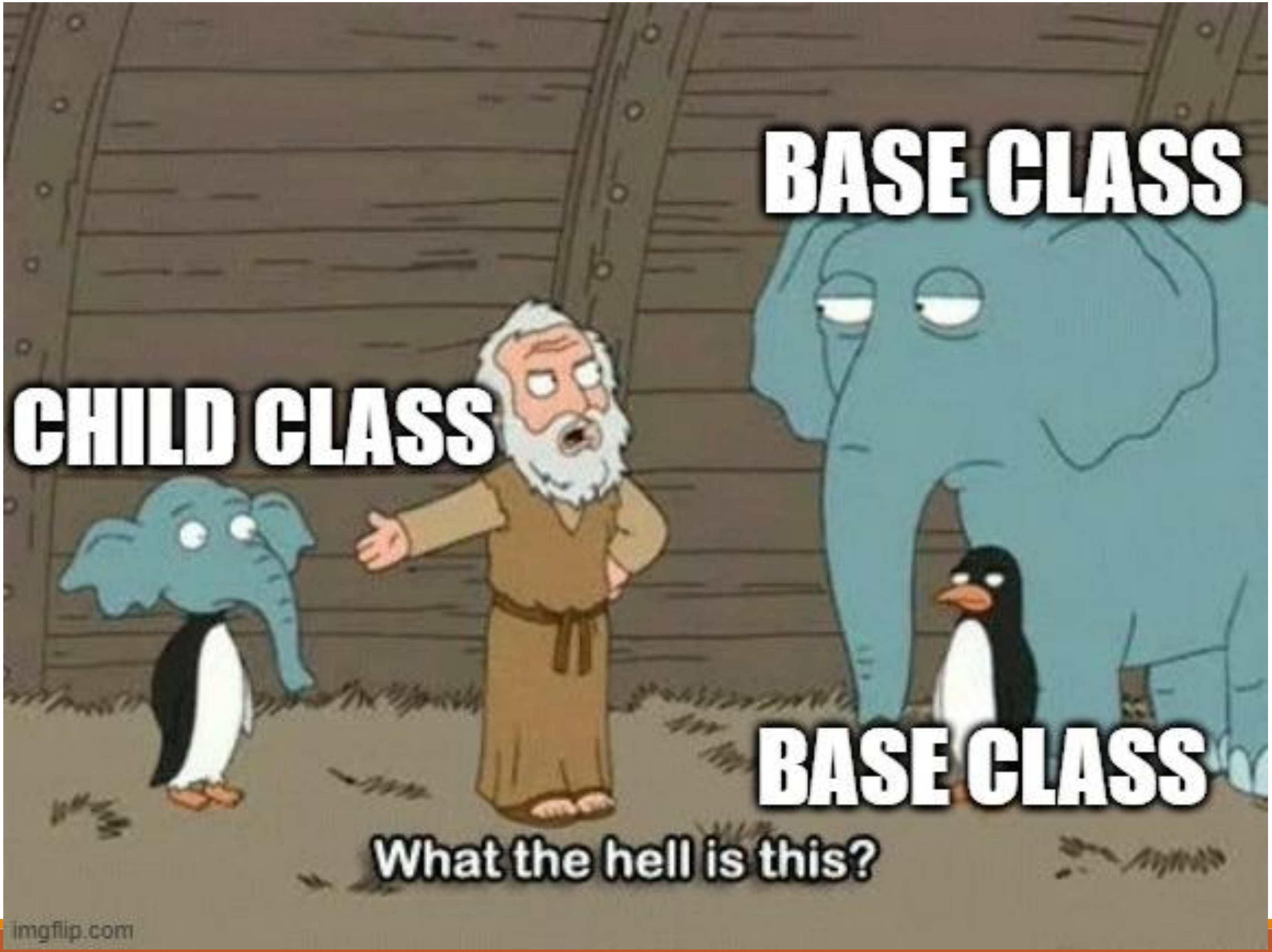
OOP & data struct

3. Inheritance

BY SOMSIN THONGKRAIRAT



BASE CLASS
CHILD CLASS
BASE CLASS
What the hell is this?
imgflip.com

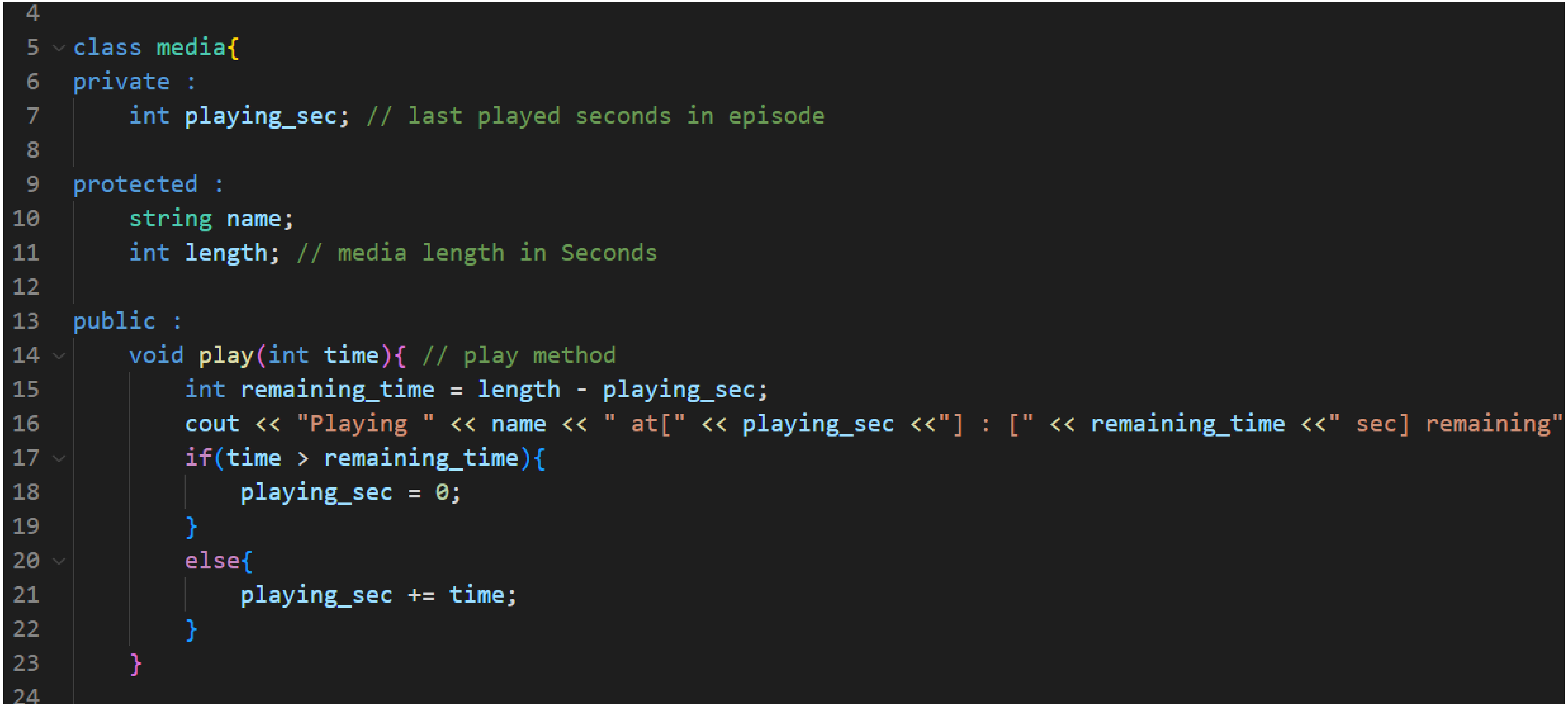
Class Con.

Media class (1 media)

Properties

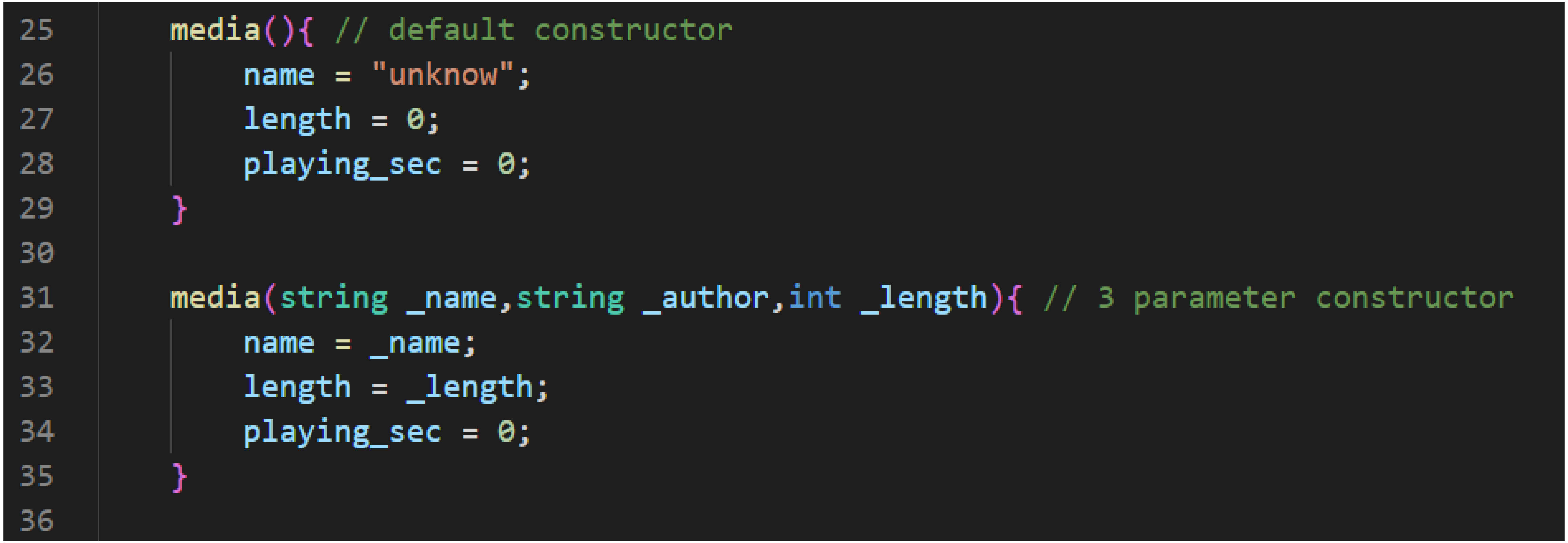
- Public -> name, length // ชื่อ, ความยาว (sec)
- Private -> playing_sec // เวลาล่าสุดที่เล่น (pause)

Method – play(int)



