OOP & data struct

3. Inheritance (การสืบทอด)

BY SOMSIN THONGKRAIRAT

ATTASIT LASAKUL EDITTION

Inheritance (การสืบทอด)

คือการสร้างคลาสใหม่(คลาสลูก)โดยการใช้คลาสเดิม(คลาสแม่)เป็นต้นแบบ ข้อดีคือ

- คลาสใหม่(คลาสลูก)จะสามารถใช้ตัวแปร, methods และอื่นๆของคลาสแม่ได้ 90% (ตามข้อจำกัดของการเข้าถึง private, protect, public)
- คลาสใหม่(คลาสลูก)จะสามารถเพิ่มตัวแปร, methods และอื่นๆได้ตามต้องการ
- สามารถให้คลาสใหม่อื่นนำไปสืบทอดต่อได้อีก

โครงร่างโปรแกรม 3_1..CPP

class

methods

constructor

deconstructor

main

```
#include <iostream>
using namespace std;
class media
 {  private:
        int playing_sec;
    public:
       string fullname;
    void play(int time){.. }
    media( ) { ..... }
    ~media( ) { ..... }
}
int main( )
{
    media s1, s2 ,....
     s1.play(500);
      ...............
}
```

โครงร่างโปรแกรม 3_2..CPP

```
#include <iostream>
using namespace std;
class media
    class song
        class movie
            class episode
             {  private:
                    int playing_sec;
                public:
                    int episode_number;
                void play(int time){.....}
                episode( ) { ..... }
                ~episode( ) { ..... }
             }
int
{
    medi
     s1.play
      ...............
}
```

โครงร่างโปรแกรม 3_3..CPP

```
#include <iostream>
using namespace std;
class media
{  private:
   class song : public media
   {  public:  string band; }
   class movie : public media
   {  public:  string director; }
class episode : public media
{  public:  int episode_number; }
    int main( )
    {
        song s1, s2 ,....
        movie m1,m2,...
        s1.play(500);
        m1.play(200);
```

โครงร่างโปรแกรม 3_4, 3_5..CPP
(Call base constructor)

```
#include <iostream>
using namespace std;
class media
{  private:
   class song : public media
   {  public:  string band; }
   class movie : public media
   {  public:  string director; }
   class episode : public media
   {  public:  int episode_number; }
int main( )
{
    song s1, s2 ,....
     movie m1,m2,...
     s1.play(500);
     m1.play(200);
```

โครงร่างโปรแกรม 3_6..CPP
(Another example class car)

```
#include <iostream>
using namespace std;
class vehicle
{  public:
   class motorbike : public vehicle
   {  public:  void print() }
   class airplane : public vehicle
   {  public:  void print() }
   class submarine : public vechicle
   {  public:  void print(); }
int main( )
{
    motorbike vespa_sprint;
    airplane a380(22);
     submarine S26T;
    vespa_sprint.print( );
```

โครงร่างโปรแกรม 3_7..CPP
(LAB)

```
#include <iostream>
using namespace std;
class media
{  private:
   class song : public media
   {  public:  string band; }
   class movie : public media
   {  public:  string director; }
   class episode : public media
   {  public:  int episode_number; }
int main( )
{
    song s1, s2 ,....
     movie m1,m2,...
     s1.play(500);
     m1.play(200);
```

โปรแกรม 3_1.CPP

```cpp
class media
{
private:
    int playing_sec; // last played seconds in episode
public:
    string name;
    string author;
    int length; // media length in Seconds

    void play(int time)
    {
        playing_sec += time;
        if (playing_sec >= length)
            {playing_sec = length;}
        int remaining_time = length - playing_sec;
        if (remaining_time < 0)
            {remaining_time = 0;}
        if (time > remaining_time)
        {
            cout << "Playing " << name << " at[" << playing_sec << "] : [" << remaining_time << " sec] remaining" << endl;
            playing_sec = 0;
        }
        else
            cout << "Playing " << name << " at[" << playing_sec << "] : [" << remaining_time << " sec] remaining" << endl;
    }
```

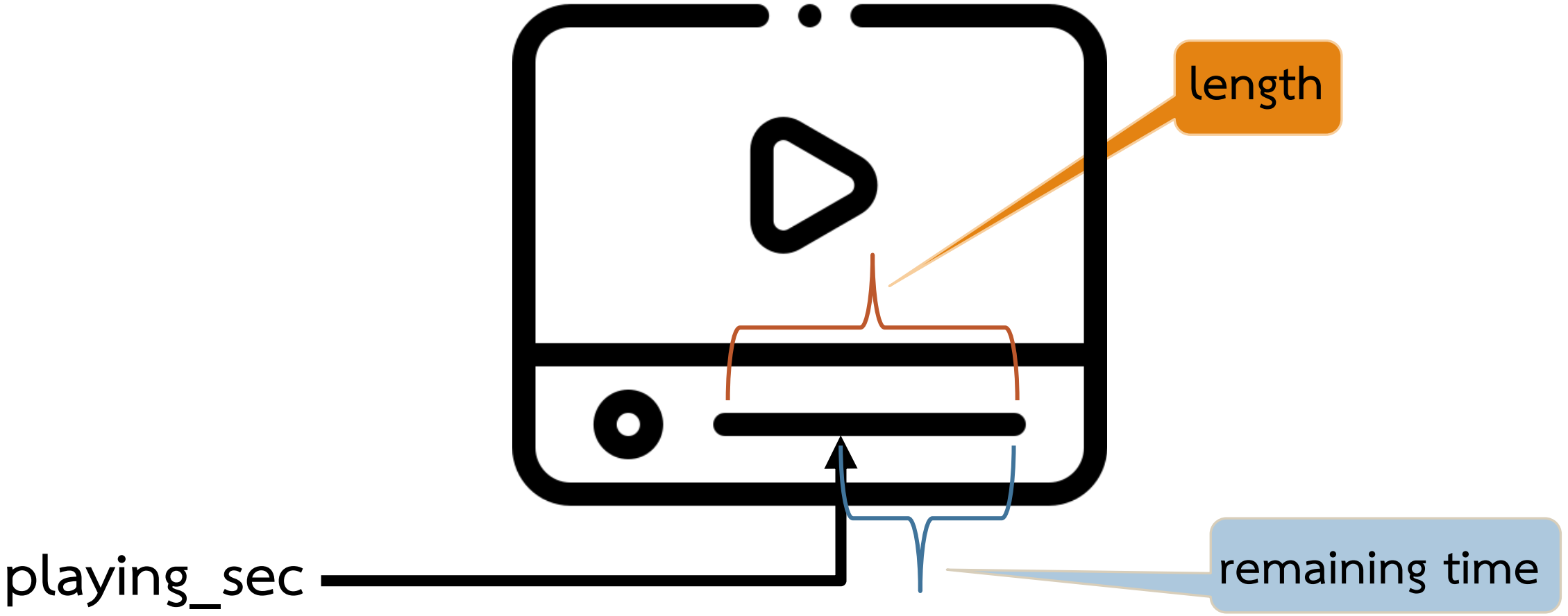
3_1.cpp

constructor ก็เป็นดังนี้

```
    media()
    { // default constructor
        name = "unknow";
        length = 0;
        playing_sec = 0;
    }

    media(string _name, string _author, int _length)
    { // 3 parameter constructor
        name = _name;
        author = _author;
        length = _length;
        playing_sec = 0;
    }
};
```

