

# C++ OOP

## ครั้งที่ 3

# หัวข้อ

---

1. การสืบทอดคุณสมบัติ (Inheritance)
2. Inheritance และ Class diagram
3. รูปแบบของการสืบทอด (Type of Inheritance)
  1. Single Inheritance
  2. Multiple Inheritance
  3. Multilevel Inheritance
  4. Hierarchical Inheritance
  5. Hybrid (Virtual) Inheritance

# แนวคิดของ OOP

---

- กฎหลักของ OOP คือ abstraction ซึ่งประกอบด้วยแนวคิด 3 ประการ
  1. Encapsulation เป็นสร้างวัตถุที่มีทั้งข้อมูลและวิธีการทำงานครบในตัว ของออบเจกต์เอง อีกทั้งยังซ่อนรายละเอียดของแต่ละองค์ประกอบไว้ ภายใน
  2. Inheritance เป็นคุณสมบัติในการเขียนโปรแกรมเชิงวัตถุที่เรียกว่า คุณสมบัติการสืบทอด โดยที่คลาสสามารถสืบทอด attribute และ method จากคลาสหลัก (base class) ไปยังคลาสน้อย (derived class)
  3. Polymorphism การกำหนดให้วัตถุสามารถมีได้หลายรูปแบบตามกรณี เฉพาะต่าง ๆ ซึ่งเกิดจากการสืบทอดจาก super class และมันยังคง รักษาสภาพและคุณสมบัติของ super class

<https://linux.thai.net/~thep/docs/> และ <http://marcuscode.com/lang/java/>

# การสืบทอดคุณสมบัติ (Inheritance)

Inheritance เป็นคุณสมบัติในการเขียนโปรแกรมเชิงวัตถุที่เรียกว่าคุณสมบัติการสืบทอด โดยที่คลาสสามารถสืบทอด attribute และ method จากคลาสหลัก (base class) ไปยังคลาทย่อย (derived class)

- Inheritance เป็นคุณสมบัติสำคัญอย่างหนึ่งของ Object Oriented Programming.
- นิยามคำศัพท์
  - base class เป็นคลาสหลักที่ใช้งานร่วมกัน
  - derived class เป็นคลาสที่เรียกคลาสหลักและเขียนต่อเติมตามกรณีต่าง
- ในตำราอื่นๆ
  - superclass / subclass superclass เป็นชนิดทั่วไป, subclass เป็นชนิดย่อย
  - parent class / child class parent class เป็นชนิดตั้งต้น, child class เป็นชนิดที่เกิดจาก parent

<https://linux.thai.net/~thep/docs/> และ <https://www.geeksforgeeks.org/inheritance-in-c/>

# แนวความคิดของการสืบทอดคุณสมบัติ

## (Inheritance)

---

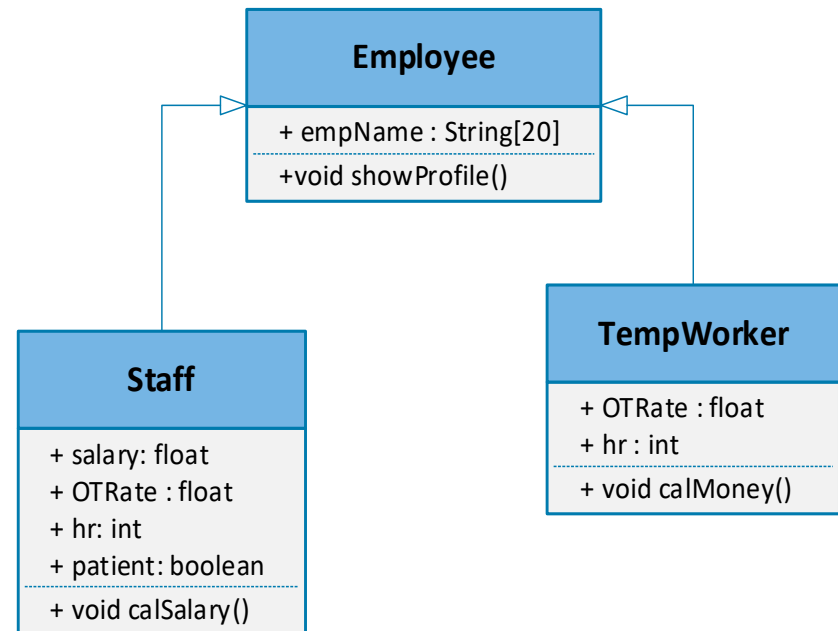
- รถยนต์ สามารถแบ่งประเภทตามการใช้งานได้ 4 ประเภท
  - รถกระบะ มี 2 ที่นั่ง พื้นที่บรรทุกของด้านท้าย
  - รถเก๋ง มี 4 ที่นั่ง ขนาดเล็ก
  - รถบรรทุก มีจำนวนล้อมากกว่า 4 ล้อและมีพื้นที่บรรทุกของ
  - รถดับเพลิง มีขนาดใหญ่ มีถังบรรจุน้ำ
- รถแต่ละประเภทนั้นมีคุณสมบัติเหมือนรถยนต์ แต่จะมีคุณสมบัติพิเศษเป็นของตนเอง

# แนวความคิดของการสืบทอดคุณสมบัติ (Inheritance)

- พนักงานบริษัท (Employee)

สามารถแบ่งประเภทสิทธิ์  
ประโยชน์ในการทำงาน

- พนักงานประจำ (Staff) จะมี  
เงินเดือนประจำตำแหน่ง มีค่า  
OT และสิทธิการรักษาพยาบาล
- พนักงานชั่วคราว (TempWorker)  
จะเงินเดือนที่ได้รับ คือ จำนวน  
ชั่วโมงที่ทำงาน \* อัตราค่าแรง  
และไม่มีสิทธิการรักษาพยาบาล