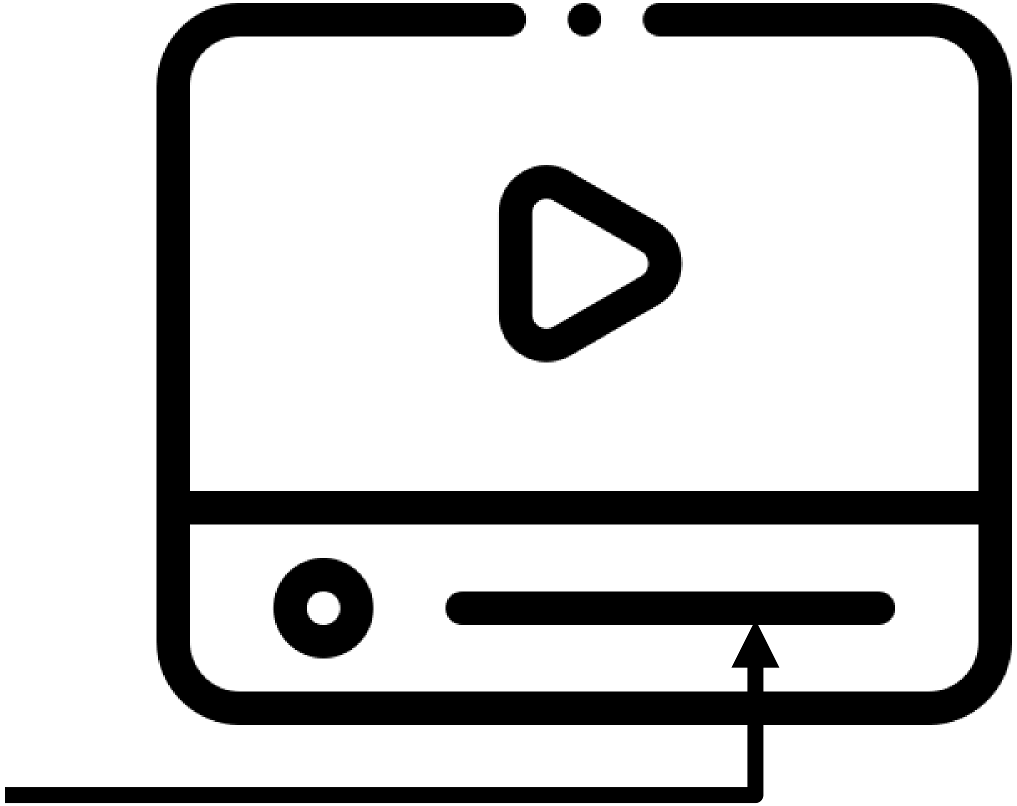
```cpp
class media{
private :
    int playing_sec; // last played seconds in episode

protected :
    string name;
    int length; // media length in Seconds

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

```

Media class Constructor

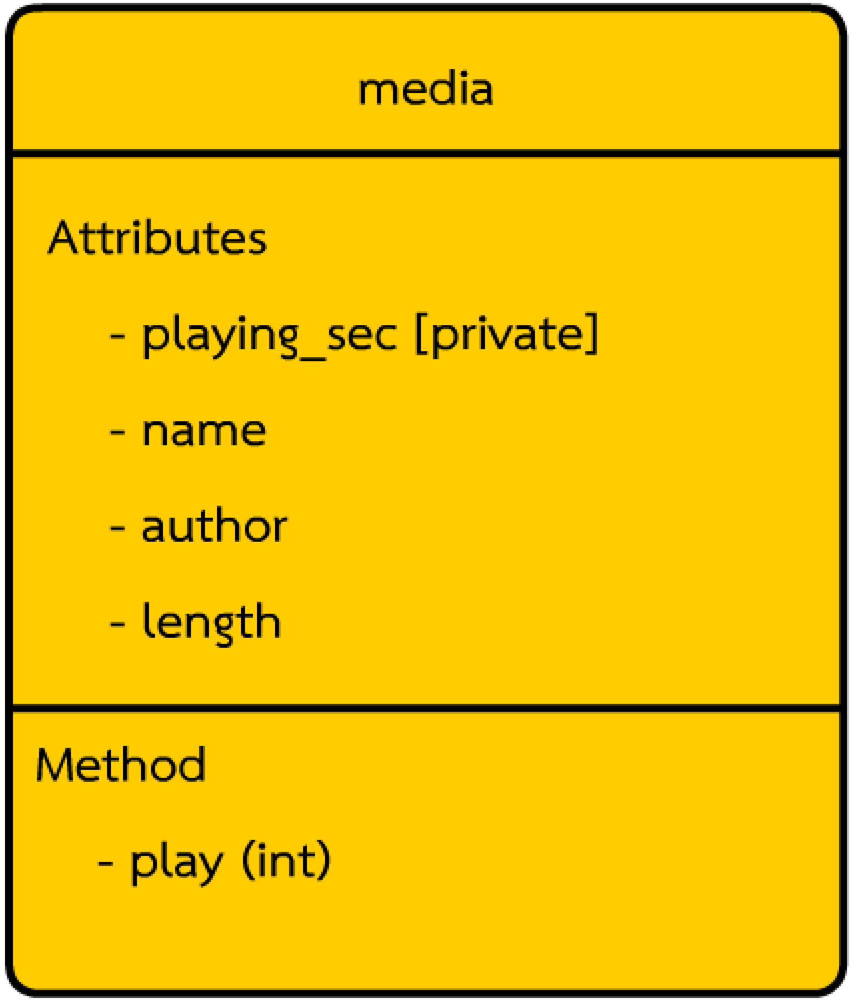


```
media(){ // default constructor
    name = "unknow";
    length = 0;
    playing_sec = 0;
}

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

```

Media Class



Playing_sec



media

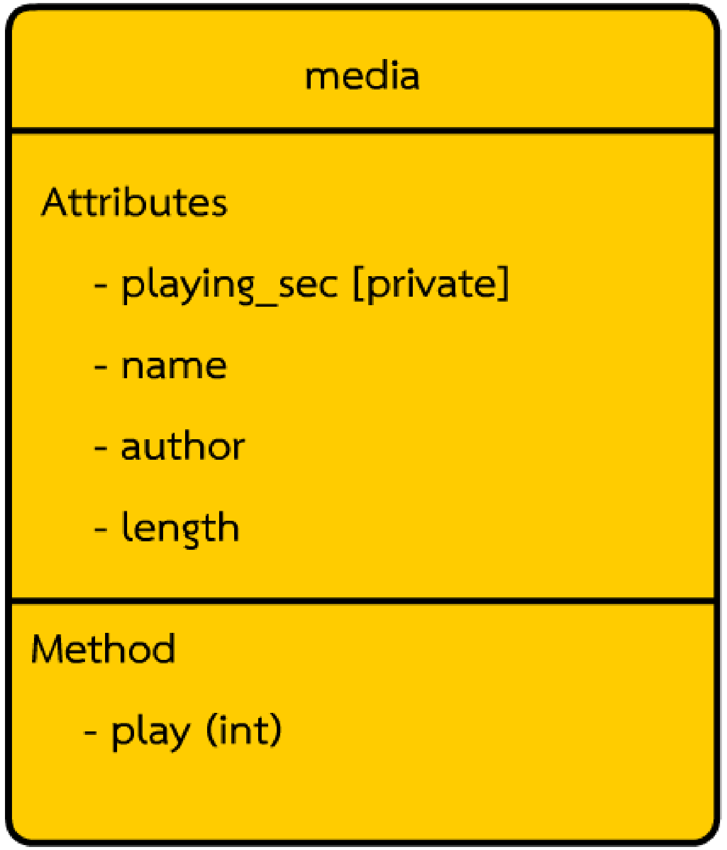
Attributes

- playing_sec [private]
- name
- author
- length

Method

- play (int)

Everything look good!



media

Attributes

- playing_sec [private]
- name
- author
- length

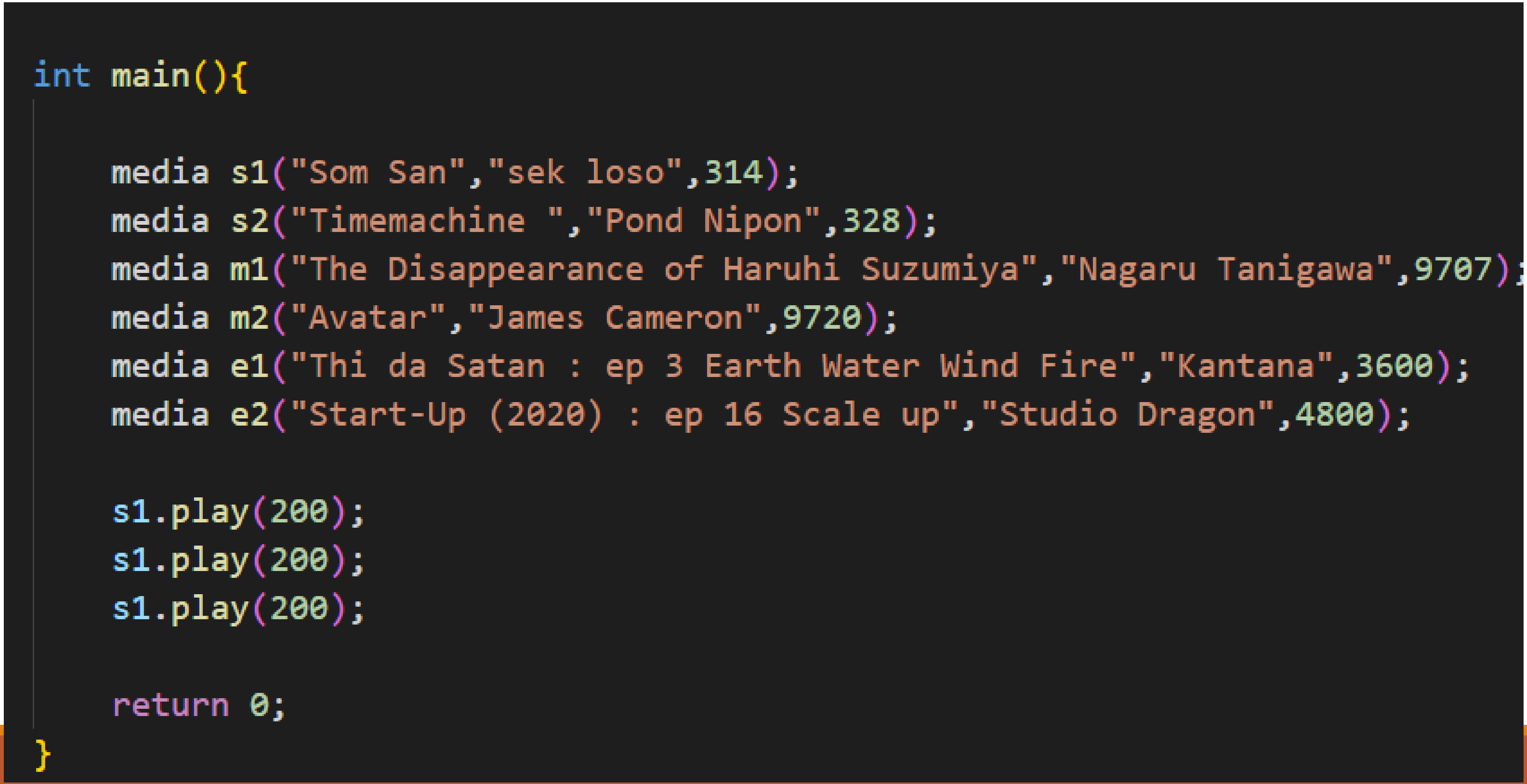
Method

- play (int)