class Media เสมือนเป็นเครื่องเล่นหนัง



class media (เพียง Class เดียว) หากเรานำมาใช้เล่นดูหนัง (play) แต่ละครั้งก็จะทราบเพียง => [ชื่อหนัง] [เวลาที่เล่นถึงอยู่] : [เหลือเวลาหนังค้างอยู่เท่าไหร่]

```
int main()
{
    media s1("Som San", "sek loso", 314);
    media s2("Timemachine ", "Pond Nipon", 328);
    media m1("The Disappearance of Haruhi Suzumiya", "Nagaru Tanigawa", 9707);
    media m2("Avatar", "James Cameron", 9720);
    media e1("Thi da Satan : ep 3 Earth Water Wind Fire", "Kantana", 3600);
    media e2("Start-Up (2020) : ep 16 Scale up", "Studio Dragon", 4800);
    system("cls");
    s1.play(314);
    s1.play(20);
    s1.play(30);
    s1.play(30);
    s1.play(294);
    s1.play(314);
    return 0;
}
```

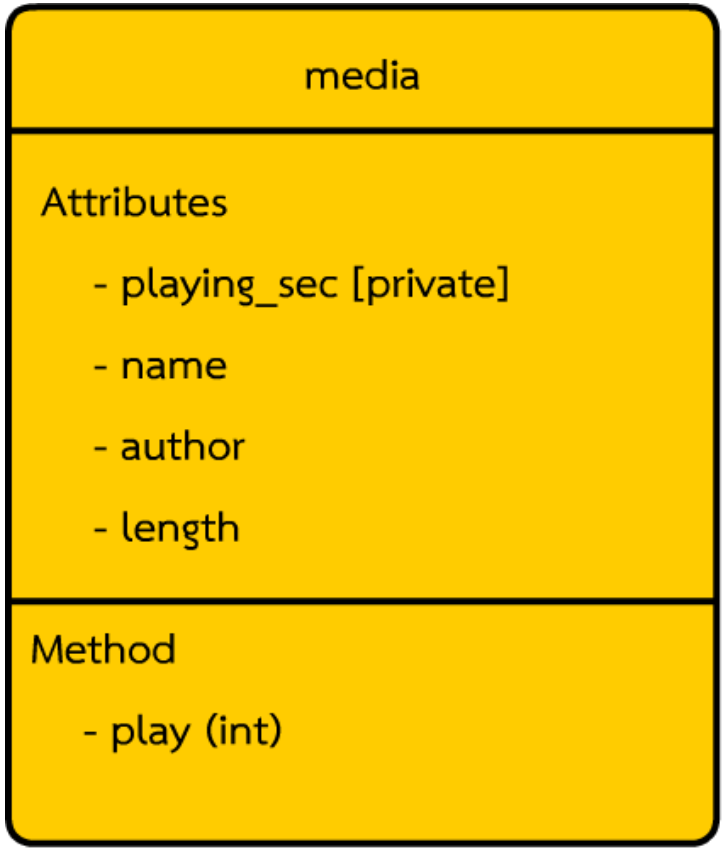
ผลลัพธ์

```
Playing Som San at[314] : [0 sec] remaining
Playing Som San at[20] : [294 sec] remaining
Playing Som San at[50] : [264 sec] remaining
Playing Som San at[80] : [234 sec] remaining
Playing Som San at[314] : [0 sec] remaining
Playing Som San at[314] : [0 sec] remaining
```

ประโยชน์หลักของการสืบทอดคลาส

หากเราจะเพิ่มความสามารถของคลาส media ให้สามารถนำไปใช้ เล่นหนัง, เล่นเพลงหรือให้สามารถเลือกตอน (หนัง/เพลง)ได้ เราสามารถทำได้โดยง่ายด้วย ใช้วิธีสืบทอดจากคลาส media เพื่อสร้างคลาสใหม่เพื่อเพิ่มความสามารถได้ตามต้องการ

โปรแกรม 3_2 - 3_3.cpp ตัวอย่างสร้างคลาสใหม่



สร้าง Class ต่อไปนี้ใหม่จาก Class media

- Class Movie
- Class Song
- Class Episode

โครงร่างโปรแกรม 3_1..CPP

class

methods

constructor

deconstructor

main

```
#include <iostream>
using namespace std;
class media
 {  private:
        int playing_sec;
    public:
       string fullname;
    void play(int time){.. }
    media( ) { ..... }
    ~media( ) { ..... }
}
int main( )
{
    media s1, s2 ,....
     s1.play(500);
      ...............
}
```

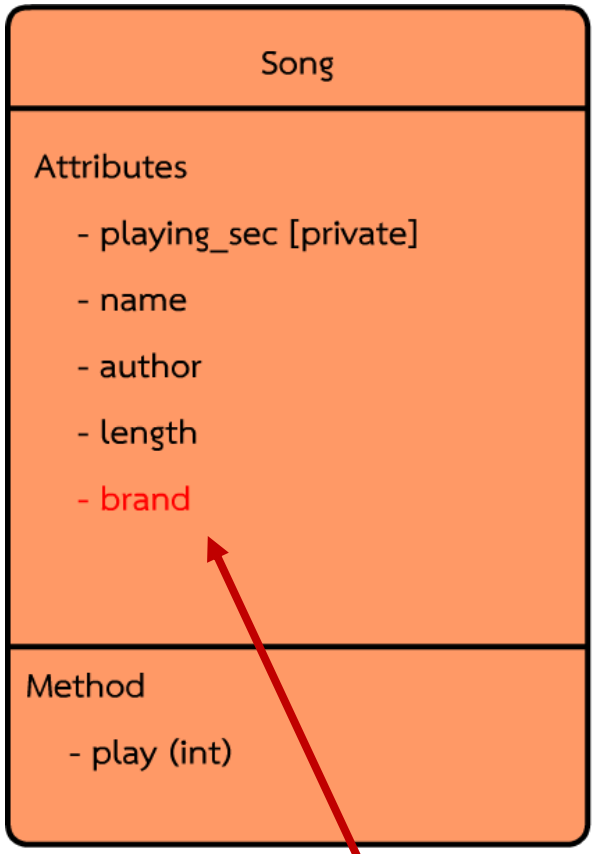
โครงร่างโปรแกรม 3_2..CPP

```
#include <iostream>
using namespace std;
class media
    class song
        class movie
            class episode
             {  private:
                    int playing_sec;
                public:
                    int episode_number;
                void play(int time){.....}
                episode( ) { ..... }
                ~episode( ) { ..... }
             }
int
{
    medi
     s1.play
      ...............
}
```