# รูปแบบของการสืบทอด (Type of Inheritance)

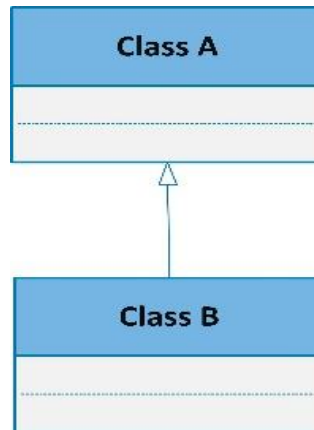
---

1. Single Inheritance
2. Multiple Inheritance
3. Multilevel Inheritance
4. Hierarchical Inheritance
5. Hybrid (Virtual) Inheritance

# รูปแบบของการสืบทอด (Type of Inheritance)

---

1. **Single Inheritance:** หมายถึง คลาสย่อยที่สืบทอดคุณสมบัติจากคลาสหลักจำนวน 1 คลาส โดยมีลักษณะการสืบทอดดังรูปด้านล่าง

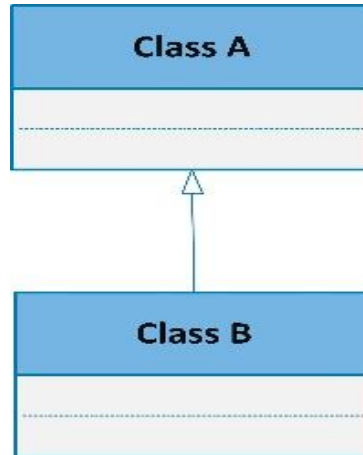


<http://www.trytoprogram.com/cplusplus-programming/single-inheritance/>



# Single Inheritance

---



รูปแบบการเขียน

**class A { ..... }; //เป็นคลาสหลัก**

**class B : access\_specifier A { ..... }; //คลาสน้อยเรียกสืบทอดคลาสหลัก**

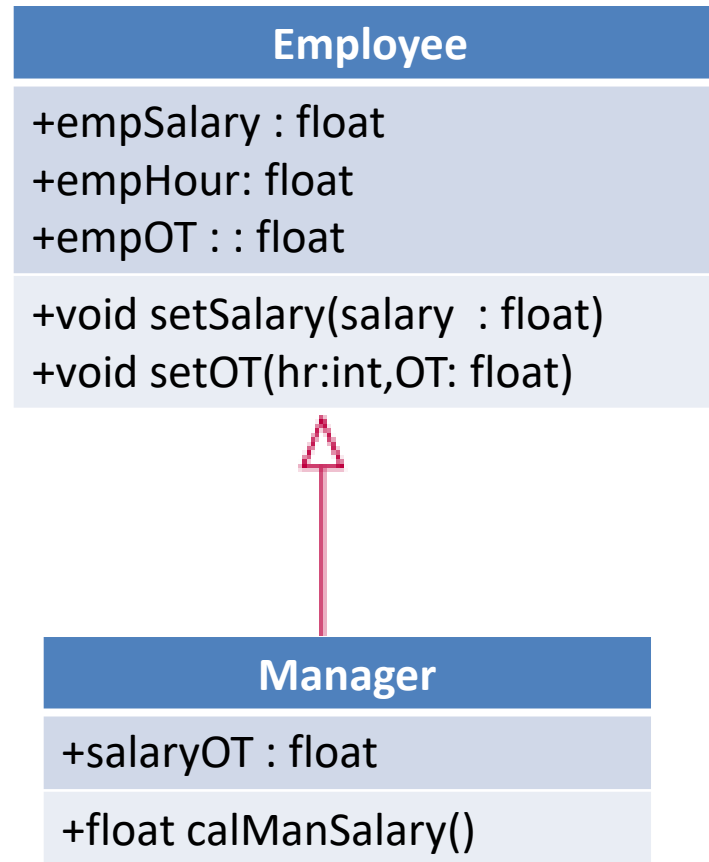
ตัวอย่าง

**class Employee { ..... };**

**class Manager: public Employee { ..... };**

# Single Inheritance

---



```
1. #include <iostream>
2. #include <iomanip>
3. using namespace std;
4. class Employee { // base class
5.     public:
6.         float empSalary;
7.         float empHour;
8.         float empOT;
9.         void setSalary(float salary){
10.             empSalary = salary;
11.             cout<<"Salary = "<<empSalary<<endl;
12.         };
13.         void setOT(int hr,float OT){
14.             empHour = hr;
15.             empOT = OT;
16.         };
17. };
```

19. // sub class derived from two base classes

20. **class Manager: public Employee{**

21. **public:**

22. **float salaryOT;**

23. **float calManSalary(){**

24. **salaryOT =empHour\*empOT;**

25. **return(salaryOT);**

26. **};**

27. **};**

28.

29. // main function

30. **main()**

31. **{**

32. **Manager man1;**

33. **man1.setOT(2,1000); //base class (Employee)**

34. **cout<<fixed<<setprecision(2);**

35. **cout<<"OT = "<<man1.calManSalary()<<endl; //derived class (Manager)**

36. **man1.setSalary(50000); //base class (Employee)**

37. **}**

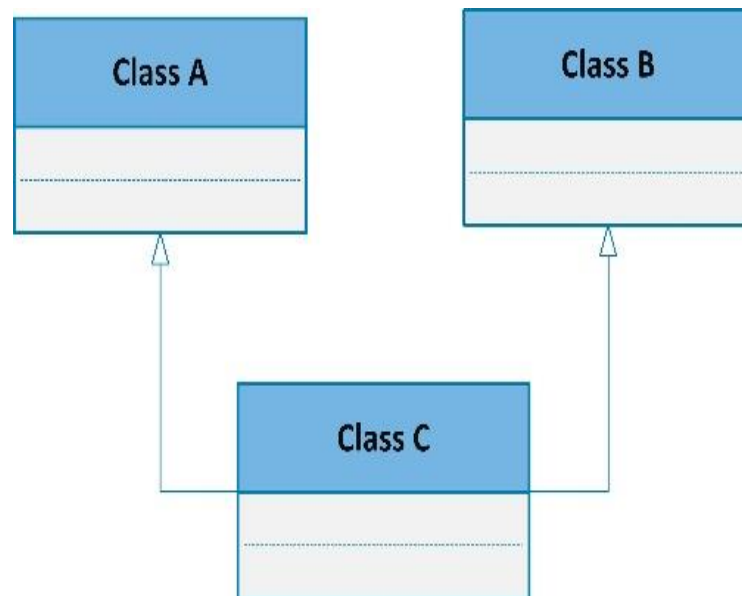
E:\Jirawan drive\Teacher\สอน สอน ส

```
OT = 2000.00
Salary = 50000.00
```