Can handle

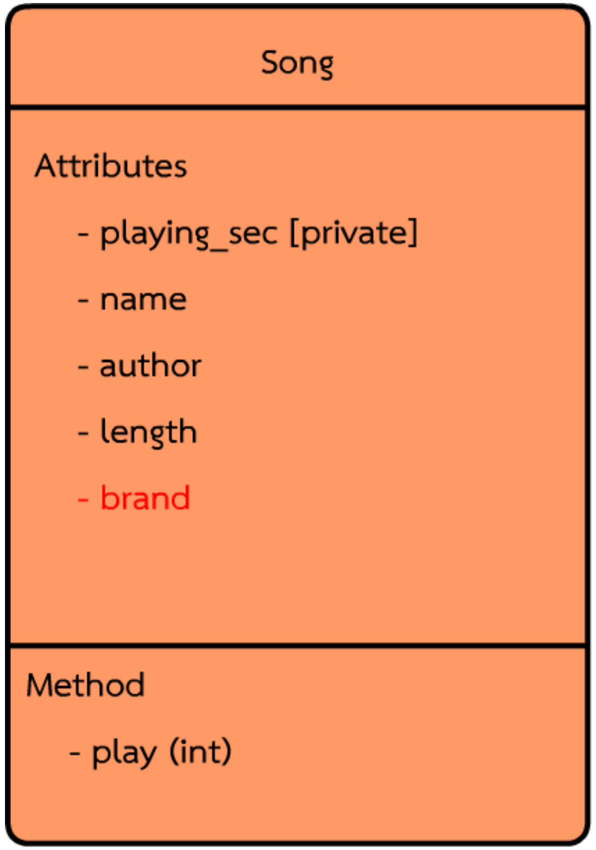
- Movie
- Song
- Episode

One class handle every classes



```
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);

    s1.play(200);
    s1.play(200);
    s1.play(200);

    return 0;
}
```

Make it more specific



Song

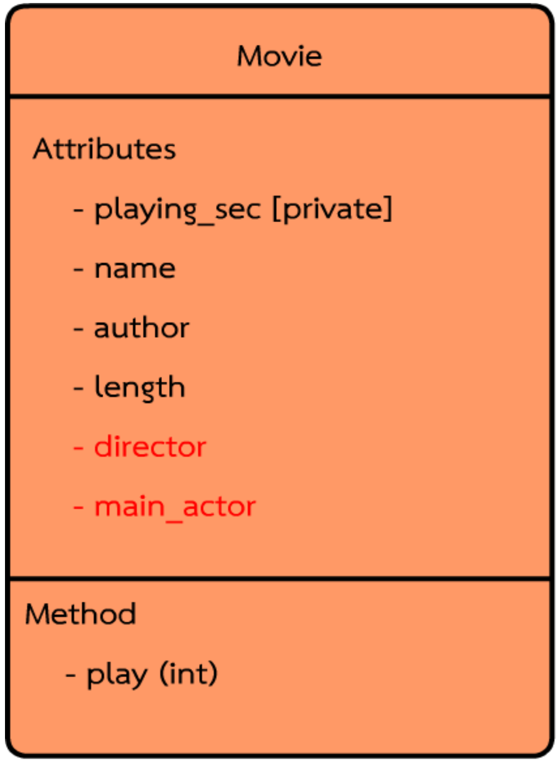
Attributes

- playing_sec [private]
- name
- author
- length
- brand

Method

- play (int)

Add Brand



Movie

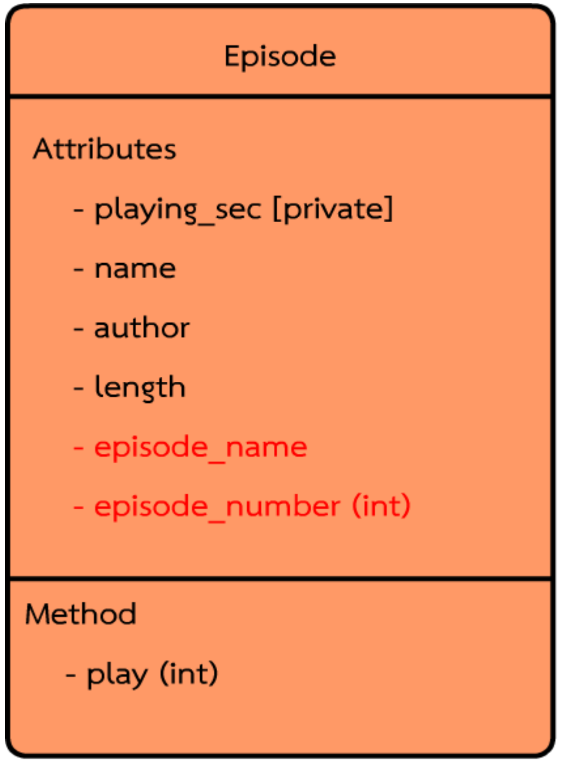
Attributes

- playing_sec [private]
- name
- author
- length
- director
- main_actor

Method

- play (int)

Add director , actor



Episode

Attributes

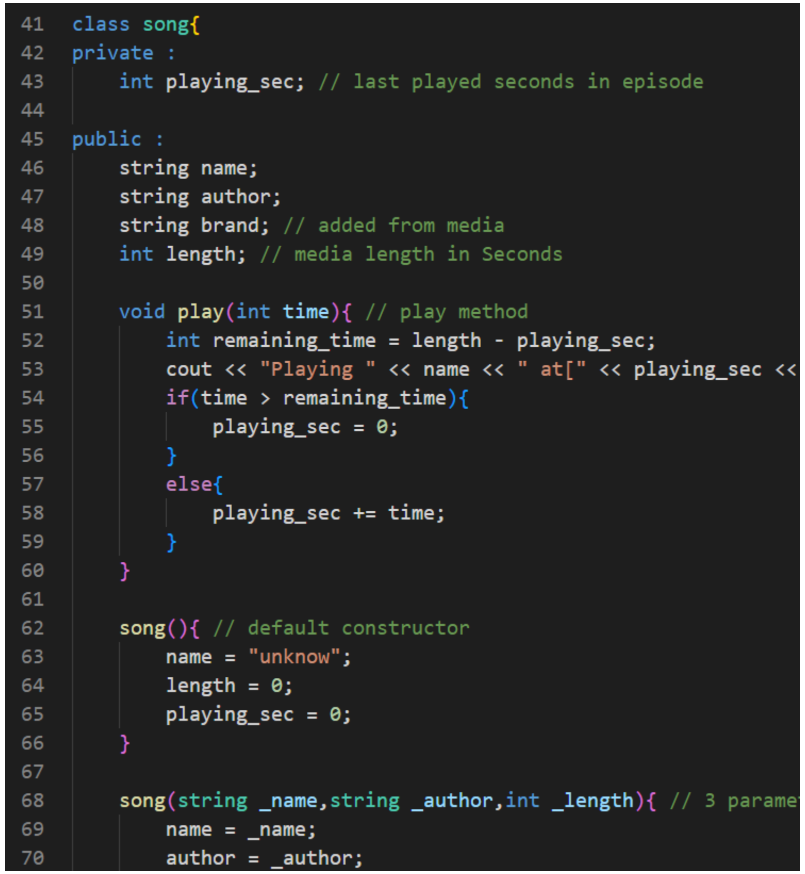
- playing_sec [private]
- name
- author
- length
- episode_name
- episode_number (int)

Method

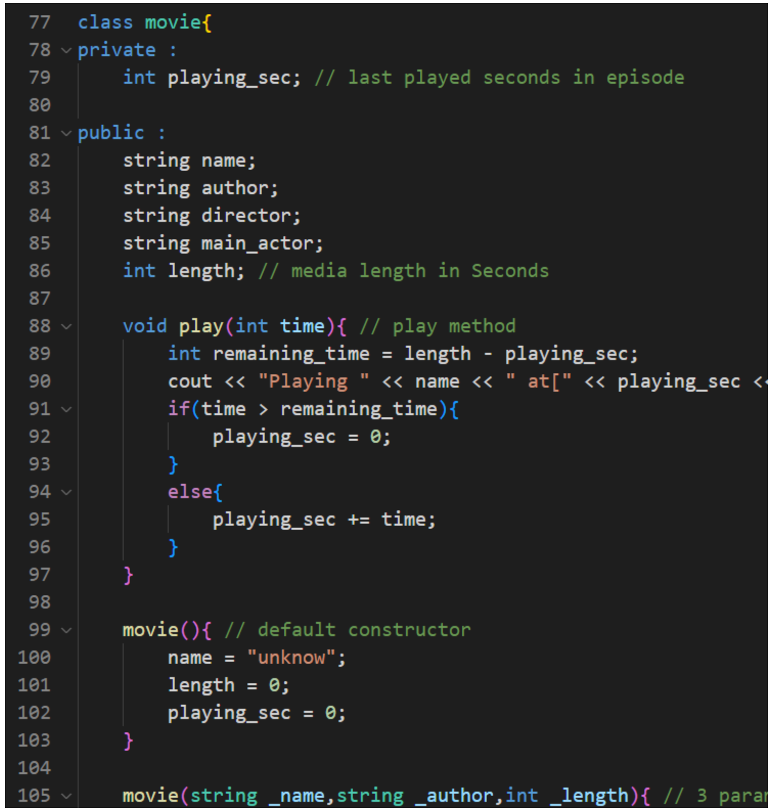
- play (int)

Add episode name and num

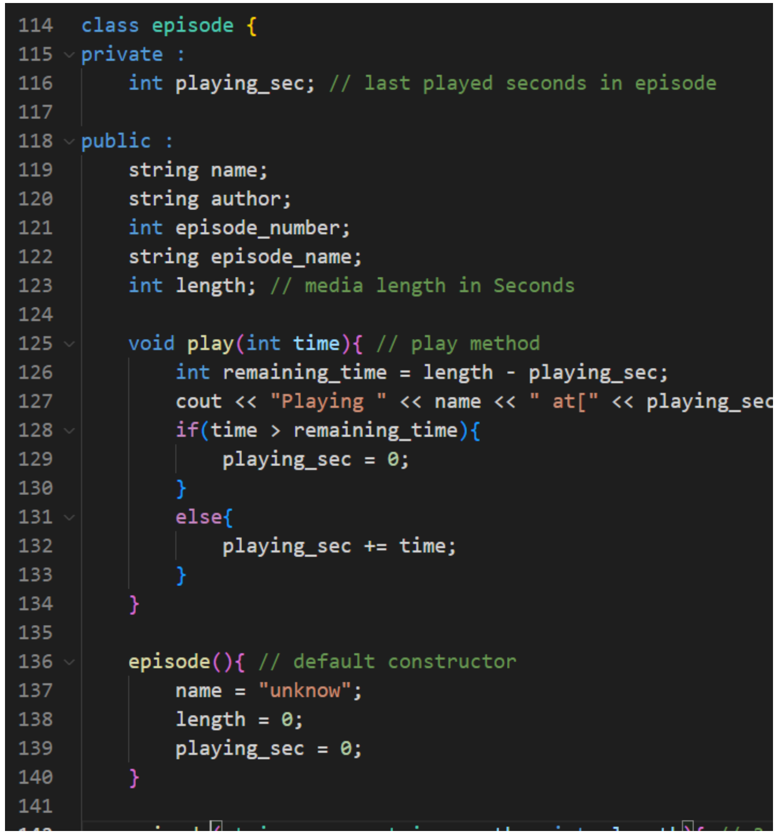
Hard code method (create 3 Classes)