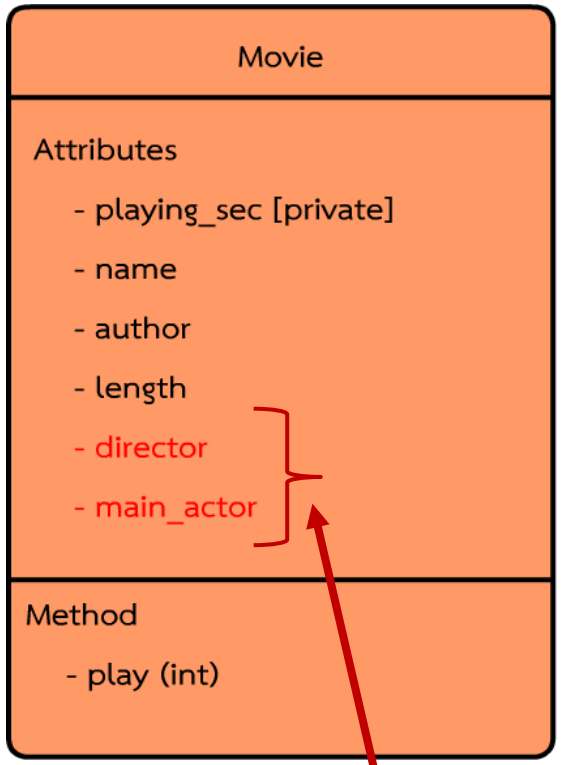
โครงร่างโปรแกรม 3_3..CPP

```
#include <iostream>
using namespace std;
class media
{  private:
   class song : public media
   {  public:  string band; }
   class movie : public media
   {  public:  string director; }
class episode : public media
{  public:  int episode_number; }
    int main( )
    {
        song s1, s2 ,....
        movie m1,m2,...
        s1.play(500);
        m1.play(200);
```

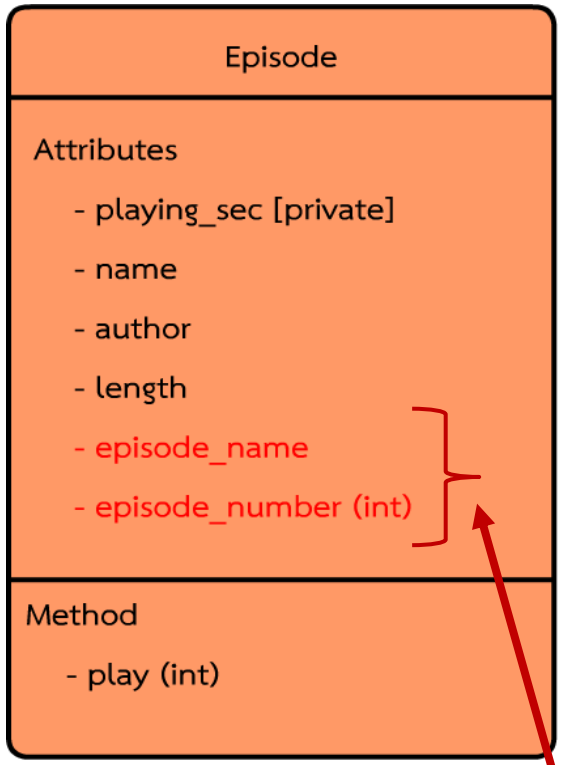
Class ใหม่จาก Class media จะเพิ่มความสามารถไปอีก ให้ตรงกับต้องการ เช่น คลาส Song ใส่ชื่อวง (Brand) เพิ่มได้ และ คลาสภาพยนตร์ (movie) ก็ใส่เพิ่ม director, actor ได้ หรือ คลาส Episode ก็เลือกตอน (Episode)ได้เป็นต้น



Add Brand



Add director , actor



Add episode name and number

หากเราไม่ใช้การสืบทอดก็จะต้องเขียนคลาสใหม่เพิ่มดังนี้ (3_2.CPP)

```
41  class song{
42  private :
43      int playing_sec; // last played seconds in episode
44
45  public :
46      string name;
47      string author;
48      string brand; // added from media
49      int length; // media length in Seconds
50
51      void play(int time){ // play method
52          int remaining_time = length - playing_sec;
53          cout << "Playing " << name << " at[" << playing_sec <<
54          if(time > remaining_time){
55              playing_sec = 0;
56          }
57          else{
58              playing_sec += time;
59          }
60      }
61
62      song(){ // default constructor
63          name = "unknow";
64          length = 0;
65          playing_sec = 0;
66      }
67
68      song(string _name,string _author,int _length){ // 3 parame
69          name = _name;
70          author = _author;
```

Class song

```
77  class movie{
78  private :
79      int playing_sec; // last played seconds in episode
80
81  public :
82      string name;
83      string author;
84      string director;
85      string main_actor;
86      int length; // media length in Seconds
87
88      void play(int time){ // play method
89          int remaining_time = length - playing_sec;
90          cout << "Playing " << name << " at[" << playing_sec <<
91          if(time > remaining_time){
92              playing_sec = 0;
93          }
94          else{
95              playing_sec += time;
96          }
97      }
98
99      movie(){ // default constructor
100         name = "unknow";
101         length = 0;
102         playing_sec = 0;
103     }
104
105     movie(string _name,string _author,int _length){ // 3 para
```