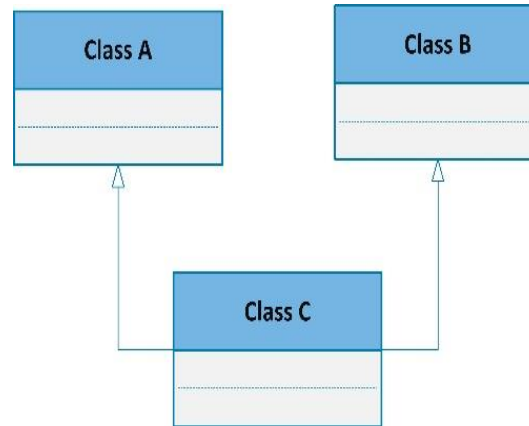
# รูปแบบของการสืบทอด (Type of Inheritance)

---

2. Multiple Inheritance: คลาสย่อยที่สืบทอดคุณสมบัติจากคลาสหลักมากกว่า 1 คลาส



# Multiple Inheritance



รูปแบบการเขียน

```
class A { ..... }; //เป็นคลาสหลัก A
```

```
class B { ..... }; //เป็นคลาสหลัก B
```

```
class C : access_specifier A, access_specifier B { ..... }; //คลาสย่อยเรียกสืบทอดคลาสหลัก A และ B
```

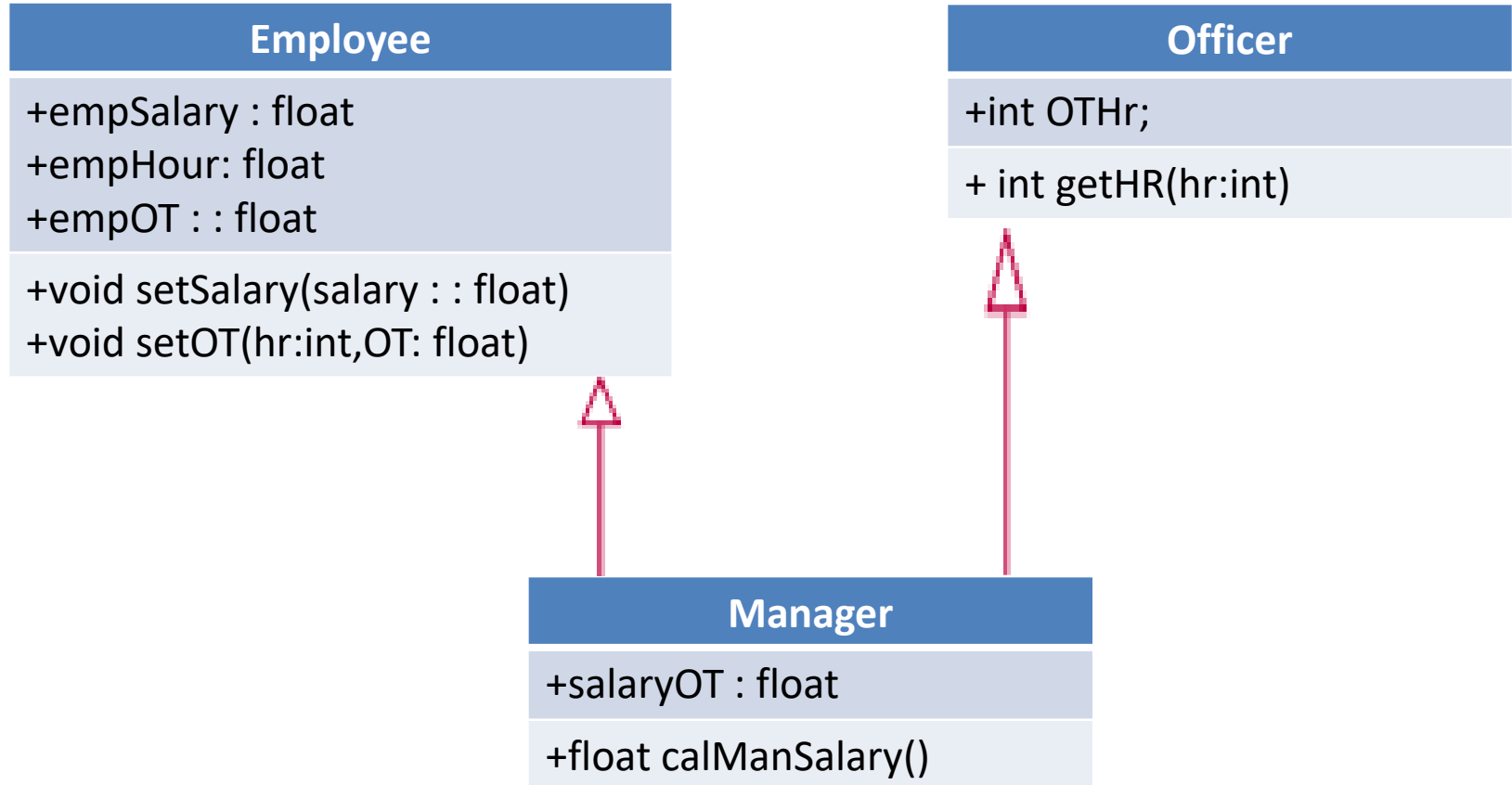
ตัวอย่าง

```
class Employee { ..... };
```

```
class Officer { ..... };
```

```
class Manager: public Employee, public Officer{ ..... };
```