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

public :
    string name;
    string author;
    string brand; // added from media
    int length; // media length in Seconds

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

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

    song(string _name,string _author,int _length){ // 3 parame
        name = _name;
        author = _author;
```



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

public :
    string name;
    string author;
    string director;
    string main_actor;
    int length; // media length in Seconds

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

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

    movie(string _name,string _author,int _length){ // 3 para
```



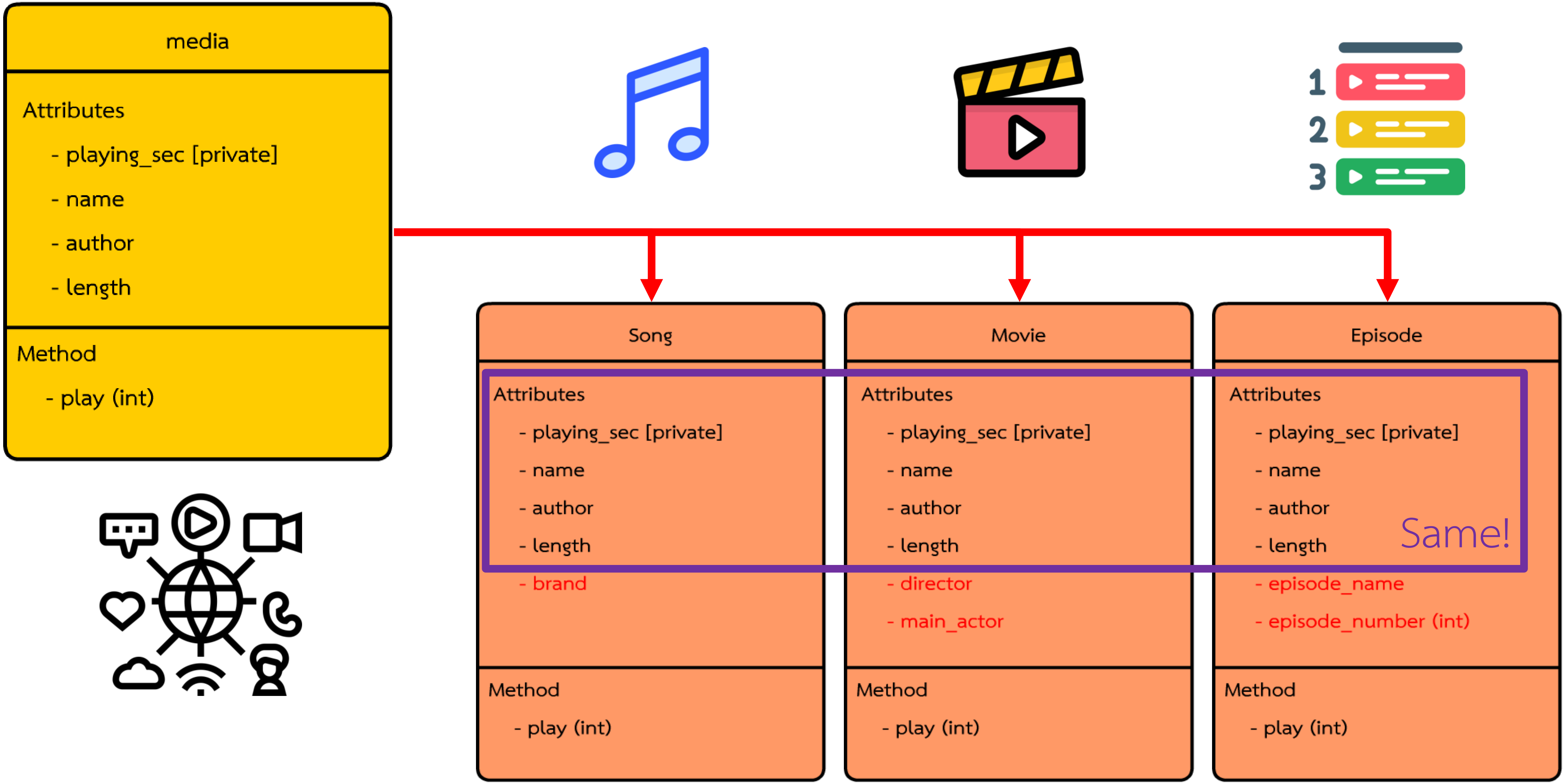
```cpp
class episode {
private :
    int playing_sec; // last played seconds in episode

public :
    string name;
    string author;
    int episode_number;
    string episode_name;
    int length; // media length in Seconds

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

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

At less it work!

Inheritance method

- Transfer class component to child class

- ส่งต่อส่วนประกอบไปยัง class อื่น (คลาสลูก)



media

Attributes
- playing_sec [private]
- name
- author
- length

Method
- play (int)

Song

Attributes
- playing_sec [private]
- name
- author
- length
- brand

Method
- play (int)

Movie

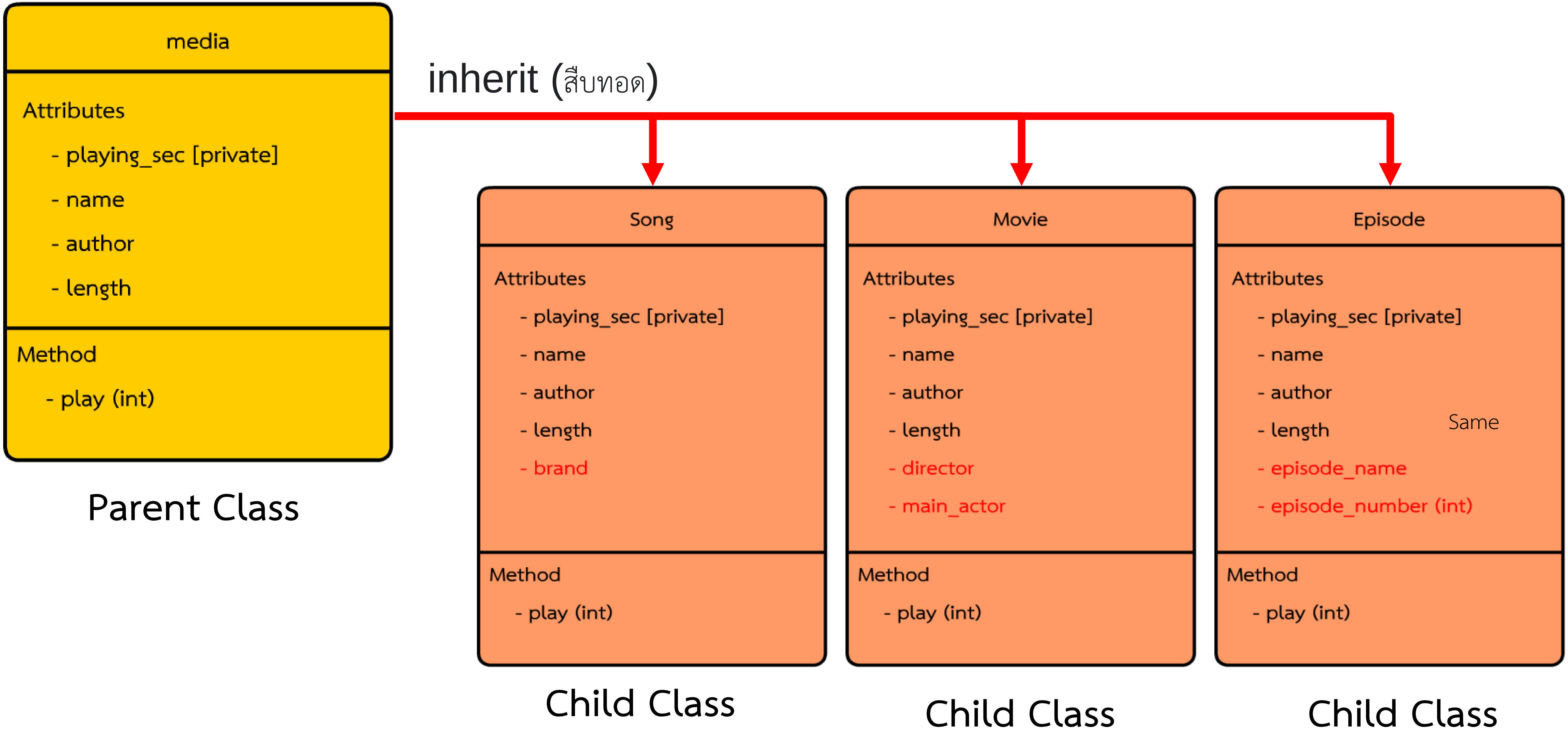
Attributes
- playing_sec [private]
- name
- author
- length
- director
- main_actor

Method
- play (int)

Episode

Attributes
- playing_sec [private]
- name
- author
- length
- episode_name
- episode_number (int)

Method
- play (int)



media

Attributes

- playing_sec [private]
- name
- author
- length

Method

- play (int)

Parent Class

inherit (สืบทอด)

Song

Attributes

- playing_sec [private]
- name
- author
- length
- brand

Method

- play (int)

Child Class

Movie

Attributes

- playing_sec [private]
- name
- author
- length
- director
- main_actor

Method

- play (int)

Child Class

Episode

Attributes

- playing_sec [private]
- name
- author
- length
- episode_name
- episode_number (int)

Same

Method

- play (int)

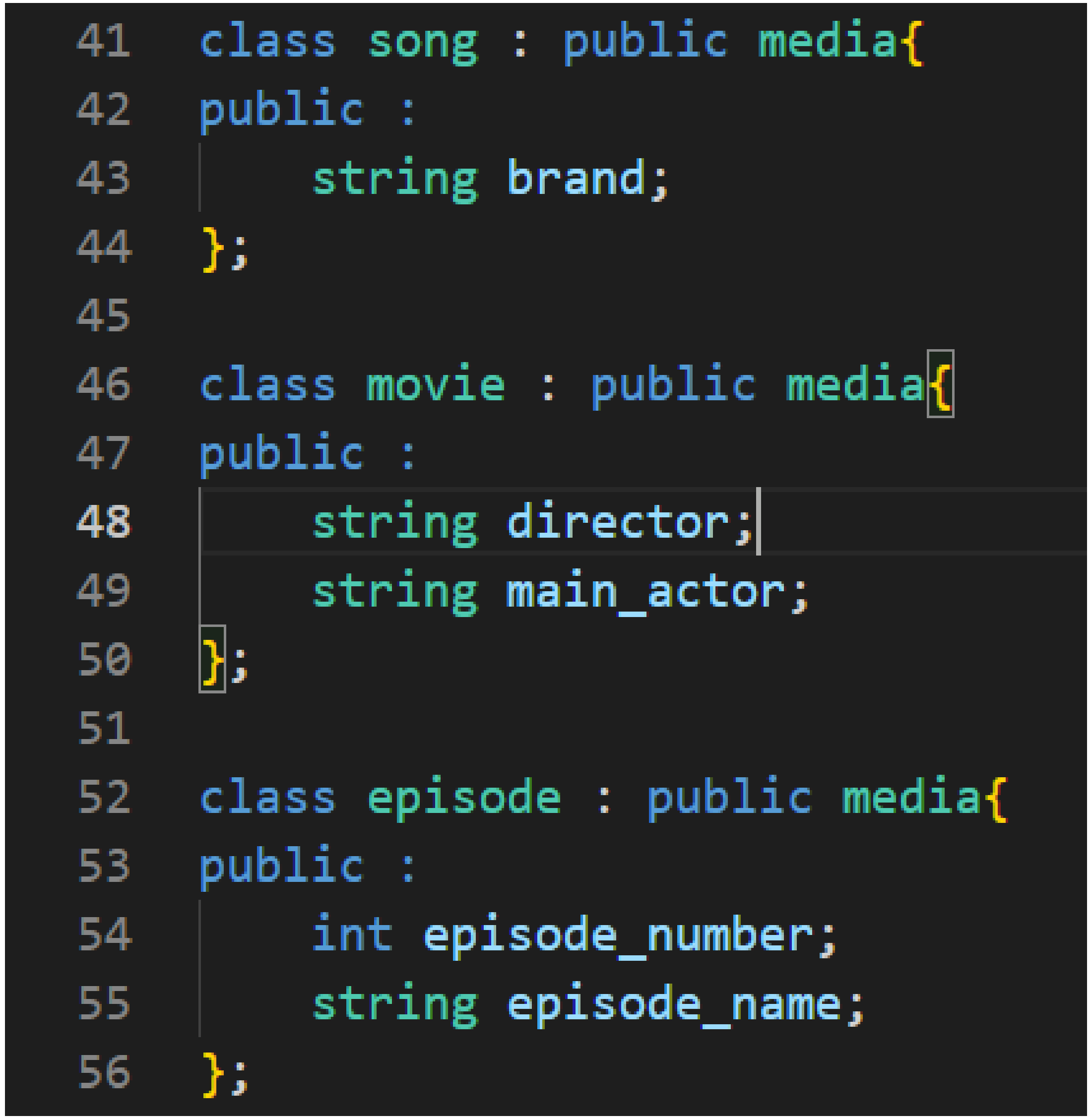
Child Class

C++ Syntax

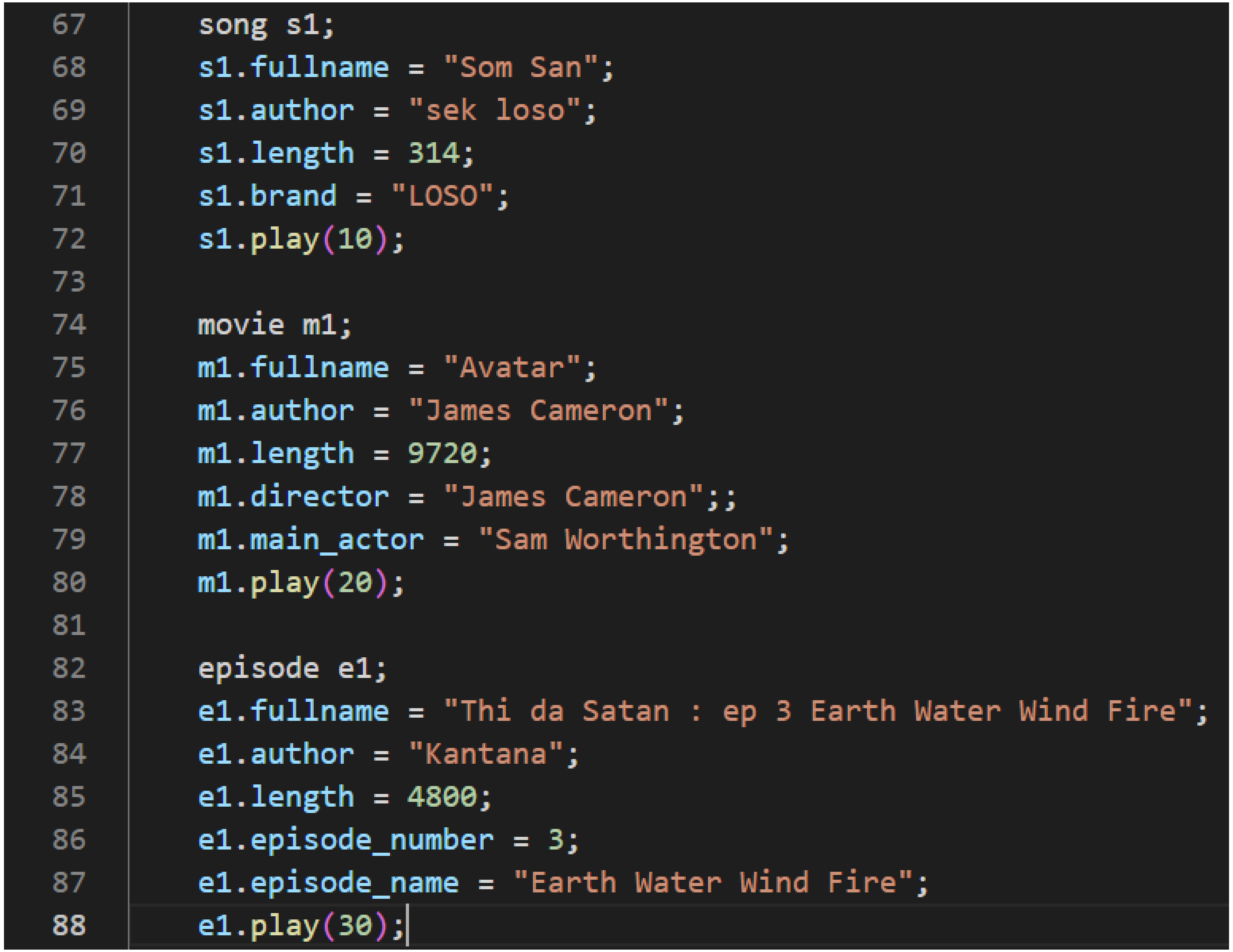
```
class [child_class_name] : [modifier1] [base_class_name1] , [modifier2] [base_class_name2] ...{