Class movie

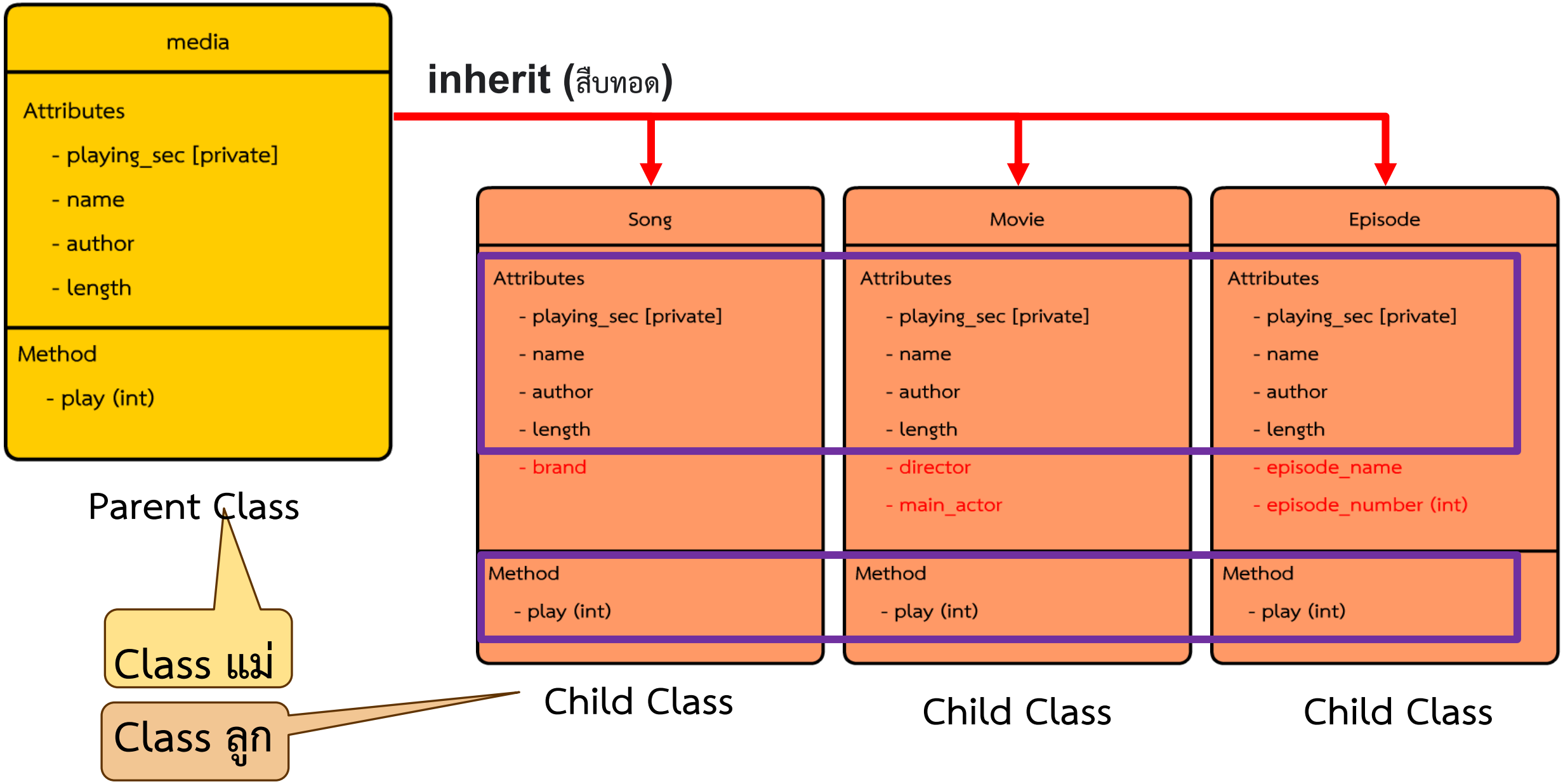
```
114 class episode {
115 private :
116     int playing_sec; // last played seconds in episode
117
118 public :
119     string name;
120     string author;
121     int episode_number;
122     string episode_name;
123     int length; // media length in Seconds
124
125     void play(int time){ // play method
126         int remaining_time = length - playing_sec;
127         cout << "Playing " << name << " at[" << playing_sec
128         if(time > remaining_time){
129             playing_sec = 0;
130         }
131         else{
132             playing_sec += time;
133         }
134     }
135
136     episode(){ // default constructor
137         name = "unknow";
138         length = 0;
139         playing_sec = 0;
140     }
141
```

Class episode

เราจะใช้วิธี สืบทอด (Inheritance) จะง่ายกว่าเพราะ...

- คลาสแม่จะส่งต่อส่วนประกอบตนเองไปยัง child class (คลาสลูก) หรือ derived class

(**นอกจาก method Constructure( )จะไม่ส่งให้)


media
Attributes
- playing_sec [private]
- name
- author
- length
Method
- play (int)
inherit (สืบทอด)
Song
Attributes
- playing_sec [private]
- name
- author
- length
- brand
Method
- play (int)
Movie
Attributes
- playing_sec [private]
- name
- author
- length
- director
- main_actor
Method
- play (int)
Episode
Attributes
- playing_sec [private]
- name
- author
- length
- episode_name
- episode_number (int)
Method
- play (int)
Parent Class
Class แม่
Class ลูก
Child Class
Child Class
Child Class

วิธีการเขียนสืบทอดคลาส (C++ Syntax)

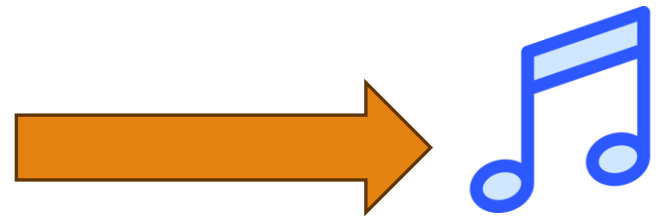
```
class [child_class_name] : [modifier1] [base_class_name1] , [modifier2] [base_class_name2] ...{
    // class component
}
```

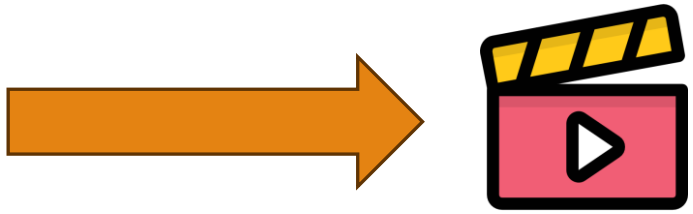
```
class movie : public media{
public :
    string director;
    string main_actor;
};
```