# Multiple Inheritance



```
1. #include <iostream>
2. #include <iomanip>
3. using namespace std;
4. class Employee { // base class
5.     public:
6.         float empSalary, empHour, empOT;
7.         void setSalary(float salary){
8.             empSalary = salary;
9.             cout<<"Salary = "<<empSalary<<endl;
10.        };
11.        void setOT(int hr,float OT){
12.            empHour = hr;
13.            empOT = OT;
14.        };
15. };
16. class Officer {
17.     public:
18.         int OTHr;
19.         int getHR(int hr){
20.             OTHr = hr-8;      return(OTHr);
21.         };
22. };
```



```

23. // sub class derived from two base classes
24. class Manager: public Employee, public Officer{
25.     public:
26.         float salaryOT;
27.         float calManSalary(){
28.             salaryOT = empHour*empOT;
29.             return(salaryOT);
30.         };
31. };
32.
33. // main function
34. main()
35. {
36.     Manager man1;
37.     int numHr;
38.     numHr= man1.getHR(10); //base class (Officer)
39.     man1.setOT(numHr,1000); //base class (Employee)
40.     cout<<fixed<<setprecision(2);
41.     cout<<"OT = "<<man1.calManSalary()<<endl; //derived class (Manager)
42.     man1.setSalary(50000); //base class (Employee)
43. }

```

```

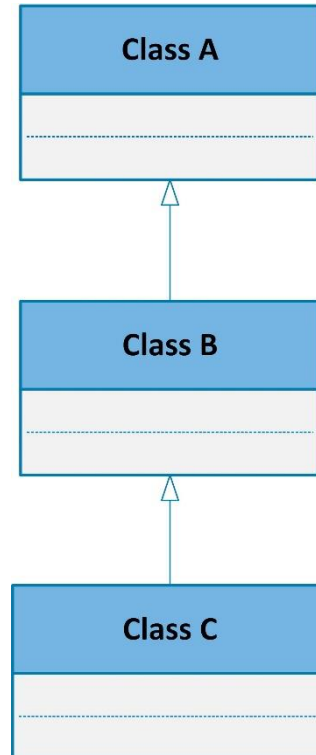
OT = 2000.00
Salary = 50000.00

```

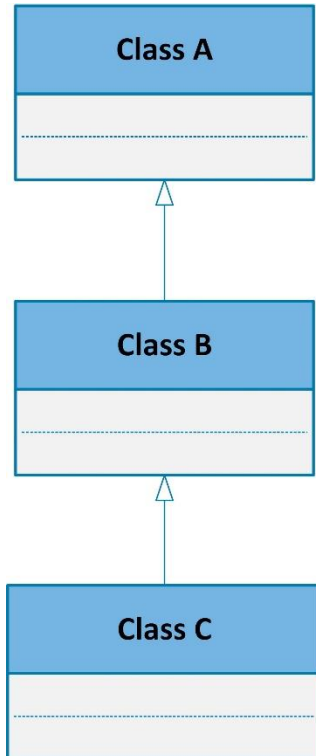
# รูปแบบของการสืบทอด (Type of Inheritance)

---

3. Multilevel Inheritance: คลาสย่อยที่สืบทอดคุณสมบัติมาจากคลาสที่มีการสืบทอดมาจากหลักอีกครั้งหนึ่ง



# Multilevel Inheritance



รูปแบบการเขียน

```
class A { ..... }; //เป็นคลาสหลัก
```

```
class B : access_specifier A { ..... }; //คลาสย่อย B เรียกสืบทอดคลาสหลัก A
```

```
class C : access_specifier B { ..... }; //คลาสย่อย C เรียกสืบทอดคลาสย่อย B
```

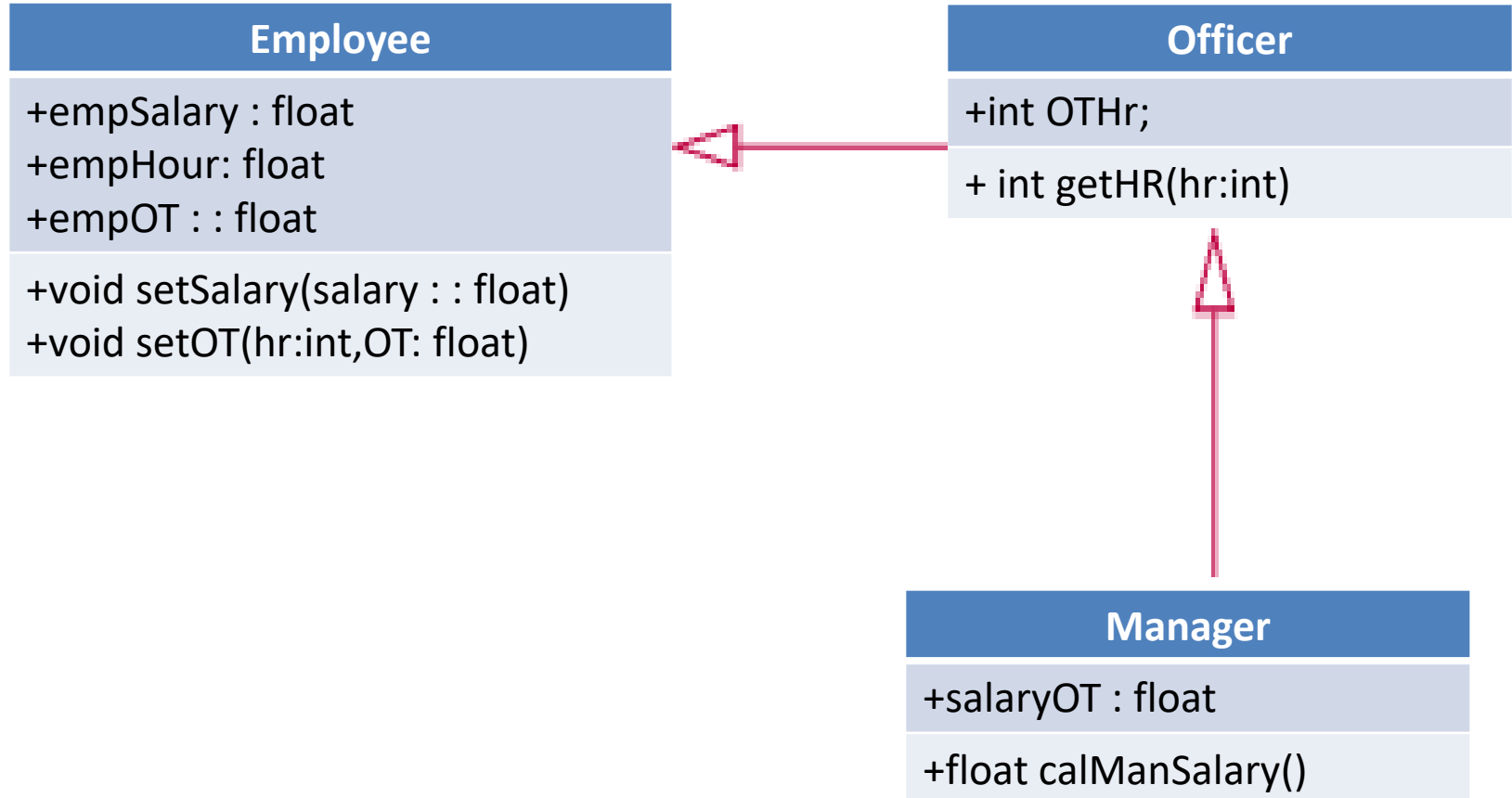
ตัวอย่าง

```
class Employee { ..... };
```

```
class Officer: public Employee {..... };
```

```
class Manager: public Officer{..... };
```

