance

class ClassE: public ClassC,public ClassD

{

public:

void eClass()

{

cout<<"Iam a ClassE\n";

}

};

//Multi-Level Inheritance

class ClassF: public ClassD

{

public:

void fClass()

{

cout<<"Iam a ClassF\n";

}

};

//Hierarchical Inheritance

class ClassG:public ClassA

{

public:

void gClass()

{

cout<<"Iam a ClassG\n";

}

};

class ClassH:public ClassA

{

public:

void hClass()

{

cout<<"Iam a ClassH\n";

}

};

int main()

{

//Simple Inheritance

cout<<"Simple Inheritance\n";

ClassD obj1;

obj1.aClass();

obj1.dClass();

//Multiple Inheritance

cout<<"Multiple Inheritance\n";

ClassC obj2;

obj2.aClass();

obj2.bClass();

obj2.cClass();

//Multi-Level Inheritance

cout<<"Multi-Level Inheritance\n";

ClassF obj3;

obj3.aClass();

obj3.dClass();

obj3.fClass();

//Hierarchical Inheritance

cout<<"Hierarchical Inheritance\n";

ClassG obj4;

ClassH obj5;

obj4.aClass();

obj4.gClass();

obj5.aClass();

obj5.hClass();

//Hybrid Inheritance

cout<<"Hybrid Inheritance\n";

ClassE obj6;

//obj6.aClass();

//aClass() method becomes ambiguous here

obj6.bClass();

obj6.cClass();

obj6.dClass();

obj6.eClass();

return 0;

}