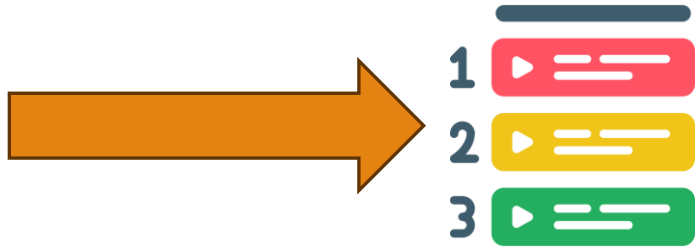
```
class episode : public media{
public :
    int episode_number;
    string episode_name;
};
```

โปรแกรม 3_3.CPP (ยังไม่มี constructor)

```
41  class song : public media{
42  public :
43      string brand;
44  };
45
46  class movie : public media{
47  public :
48      string director;
49      string main_actor;
50  };
51
52  class episode : public media{
53  public :
54      int episode_number;
55      string episode_name;
56  };
```

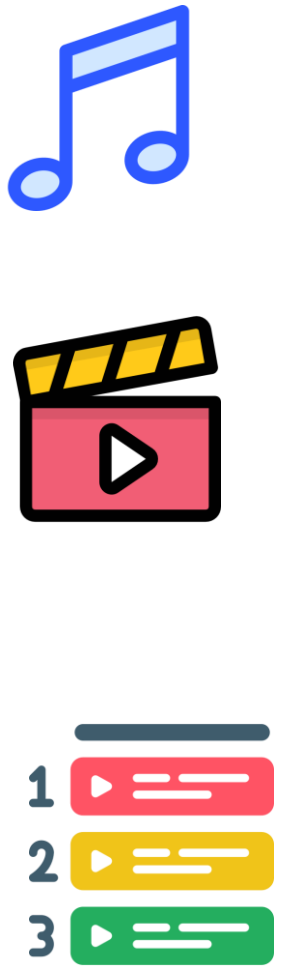






ฉะนั้น ใน main( ) เราก็สามารถนำทั้งสามคลาสมาสร้าง Object เพื่อกำหนดค่าต่างๆได้ทันที

```
67      song s1;
68      s1.fullname = "Som San";
69      s1.author = "sek loso";
70      s1.length = 314;
71      s1.brand = "LOSO";
72      s1.play(10);
73
74      movie m1;
75      m1.fullname = "Avatar";
76      m1.author = "James Cameron";
77      m1.length = 9720;
78      m1.director = "James Cameron";;
79      m1.main_actor = "Sam Worthington";
80      m1.play(20);
81
82      episode e1;
83      e1.fullname = "Thi da Satan : ep 3 Earth Water Wind Fire";
84      e1.author = "Kantana";
85      e1.length = 4800;
86      e1.episode_number = 3;
87      e1.episode_name = "Earth Water Wind Fire";
88      e1.play(30);
```



Constructor จะไม่สืบทอดไปคลาสลูก

(ในโปรแกรมที่ 3_4, 3_5.CPP คลาสลูกจะสร้าง constructor ของคลาสตนเอง (แต่ก็จะมีการเรียกใช้ constructor ของคลาสแม่ด้วยเช่นกัน)

คือ method พิเศษที่ เรามักจะสร้างไว้ในคลาส (หรือไม่สร้างก็ได้ แต่ตัวโปรแกรมก็จะสร้างให้เอง (หากเราไม่สร้าง) เพียงแต่มองไม่เห็น เรียก Default constructor ซึ่งแบบนี้จะเป็นแบบที่ ไม่มีการับค่าหรือส่งค่ากลับแต่อย่างใด

คือ method พิเศษที่ หากเราได้สร้างไว้ในคลาสก็สามารถจะให้มีการรับค่าด้วยก็ได้ ซึ่งมักจะใช้กำหนดค่าเริ่มต้นต่างๆ แต่ก็จะไม่มีการส่งค่ากลับแต่อย่างใด

คือ method พิเศษที่ จะถูกเรียกใช้งานอัตโนมัติเป็น method แรก ทุกครั้งเมื่อมีการนำคลาสนั้นไปสร้าง Object และจะถูกเรียกใช้ใช้เพียงครั้งเดียวเท่านั้น

คือ method พิเศษที่ คลาสที่ไม่สืบทอดไป (คลาสลูก) แต่ก็จะสามารถเรียกใช้งาน method นี้ได้

constructor เป็น methods มักจะใช้เพื่อกำหนดค่าเริ่มต้นของ Object ที่สร้างจากคลาสนั้น

```
5  class media{
6  public :
7
8      media(){ // default constructor
9
10         cout << "default constructor of [media] class" << endl;
11
12         name = "unknow";
13         length = 0;
14         playing_sec = 0;
15     }
16
```

เมื่อเรานำไปสร้าง Object ดังนี้ → `media m2;`

ก็จะเกิดการใช้งานหนึ่งครั้งทันที → default constructor of [media] class

3_4.cpp

กรณีของการมีการสืบทอด Class ตัว Constructor ของ Class แม่จะถูกเรียกก่อนตามลำดับไปสู่ตัว Class ลูก

```
class media
{
public:
    int playing_sec; // last played seconds in episode
    string name;
    string author;
    int length; // media length in Seconds
media()
{ // default constructor
    cout << "default constructor of [media] class" << endl;
    name = "unknow";
    length = 0;
    playing_sec = 0;
}
```

```
class song : public media
{
public:
    string brand;

    song()
    { // default constructor
        cout << "default constructor of [song] class" << endl;
    }
```