# Multilevel Inheritance



```
1. #include <iostream>
2. #include <iomanip>
3. using namespace std;
4. class Employee { // base class
5.     public:
6.         float empSalary, empHour, empOT;
7.         void setSalary(float salary){
8.             empSalary = salary;
9.             cout<<"Salary = "<<empSalary<<endl;
10.        };
11.        void setOT(int hr,float OT){
12.            empHour = hr; empOT = OT;
13.        };
14. };
15. class Officer : public Employee {
16.     public:
17.         int OTHr;
18.         int getHR(int hr){
19.             OTHr = hr-8;
20.             return(OTHr);
21.         }
22. };
```

23. // sub class derived from two base classes

24. **class Manager: public Officer{**

25. **public:**

26. **float salaryOT;**

27. **float calManSalary(){**

28. **salaryOT = empHour\*empOT;**

29. **return(salaryOT);**

30. **};**

31. **};**

32. **main()**

33. **{**

34. **Manager man1;**

35. **int numHr;**

36. **numHr= man1.getHR(10); //derived class (Officer)**

37. **man1.setOT(numHr,1000); //base class (Employee)**

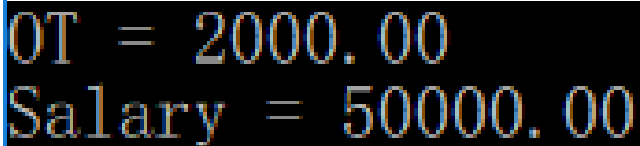
38. **cout<<fixed<<setprecision(2);**

39. **cout<<"OT = "<<man1.calManSalary()<<endl; //derived class (Manager)**

40. **man1.setSalary(50000); //derived class (Officer)**

41. **}**

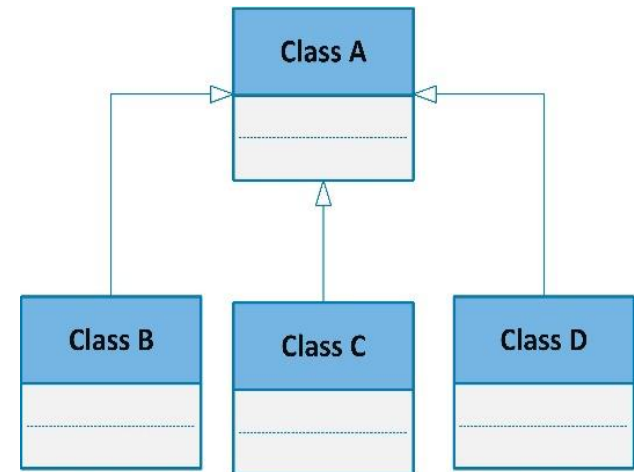
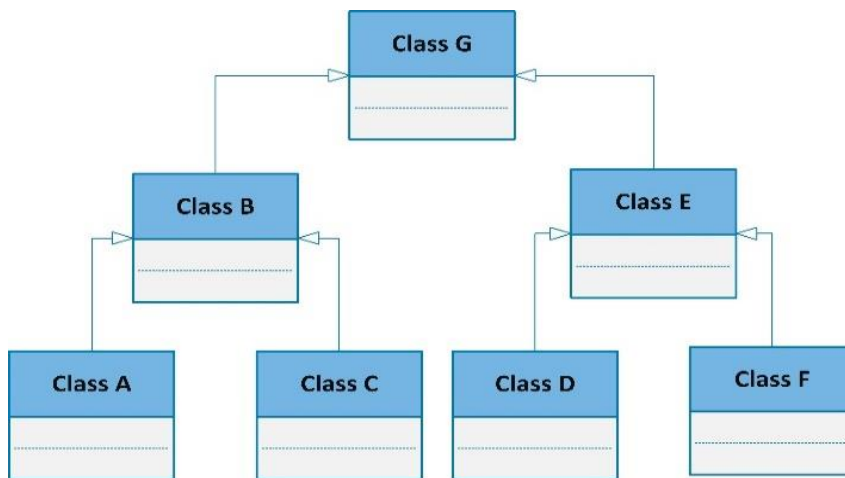
 E:\Jirawan drive\Teacher\สอน สอ