    // class component

}
```

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

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

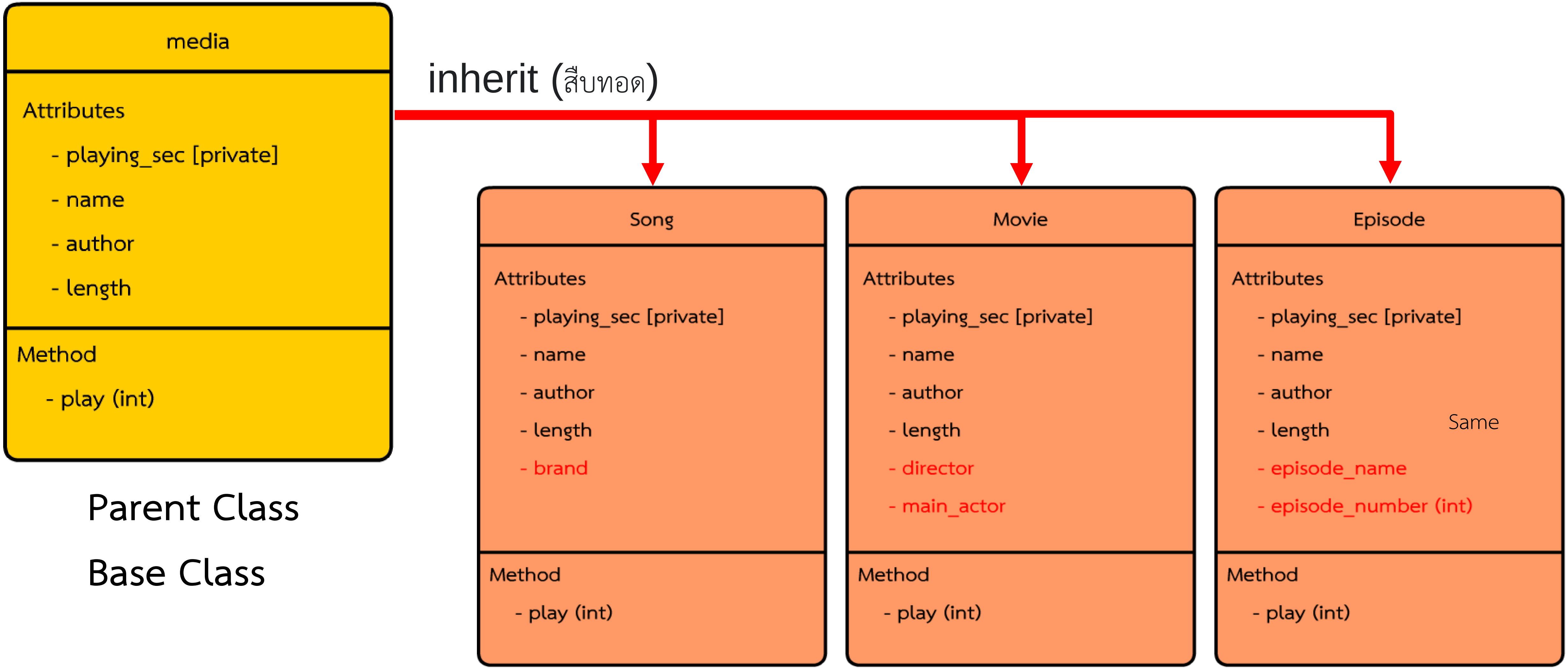


```
class song : public media{
public :
    string brand;
};

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

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



```
    song s1;
    s1.fullname = "Som San";
    s1.author = "sek loso";
    s1.length = 314;
    s1.brand = "LOSO";
    s1.play(10);

    movie m1;
    m1.fullname = "Avatar";
    m1.author = "James Cameron";
    m1.length = 9720;
    m1.director = "James Cameron";;
    m1.main_actor = "Sam Worthington";
    m1.play(20);

    episode e1;
    e1.fullname = "Thi da Satan : ep 3 Earth Water Wind Fire";
    e1.author = "Kantana";
    e1.length = 4800;
    e1.episode_number = 3;
    e1.episode_name = "Earth Water Wind Fire";
    e1.play(30);
```

All can use play();



inherit (สืบทอด)

media

Attributes

- playing_sec [private]
- name
- author
- length

Method

- play (int)

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)

Same

Method

- play (int)