เมื่อสร้าง Object ของ Class movie

```
song s1;
```

ผลคือ Con. Class แม่ และ ลูก ทำงานตามลำดับ

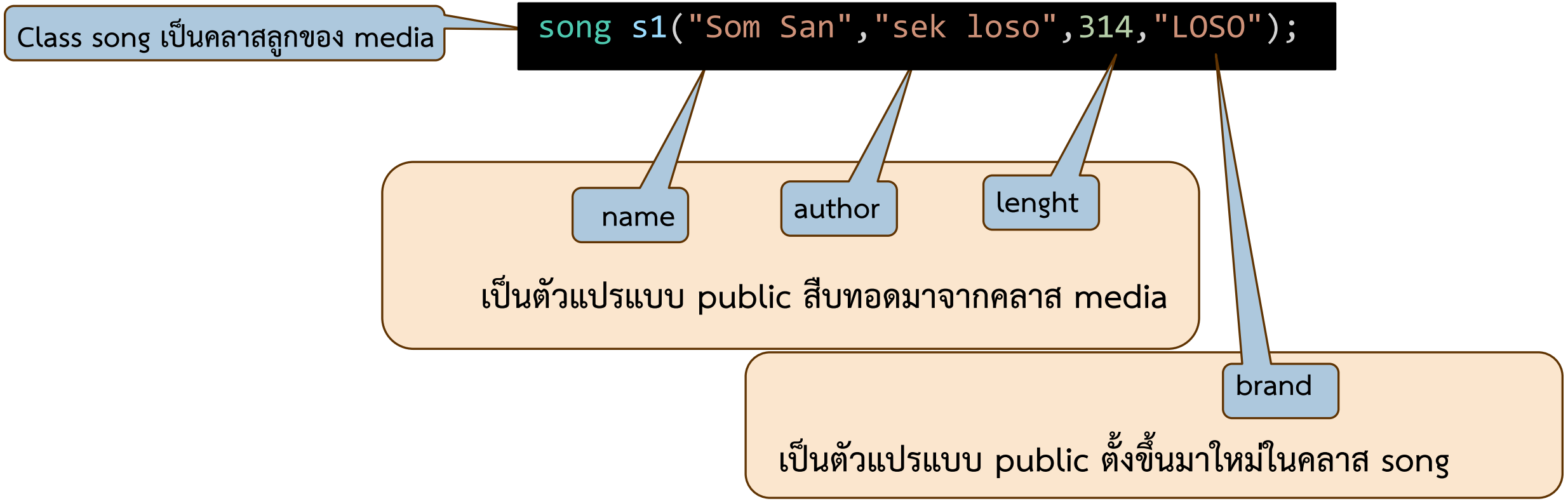
ผลลัพธ์แสดงหน้าจอ

default constructor of [media] class

default constructor of [song] class

3_4.cpp

ในกรณี construct ที่มีการรับค่านั้น (ดังในโปรแกรม 3_4.CPP) Class ลูก เราก็สามารถเรียกใช้งาน construct ของ Class แม่ได้ดังนี้



3_4.cpp

กรณีนี้เราสามารถกำหนดค่าเริ่มต้นได้ 2 วิธี

1) ใช้ constructor ของคลาสลูกเองก็เพียงพอ เพราะตัวแปรทุกตัวเราอ้างถึงได้อยู่แล้ว ดังนี้

```
song(string _name, string _author, int _length, string _brand)
{ // 4 parameter constructor
    cout << "4 parameter constructor of [song] class" << endl;
    brand = _brand;
    name = _name;
    length = _length;
    author = _author;
}
```

2) ใช้ constructor ของคลาสลูกแล้วอ้างต่อไป constructor ของคลาสแม่ เพราะที่ contrcutor ของคลาสแม่ก็มีการการกำหนดตัวแปรทุกตัวที่เราอ้างถึงอยู่แล้วเช่นกัน

Class แม่ มี constructor ที่มีการรับค่าจำนวน 3 ค่า

```
media(string _name,string _author,int _length){ // 3 parameter constructor

    cout << "3 parameter constructor of [media] class" << endl;

    name = _name;
    author = _author;
    length = _length;
    playing_sec = 0;
}
```

3_4.cpp

โดยเรากำหนดได้โดยใส่ ชื่อ constructor class แม่ ไว้ ดังนี้

```
46 class song : public media{
47 public :
48     string brand;
49
50     song(){ // default constructor
51         cout << "default constructor of [song] class" << endl;
52     }
53
54     song(string _name,string _author,int _length, string _brand) : media(_name,_author,_length){
55
56         cout << "4 parameter constructor of [song] class" << endl;
57         brand = _brand;
58
```

3_4.cpp

ดังนั้นเมื่อที่ main( ) เราสามารถสร้าง Object s1 จาก class song แล้วก็จะสามารถใช้งานได้ตามปกติ (มีการเรียก con.. ของคลาสแม่ตามลำดับ)

```
46 class song : public media{
47 public :
48     string brand;
49
50     song(){ // default constructor
51         cout << "default constructor of [song] class" << endl;
52     }
53
54     song(string _name,string _author,int _length, string _brand) : media(_name,_author,_length){
55
56         cout << "4 parameter constructor of [song] class" << endl;
57         brand = _brand;
58
```

main() {

```
song s1("Som San","sek loso",314,"LOSO");

cout << s1.name << endl;
cout << s1.author << endl;
cout << s1.length << endl;
cout << s1.brand << endl;
```

Result :

3 parameter constructor of [media] class

4 parameter constructor of [song] class

Som San

sek loso

314

LOSO

3_4.cpp

Class ลูกอื่นๆ ก็เช่นกัน สามารถส่งค่าไป Constructor class แม่ได้

```
movie(string _name,string _author,int _length, string _director, string _actor) : media(_name,_author,_length){
    cout << "default constructor of [movie] class" << endl;
    director = _director;
    main_actor = _actor;
}
```

```
episode(string _name,string _author,int _length, int ep_num, string ep_name) : media(_name,_author,_length){
    cout << "default constructor of [episode] class" << endl;
    episode_number = ep_num;
    episode_name = ep_name;
}
```

สรุป Constructor

เป็น function (method) ที่ถูกเรียกใช้ทุกครั้งที่มีการสร้าง object

- ใช้เพื่อการกำหนดค่าเริ่มต้นและตั้งค่าก่อนใช้ตัวแปร
- จะมีชื่อ method ที่ชื่อเหมือน struct หรือ class ของตนเอง
- สามารถมี parameter ได้
- Default constructor คือ constructor ที่ไม่รับ parameter หรือ มีแต่ default parameter
- Default constructor จะถูก call เสมอหากไม่มี การ call constructor อื่น

ให้นักศึกษาทดลองรันโปรแกรม 3_3, 3_4 ตรวจสอบคำตอบว่าตรงตามที่ตนคิดหรือไม่

ตัวอย่าง คำตอบ 3_4.cpp

```
3 parameter constructor of [media] class
default constructor of [media] class
4 parameter constructor of [song] class
3 parameter constructor of [media] class
5 parameter constructor of [movie] class
3 parameter constructor of [media] class
5 parameter constructor of [movie] class
3 parameter constructor of [media] class
5 parameter constructor of [episode] class
3 parameter constructor of [media] class
5 parameter constructor of [episode] class
Avatar James Cameron
Playing The Disappearance of Haruhi Suzumiya at[40] : [9667 sec] remaining
Playing Som San at[300] : [14 sec] remaining
Playing Timemachine  at[30] : [298 sec] remaining
```

Inheritance Modifier (โปรแกรม 3_5.CPP) (การสืบทอดแบบปรับเปลี่ยนพฤติกรรมบางอย่าง)