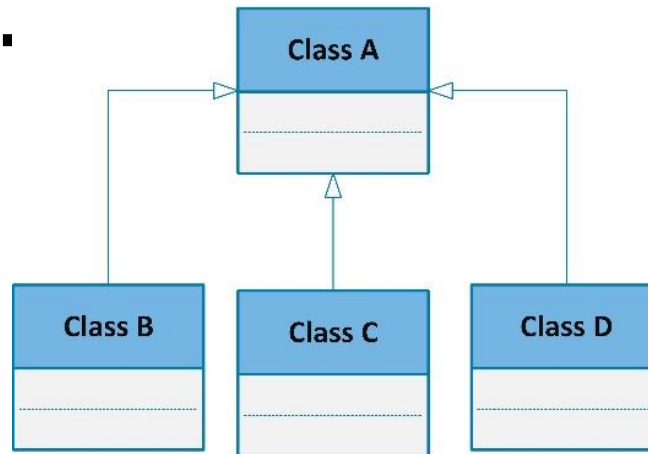
```
OT = 2000.00
Salary = 50000.00
```

# รูปแบบของการสืบทอด (Type of Inheritance)

4. Hierarchical Inheritance: คลาสหลักที่ถูกสืบทอดไปยังคลาสย่อยหลายๆคลาส



# Hierarchical Inheritance



รูปแบบการเขียน

**class A { ..... }; //เป็นคลาสหลัก**

**class B : access\_specifier A { ..... }; //คลาสย่อย B เรียกสืบทอดคลาสหลัก A**

**class C : access\_specifier A { ..... }; //คลาสย่อย C เรียกสืบทอดคลาสย่อย A**

**class D : access\_specifier A { ..... }; //คลาสย่อย D เรียกสืบทอดคลาสย่อย A**

ตัวอย่าง

**class Employee { ..... };**

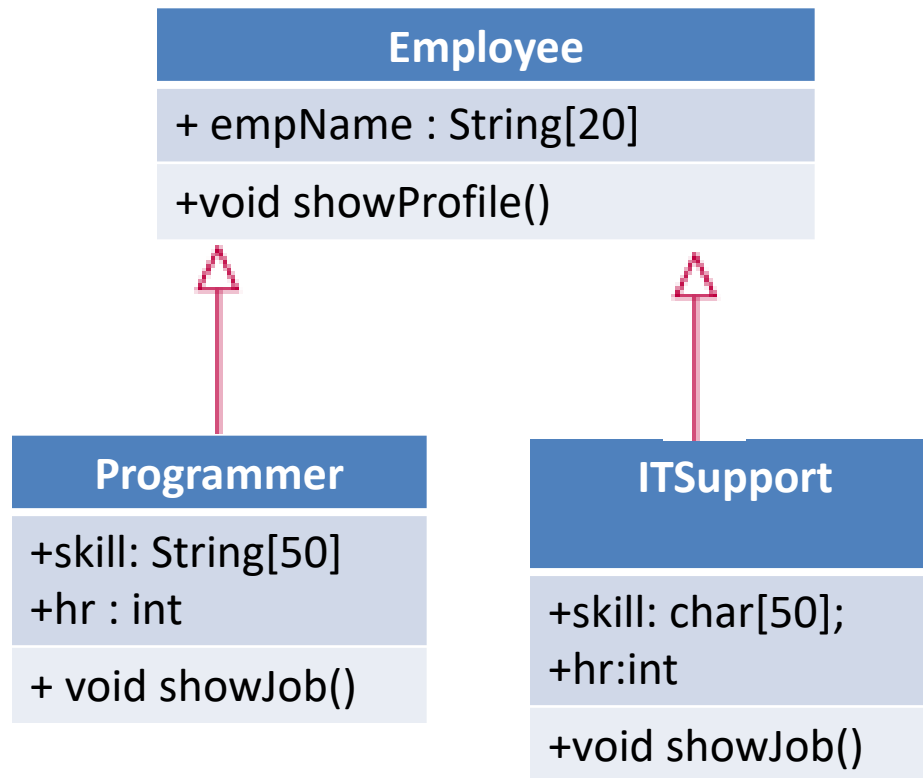
**class Programmer: public Employee { ..... };**

**class ITSupport: public Employee { ..... };**



# Hierarchical Inheritance

---



```
1.  #include <iostream>
2.  #include <iomanip>
3.  using namespace std;
4.  class Employee {
5.      public:
6.          char empName[20];
7.          void showProfile(){
8.              cout<<"Name = ";cin>>empName;
9.          };
10. };
11. class Programmer : public Employee {
12.     public :
13.         char skill[50];
14.         int hr;
15.         void showJob(){
16.             Employee::showProfile(); //base class (Employee)
17.             cout<<"Please input Programmer Skill=> ";cin>>skill;
18.             cout<<"Please input hours=> ";cin>>hr;
19.             cout<<"*****\nProgrammer skill = "<<skill<<endl;
20.             cout<<"Programmer hours = "<<hr<<endl<<endl;
21.         };
22.     };
```

```

23.  class ITSupport : public Employee {
24.      public:
25.          char skill[50];
26.          int hr;
27.          void showJob(){
28.              Employee::showProfile(); //base class (Employee)
29.              cout<<"Please input IT Skill=> ";cin>>skill;
30.              cout<<"Please input hours=> ";cin>>hr;
31.              cout<<"*****\nIT skill = "<<skill<<endl;
32.              cout<<"IT hours = "<<hr<<endl<<endl;
33.          };
34. };
35. main()
36. {
37.     Programmer pro;
38.     ITSupport it;
39.
40.     pro.showJob(); //base class (Employee)
41.     it.showJob(); //base class (Employee)
42.
43. }