Parent Class

Base Class

Child Class

Derived Class

Child Class

Derived Class

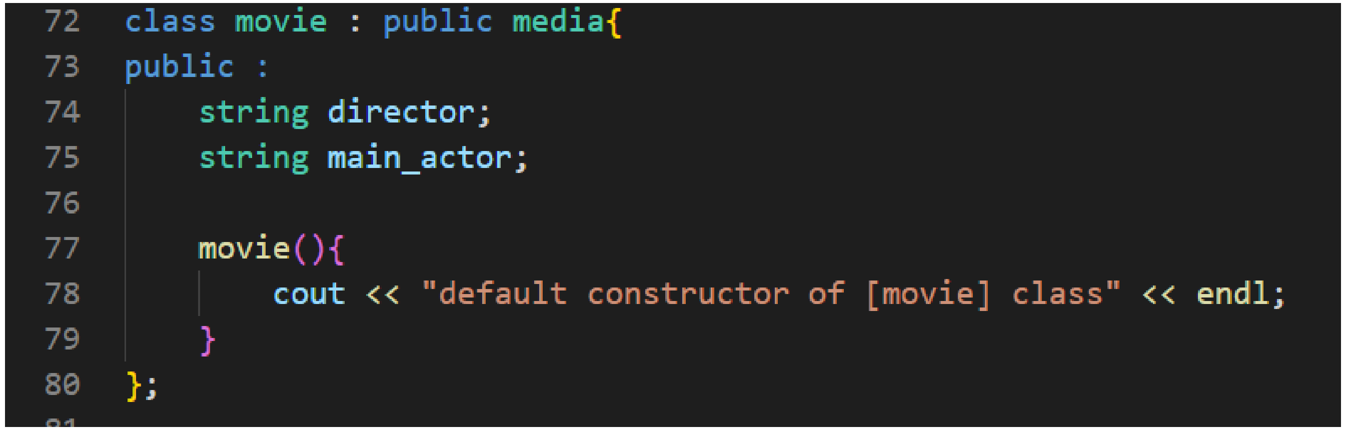
Child Class

Derived Class

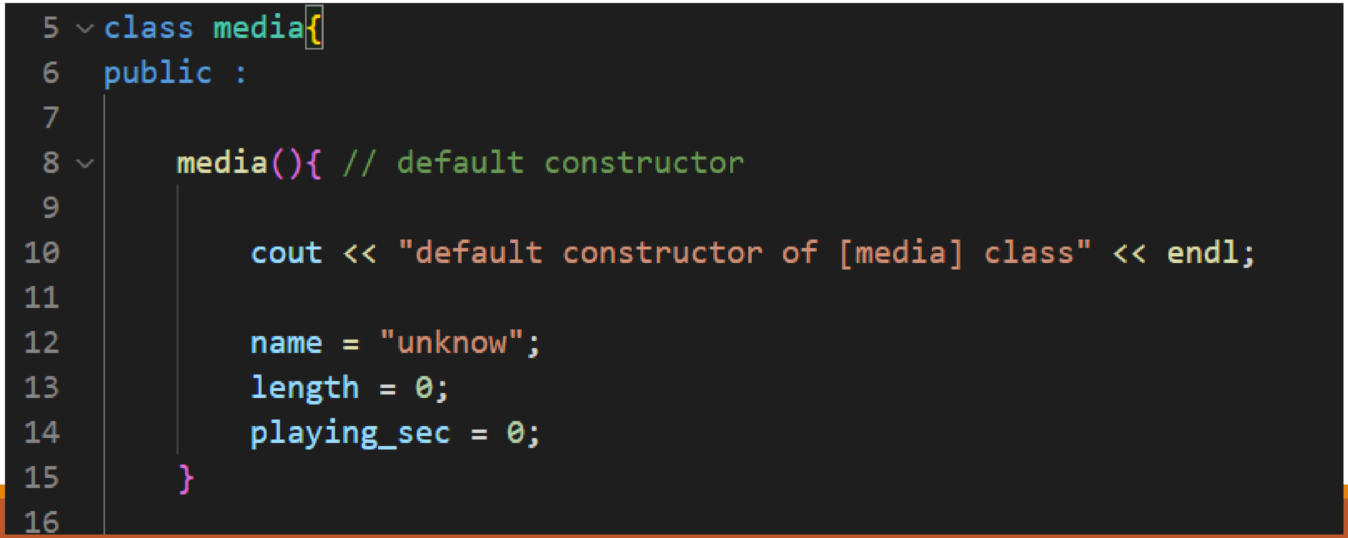
Constructor

QUIZ

Constructor (default constructor)

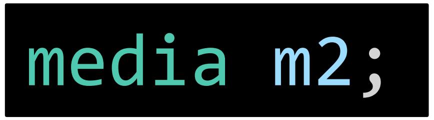


```
class movie : public media{
public :
    string director;
    string main_actor;

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



```
class media{
public :

    media(){ // default constructor

        cout << "default constructor of [media] class" << endl;

        name = "unknow";
        length = 0;
        playing_sec = 0;
    }

```

Code:

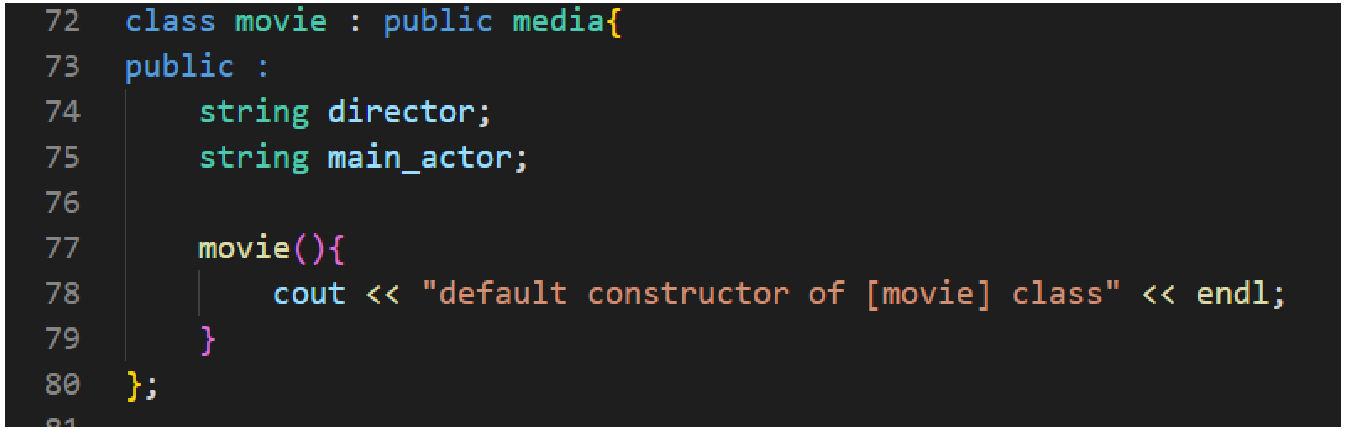


```
media m2;
```

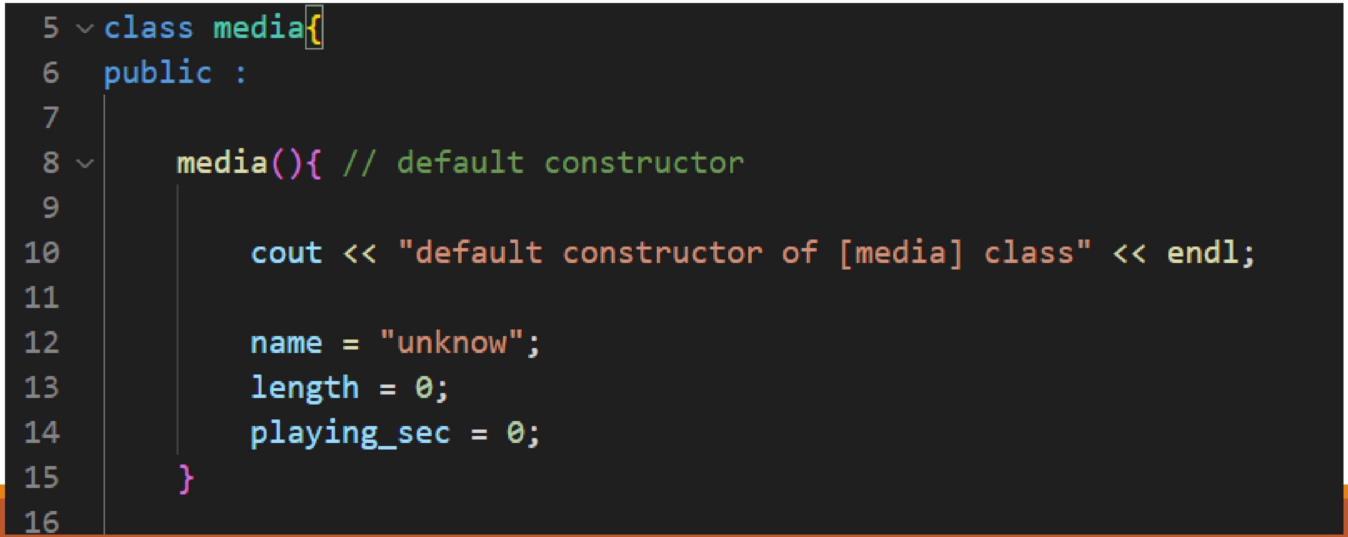
Result:

```
default constructor of [media] class
```

Constructor (default constructor)

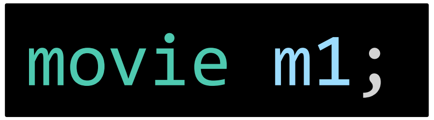


```
class movie : public media{
public :
    string director;
    string main_actor;

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



```
class media{
public :

    media(){ // default constructor

        cout << "default constructor of [media] class" << endl;

        name = "unknow";
        length = 0;
        playing_sec = 0;
    }

```

Code:

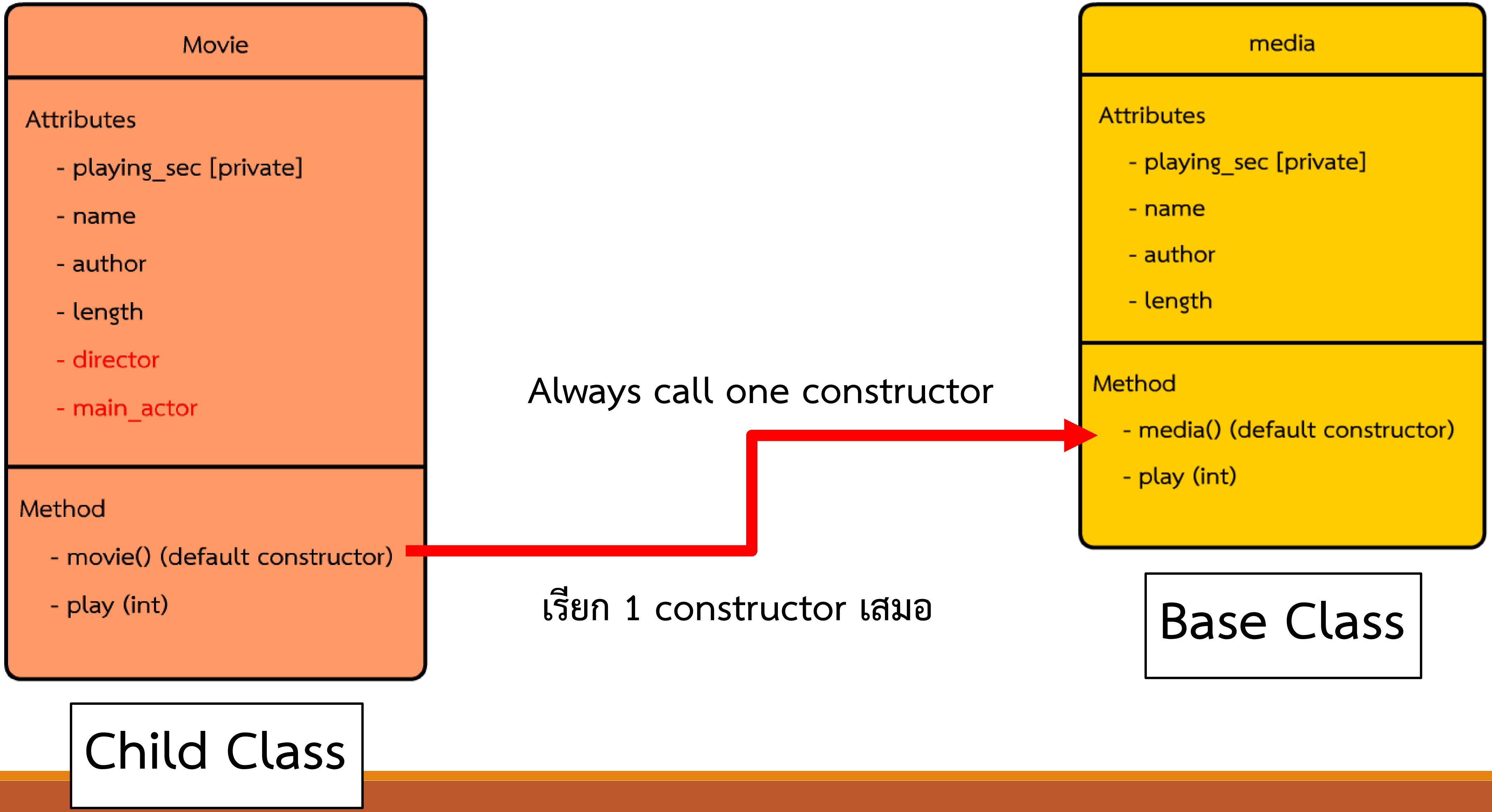


```
movie m1;
```

Result:

```
default constructor of [media] class
default constructor of [movie] class
```

Constructor (default constructor)



Movie

Attributes

- playing_sec [private]
- name
- author
- length
- director
- main_actor

Method

- movie() (default constructor)
- play (int)

Child Class

Always call one constructor

เรียก 1 constructor เสมอ

media

Attributes

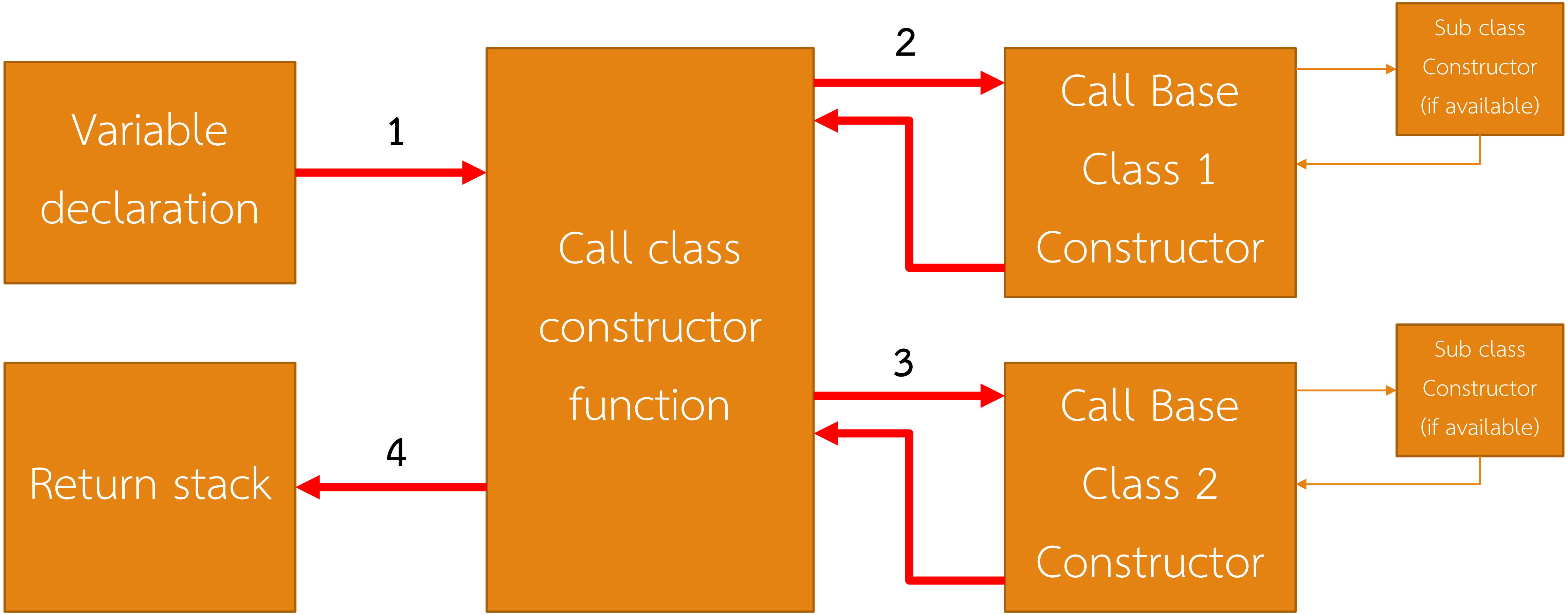
- playing_sec [private]
- name
- author
- length

Method

- media() (default constructor)
- play (int)

Base Class

Constructor



Variable declaration

1

Call class constructor function

2

Call Base Class 1 Constructor

Sub class Constructor (if available)

3

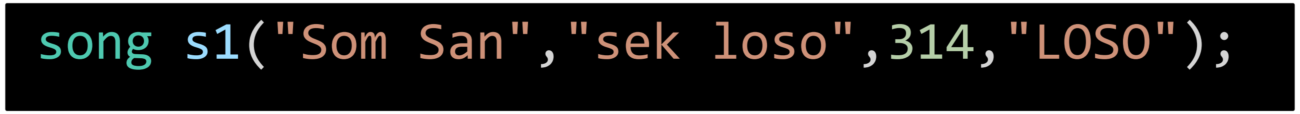
Call Base Class 2 Constructor

Sub class Constructor (if available)

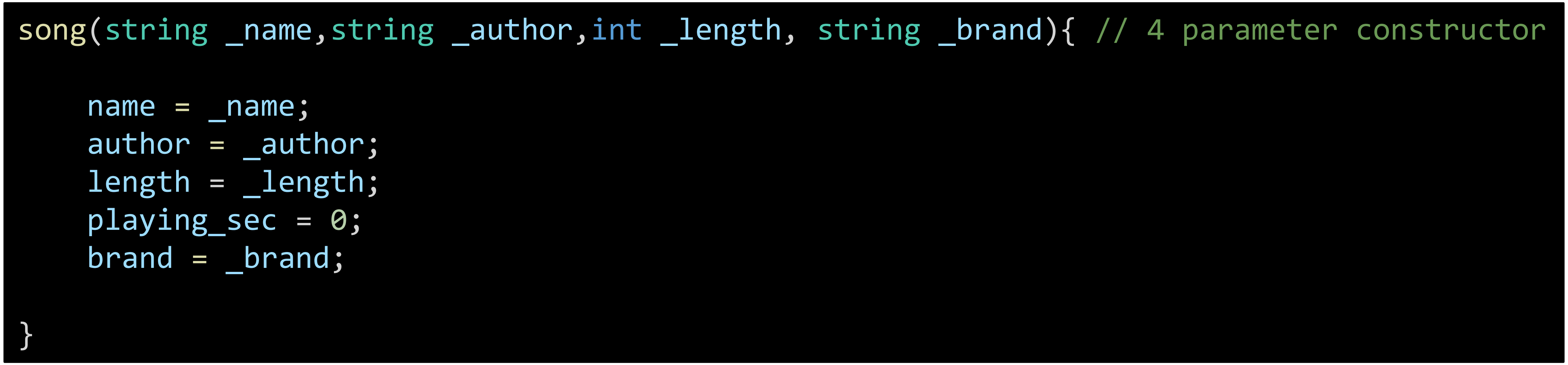
4

Return stack

Constructor



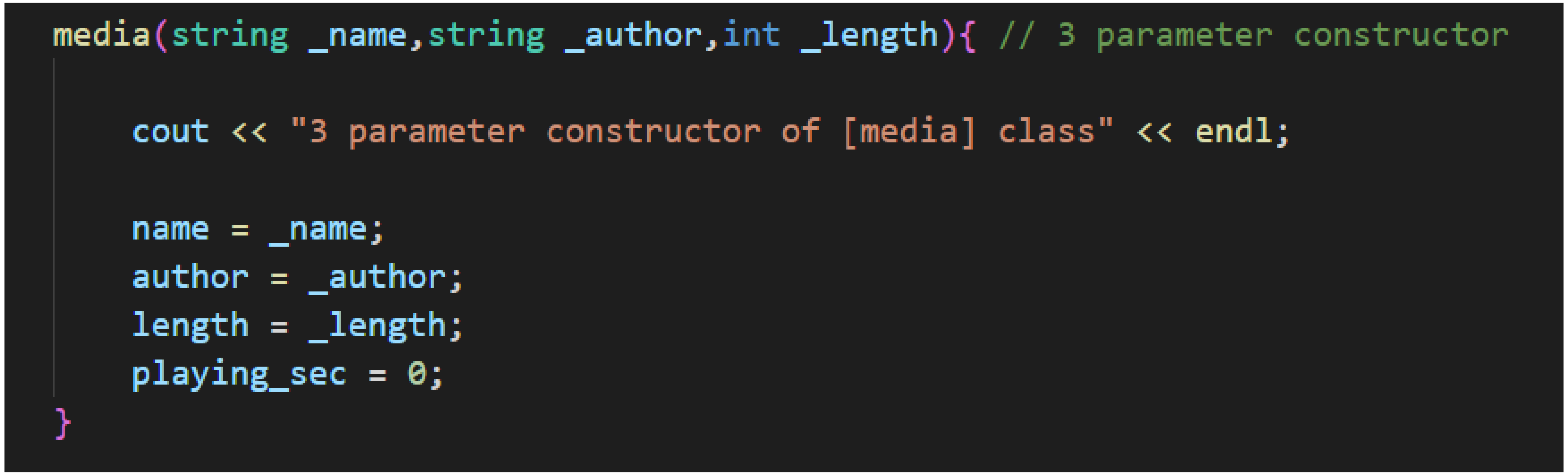
```
song s1("Som San","sek loso",314,"LOSO");
```



```
song(string _name,string _author,int _length, string _brand){ // 4 parameter constructor

    name = _name;
    author = _author;
    length = _length;
    playing_sec = 0;
    brand = _brand;

}
```

But can we use existed media constructor? / แล้วสามารถใช้ constructor ของ media ได้หรือไม่