ปกติคลาสแม่หรือคลาสใดๆ เราสามารถกำหนดให้แต่ละส่วนมีการเข้าถึงได้ดังนี้

`private :`

- accessible only in the same class , เข้าถึงได้เฉพาะใน class เดียวกัน

`protected :`

- เข้าถึงจาก class เดียวกัน และ class ที่สืบทอดไป (class ลูกๆ)

`public :`

- accessible everywhere, เข้าถึงได้จากทุกที่

```cpp
class media{
private :
    int playing_sec; // last played seconds in episode

protected :
    string name;
    string author;
    int length; // media length in seconds

public :
    void play(int time){ // play method
        int remaining_time = length - playing_sec;
        cout << "Playing " << name << " at[" << playing_s
        if(time > remaining_time){
            playing_sec = 0;
        }
        else{
            playing_sec += time;
        }
    }

    int get_playing_sec(){
        return playing_sec;
    }

    void reset(){
        playing_sec = 0;
```

ไม่มีใครสามารถเปลี่ยน playing sec โดยตรงได้

Name ,author ,length สามารถเข้าถึงผ่านทาง class song ,movie ,episode ได้

Play(int) , get_playing_sec(), reset สามารถเข้าถึงได้จากทุกที่

3_5.cpp

เมื่อเรามีการสืบทอด ก็ยังสามารถกำหนดรูปแบบการกำหนดชนิดของคลาสที่เราจะถ่ายทอดได้ด้วยเช่นกันดังนี้

```
class [child_class_name] : [modifier1] [base_class_name1] , [modifier2] [base_class_name2] ...{
    // class component
}
```

```
class movie : public media{
public :
    string director;
    string main_actor;
};
```

```
class episode : private media{
public :
    int episode_number;
    string episode_name;
};
```

ปกติจะนิยมชนิด public

- **Public** inheritance is the most common usage

- การใช้งาน **Public** inheritance คือรูปแบบที่ใช้งานกันมากที่สุดและปกติที่สุด

```
class movie : public media{
public :
    string director;
    string main_actor;
};
```

public inheritance

protected => protected

public = > public

media

private:

float playing_sec;

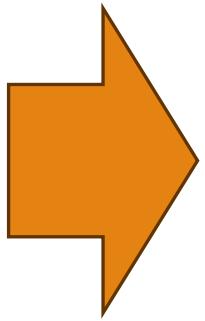
protected:

string name;

string authur;

public:

void play(int time)



movie : public media

private:

float number_sec;

protected:

Int length;

string name;

string authur;

public:

string director;

string main_actor;

void play(int time)

movie

private:

float number_sec;

protected:

Int length;

public:

string director;

string main_actor;

protected inheritance

protected => protected

public = > protected

media

private:

float playing_sec;

protected:

string name;

string authur;

public:

void play(int time)

movie : protected media

private:

float number_sec;

protected:

Int length;

string name;

string authur;

void play(int time)

public:

string director;

string main_actor;

movie

private:

float number_sec;

protected:

Int length;

public:

string director;

string main_actor;

private inheritance

protected => private

public = > private

media

private:

float playing_sec;

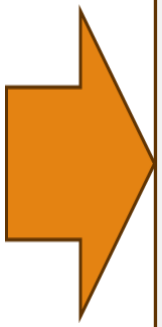
protected:

string name;

string authur;

public:

void play(int time)



movie : private media

private:

float number_sec;

float playing_sec;

string name;

string authur;

void play(int time)

protected:

Int length;

public:

string director;

string main_actor;

movie

private:

float number_sec;

protected:

Int length;

public:

string director;

string main_actor;

***ลองพิจารณาตัวอย่างนี้ (โปรแกรม 3_5.CPP)

```
49  class song : public media{
50  private :
51      string brand;
52
53  public :
54      song(){ // default constructor
55      }
56
57      song(string _name,string _author,int _length, string _brand) : media(_name,_author,_length){ // 4 parameter
58          brand = _brand;
59      }
60
61      void print_song(){
62          cout << "now we playing " << name << " of " << brand << " at " <<  get_playing_sec() << " sec" << endl;
63      }
64  };
```

3_5.cpp

หากเรากำหนด modified ใหม่เป็นดังนี้

```
class song : private media
{
private:
    string brand;

public:
    song()
    { // default constructor
    }
    song(string _name, string _author, int _length, string _brand) : media(_name, _author, _length)
    { // 4 parameter constructor
        brand = _brand;
    }
    void print_song()
    {
        cout << "now we playing " << name << " of " << brand << " at " << get_playing_sec() << " sec" << endl;
    }
};
```

```
5  class media{
6  private :
7      int playing_sec; // last played seconds in episode
8
9  protected :
10     string name;
11     string author;
12     int length; // media length in Seconds
13
14 public :
15     void play(int time){ // play method
16         int remaining_time = length - playing_sec;
17         cout << "Playing " << name << " at[" << playing_s
18         if(time > remaining_time){
19             playing_sec = 0;
20         }
21         else{
22             playing_sec += time;
23         }
24     }
25
26     int get_playing_sec(){
27         return playing_sec;
28     }
29
30     void reset(){
31         playing_sec = 0;
```

ไม่มีใครสามารถเปลี่ยน playing sec โดยตรงได้

ไปอยู่ส่วน private ของคลาส song

ไปอยู่ส่วน private ของคลาส song

3_5.cpp

การใช้งาน object s1 ก็จะต้องเปลี่ยนไป เพราะเรียกใช้ตรงๆไม่ได้อีกแล้ว

```
int main()
{
    song s1("Som San", "sek loso", 314, "LOSO");
    s1.play(100);
    cout << s1.name<<endl;
    return 0;
}
```

แก้ไขโดยเพิ่ม method ที่ต้องการเข้าไปใน คลาสลูก (class song)

```
class song : private media
{
private:
    string brand;
public:
    song()
    { // default constructor
    }
    song(string _name, string _author, int _length, string _brand) : media(_name, _author, _length)
    { // 4 parameter constructor
        brand = _brand;
    }
    void song_play(int time)
    {
        play(time);
    }
    string get_name()
    {
        return name;
    }
};
```

ก็สามารถใช้งานตัวแปรหรือmethod ที่เราต้องการได้

```cpp
int main()
{
    song s1("Som San", "sek loso", 314, "LOSO");
    s1.song_play(100);
    cout << s1.get_name() <<endl;
    return 0;
}
```

```
Playing Som San at[100] : [214 sec] remaining
Som San
```

นักศึกษาลองดู Class แม่ อื่นบ้าง.. (โปรแกรม 3_6.CPP) ลองรันแล้วเติมช่องว่างให้ถูกต้องตามรูป