```

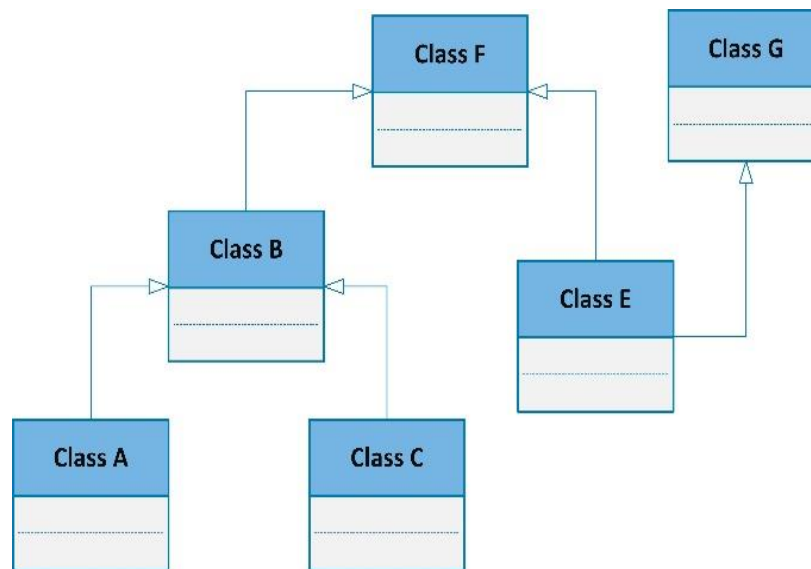
```
Name = Jirawan  
Please input Programmer Skill=> C++  
Please input hours=> 10  
*****  
Programmer skill = C++  
Programmer hours = 10
```

```
Name = Jaruwan  
Please input IT Skill=> Network  
Please input hours=> 12  
*****  
IT skill = Network  
Programmer hours = 12
```

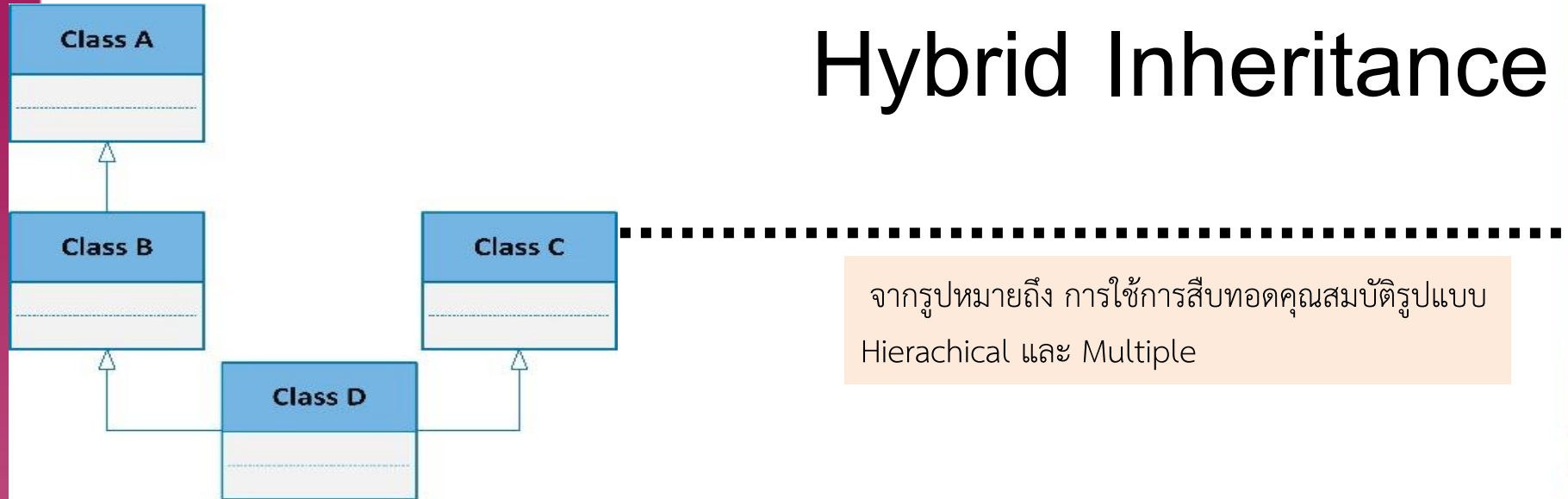
# รูปแบบของการสืบทอด (Type of Inheritance)

---

5. : Hybrid Inheritance หมายถึง การรวมรูปแบบการสืบทอด  
คุณสมบัติมากกว่า 1 รูปแบบเข้าไว้ด้วยกัน