Base class :

```
5  class vehicle{
6  protected :
7      int speed;
8      int wheel_count;
9
10     vehicle(int wheel){
11         wheel_count = wheel;
12         speed = 0;
13     }
14
15 public :
16     void print_wheel(){
17         cout << "this vehicle has [" <<  wheel_count << "] Wheel(s)" << endl;
18     }
19 };
20
```

3_6.cpp

```
24  class motorbike : public vehicle
25  {
26  public:
27      motorbike() : vehicle(2)
28      {  }
29      void print()
30      {
31          cout << "motobike is using speed [" << speed << "]Kph" << endl;
32      }
33  };
```

```
35  class airplane : public vehicle
36  {
37      int altitude;
38  public:
39      void print()
40      {
41          cout << "airplane is at [" << altitude << "] ft above sealevel using speed [" << speed << "]Kph" << endl;
42      }
43      airplane(int wheel) : vehicle(wheel)
44      {
45          altitude = 0;
46      }
47  };
```

```
53  class submarine : public vehicle
54  {
55      int depth;
56  public:
57      void print()
58      {
59          cout << "submarine is at [" << depth << "] depth level using speed [" << speed << "]Kph" << endl;
60      }
61      submarine() : vehicle(0)
62      {
63          depth = 0;
64      }
65  };
```

```
motorbike vespa_sprint;
airplane a380(22);
submarine S26T;

vespa_sprint.print_wheel();
a380.                    ;
S26T.                    ;

vespa_sprint.print();
a380.                ;
S26T.                ;
```

```
this vehicle has [2] Wheel(s)
this vehicle has [22] Wheel(s)
this vehicle has [0] Wheel(s)
motobike is using speed [0]Kph
airplane is at [0] ft above sealevel using speed [0]Kph
submarine is at [0] depth level using speed [0]Kph
```

LAB

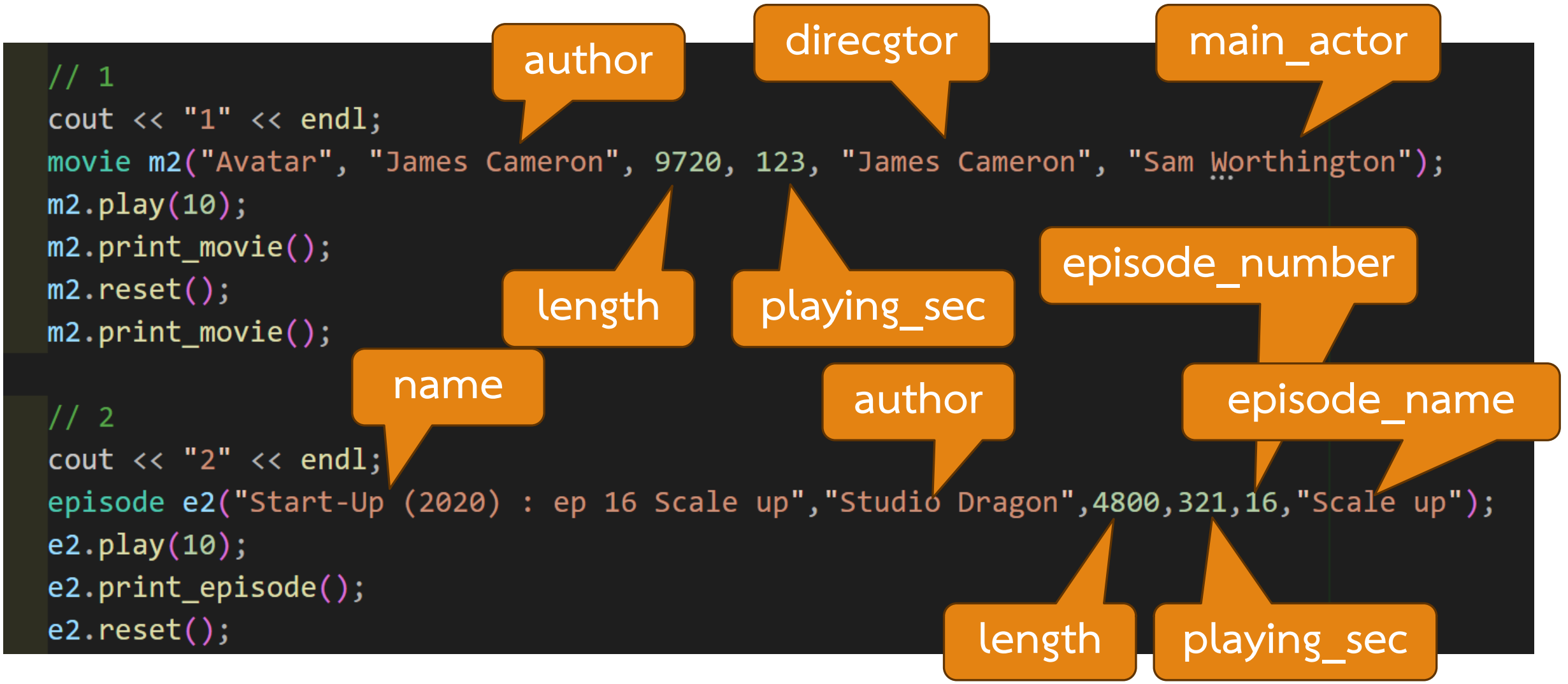
ให้นักศึกษาแก้ไขโปรแกรม 3_7_LAB.CPP เพื่อให้ได้ผลตามโจทย์

ต้องการแล้วส่ง โปรแกรม 3_6.CPP , 3_7.CPP ผ่าน MSTEAMS

ข้อกำหนด แก้ไขไฟล์ 3_7.CPP

- ห้ามแปลี่ยน modifier (public, private, protected) ของไฟล์ 3_7.cpp
- ห้ามเปลี่ยน main( )
- นอกนั้นทำได้ทุกอย่าง เขียนเพิ่มที่ไหนของไฟล์ก็ได้

ความหมายตัวแปร ในโปรแกรม 3_7.cpp




```cpp
// 1
cout << "1" << endl;
movie m2("Avatar", "James Cameron", 9720, 123, "James Cameron", "Sam Worthington");
m2.play(10);
m2.print_movie();
m2.reset();
m2.print_movie();

// 2
cout << "2" << endl;
episode e2("Start-Up (2020) : ep 16 Scale up","Studio Dragon",4800,321,16,"Scale up");
e2.play(10);
e2.print_episode();
e2.reset();
```