# Hybrid Inheritance



รูปแบบการเขียน

**class A { ..... }; //เป็นคลาสหลักลำดับที่ 1**

**class B : public A { ..... }; //คลาสร้อย B เรียกสืบทอดคลาสหลัก A**

**class C { ..... }; //เป็นคลาสหลักลำดับที่ 2**

**class D : public B, public C { ..... }; //คลาสร้อย D เรียกสืบทอดคลาสร้อย B และคลาสหลัก A**

ตัวอย่าง

**class Employee { ..... };**

**class Officer: public Employee { ..... };**

**class Programmer : public Officer { ..... };**

**class Computer{ ..... };**

**class Programmer : public Computer{ ..... };**

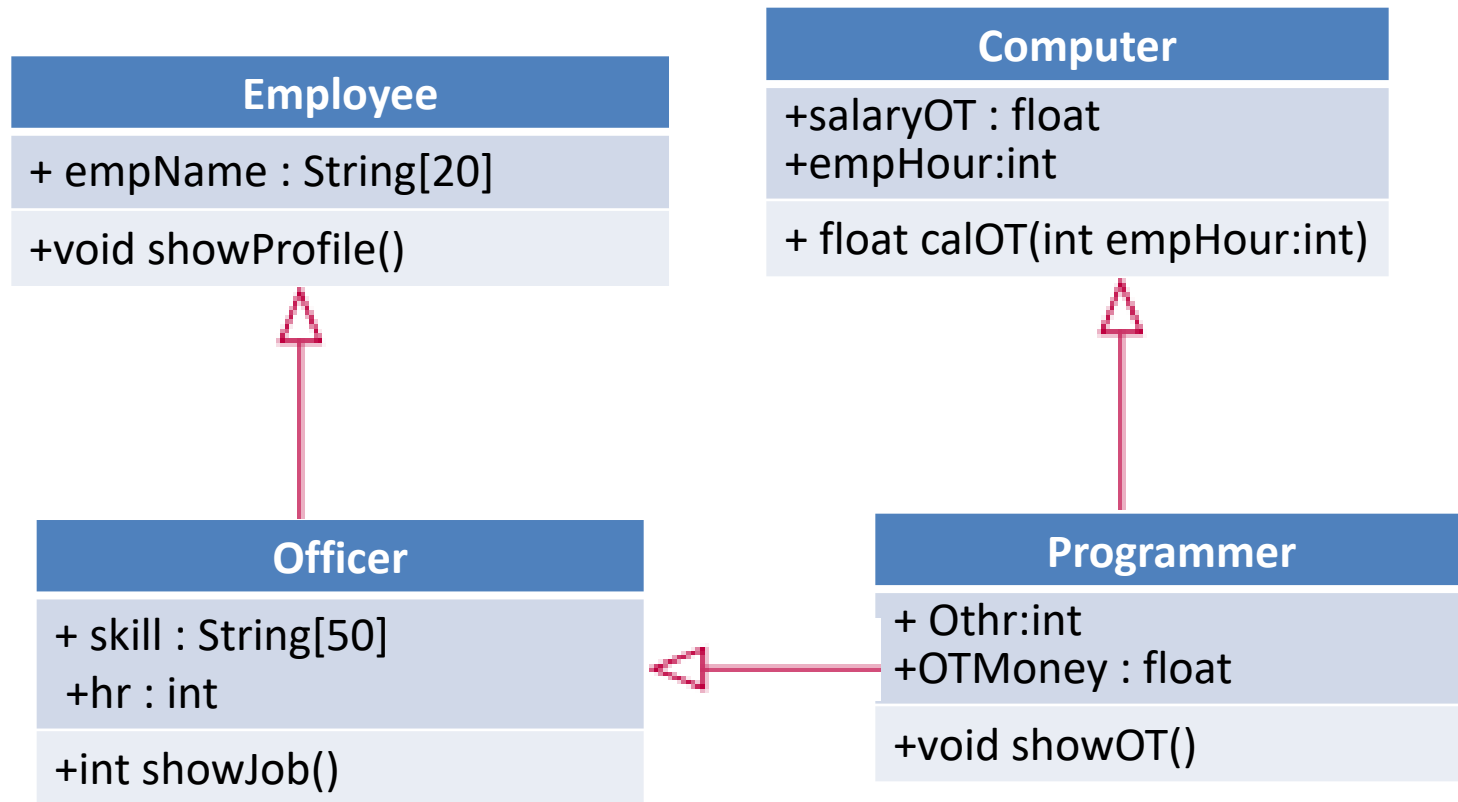
## ตัวอย่าง

```
class Employee { ..... };  
class Officer: public Employee { ..... } ;  
class Programmer : public Officer { ..... } ;  
class Computer{ ..... };  
class Programmer : public Computer{ ..... } ;
```

## ตัวอย่าง

```
class Employee { ..... };  
class Officer: public Employee { ..... } ;  
class Computer{ ..... };  
class Programmer : public Officer,public Computer {.... };
```

# Hybrid (Virtual) Inheritance





```

1. #include <iostream>
2. #include <iomanip>
3. using namespace std;
4. class Employee {
5.     public:
6.         char empName[20];
7.         void showProfile(){
8.             cout<<"Name = ";cin>>empName;
9.         };
10. };
11. class Officer : public Employee {
12.     public:
13.         char skill[50]; int hr;
14.         int showJob(){
15.             Employee::showProfile(); //base class (Employee)
16.             cout<<"Please input hours=> ";cin>>hr;
17.
18.             hr =hr-8;
19.             cout<<"OT hours = "<<hr<<endl;
20.             return(hr);
21.         };
22. };

```

23. class Computer{

24. public:

25. float salaryOT;

26. int empHour;

27. float calOT(int empHour){

28. salaryOT =empHour\*150;

29. return(salaryOT);

30. };

31. };

32. class Programmer : public Officer, pub

33. public:

34. int OThr; float OTMoney;

35. void showOT(){

36. OThr = Officer::showJob(); //derived class (Officer)

37. OTMoney = Computer::calOT(OThr);

38. cout<<"OT salary = "<<OTMoney<<endl<<endl;

39. };

40. };

41. main()

42. {

43. Programmer pro;

44. pro.showOT();

45. }

Name = JIRAWAN

Please input hours=> 10

OT hours = 2

OT salary = 300

# คำถาม

---

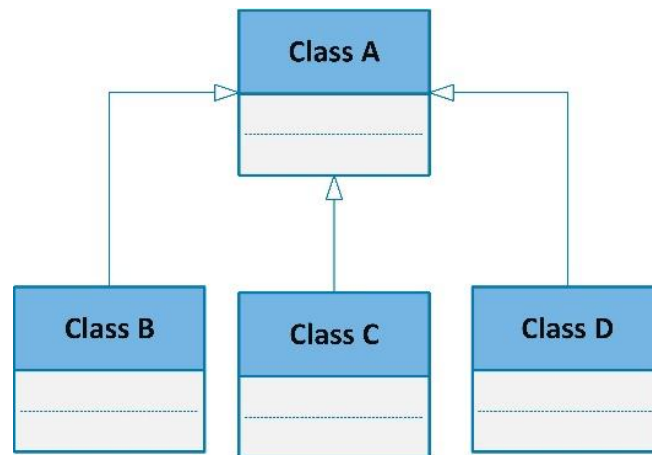


# Quiz 13

---

1) จงเขียนโปรแกรมคำนวณค่าเช่าห้องพักชื่อ ABC โดยเขียนโปรแกรมให้มีรูปแบบเป็น Hierarchical Inheritance พร้อมทั้งเขียน UML diagram ซึ่งโปรแกรมนี้จะคิดค่าเช่าเป็นรายเดือน โดยรายละเอียดค่าเช่าห้องพักมีดังต่อไปนี้

- a) ห้องพัสดุ      ค่าเช่าห้องละ 2,500 บาท/เดือน
- b) ห้องแอร์      ค่าเช่าห้องละ 3,500 บาท /เดือน
- c) ห้อง VIP      ค่าเช่าห้องละ 10,000 บาท /เดือน
- d) ค่าไฟฟ้า      คิดหน่วยละ 15 บาท
- e) ค่าน้ำ      คิดหน่วยละ 30 บาท



# Hierarchical Inheritance

---