Base constructor

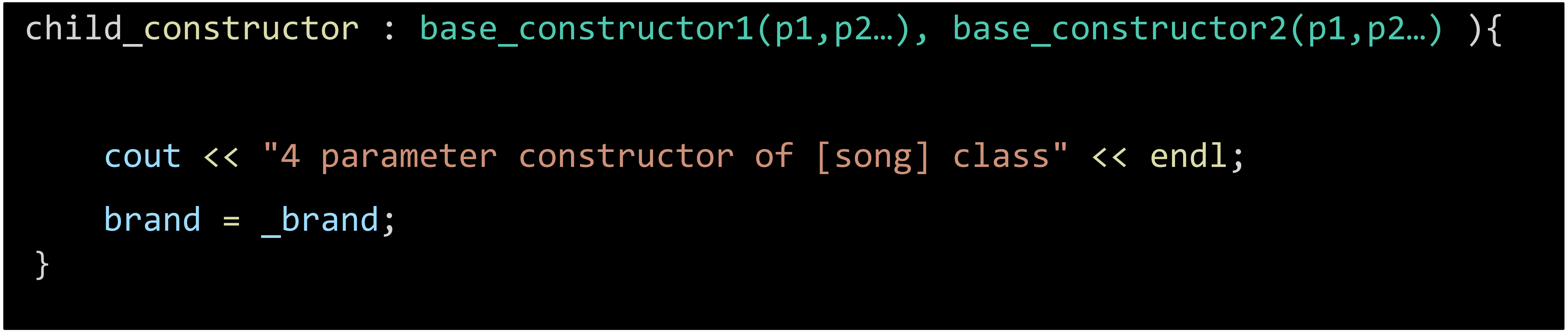


```
media(string _name,string _author,int _length){ // 3 parameter constructor

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

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

Can we use this constructor in **song** class?

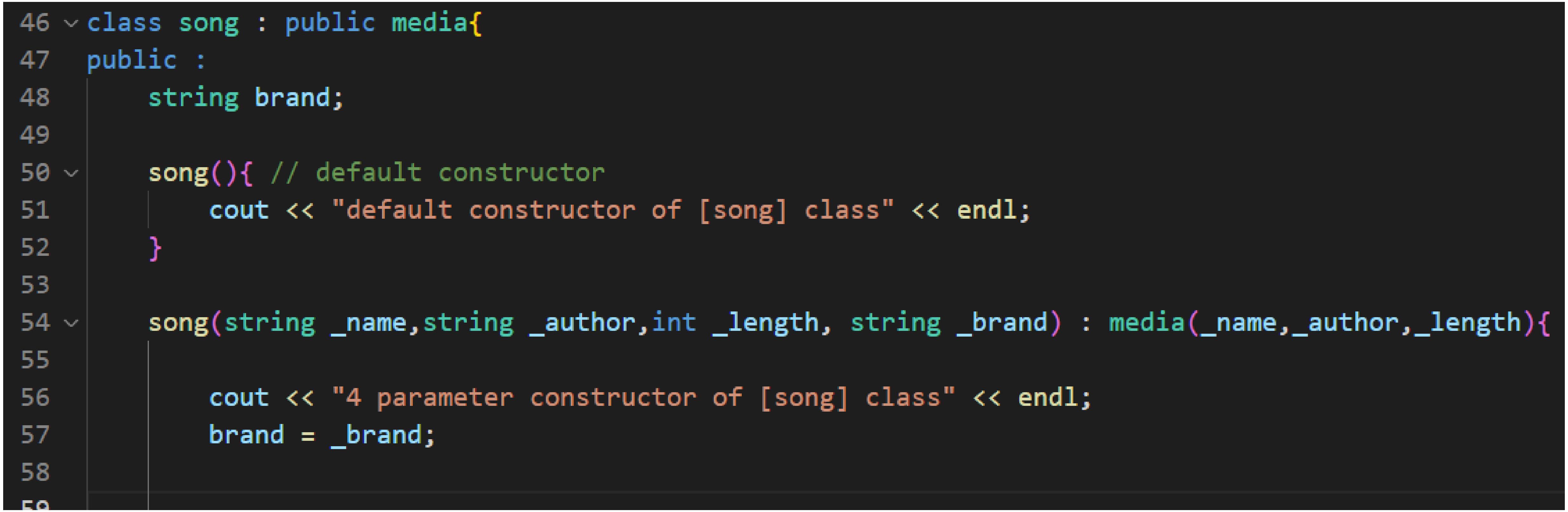
เราสามารถใช้ constructor นี้ใน class **song** ได้หรือไม่?

Base constructor syntax



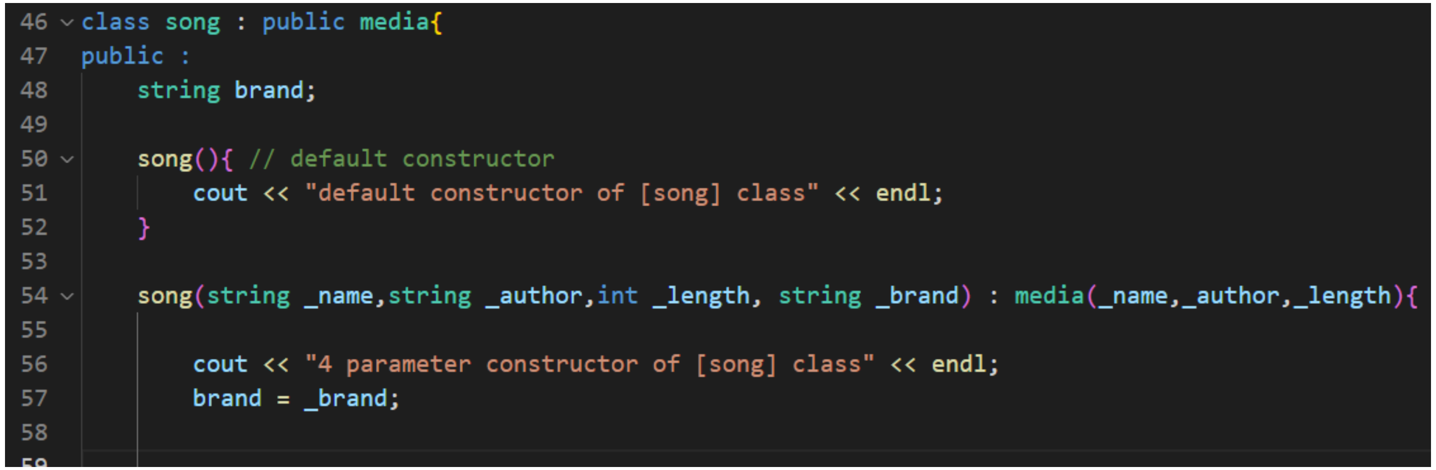
```
child_constructor : base_constructor1(p1,p2…), base_constructor2(p1,p2…) ){


    cout << "4 parameter constructor of [song] class" << endl;
    brand = _brand;
}
```

Base constructor



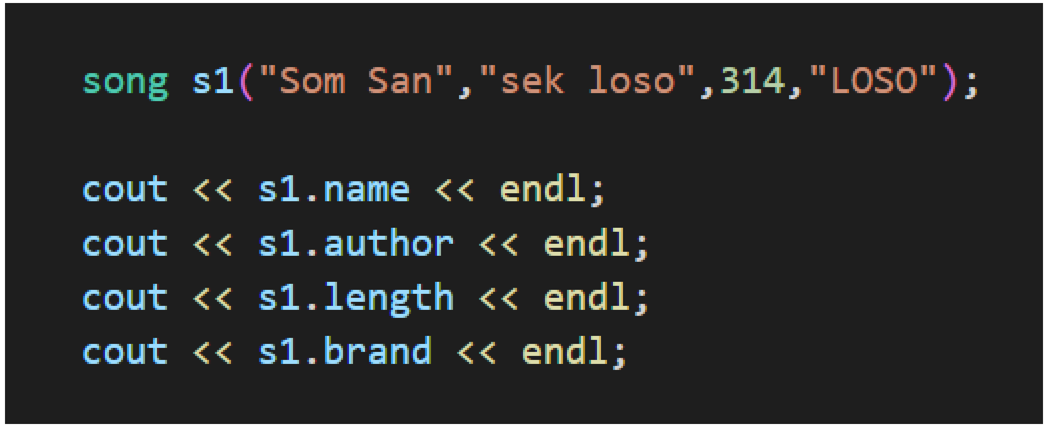
```cpp
class song : public media{
public :
    string brand;

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

    song(string _name,string _author,int _length, string _brand) : media(_name,_author,_length){

        cout << "4 parameter constructor of [song] class" << endl;
        brand = _brand;

```

Base constructor



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

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

    song(string _name,string _author,int _length, string _brand) : media(_name,_author,_length){

        cout << "4 parameter constructor of [song] class" << endl;
        brand = _brand;

```

Code :

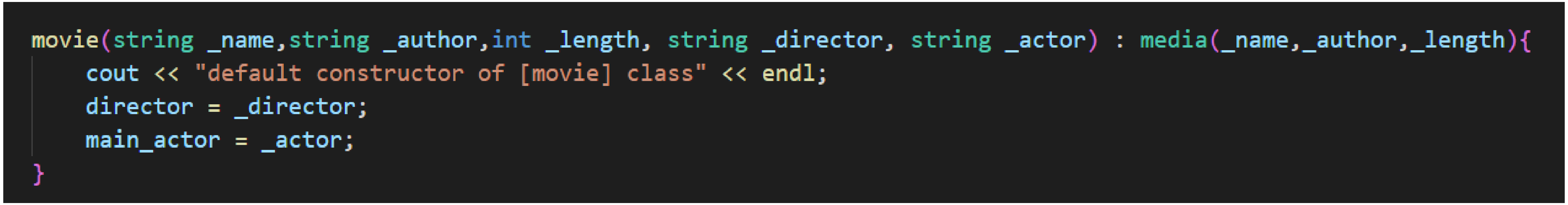


```
song s1("Som San","sek loso",314,"LOSO");

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

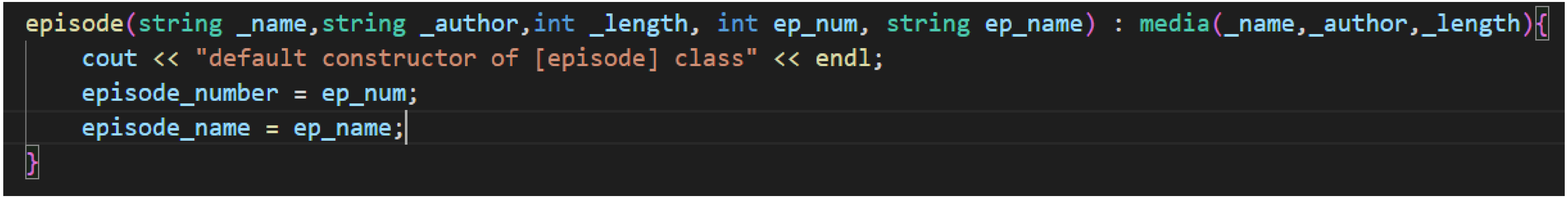
Result :

```
3 parameter constructor of [media] class
4 parameter constructor of [song] class
Som San
sek loso
314
LOSO
```

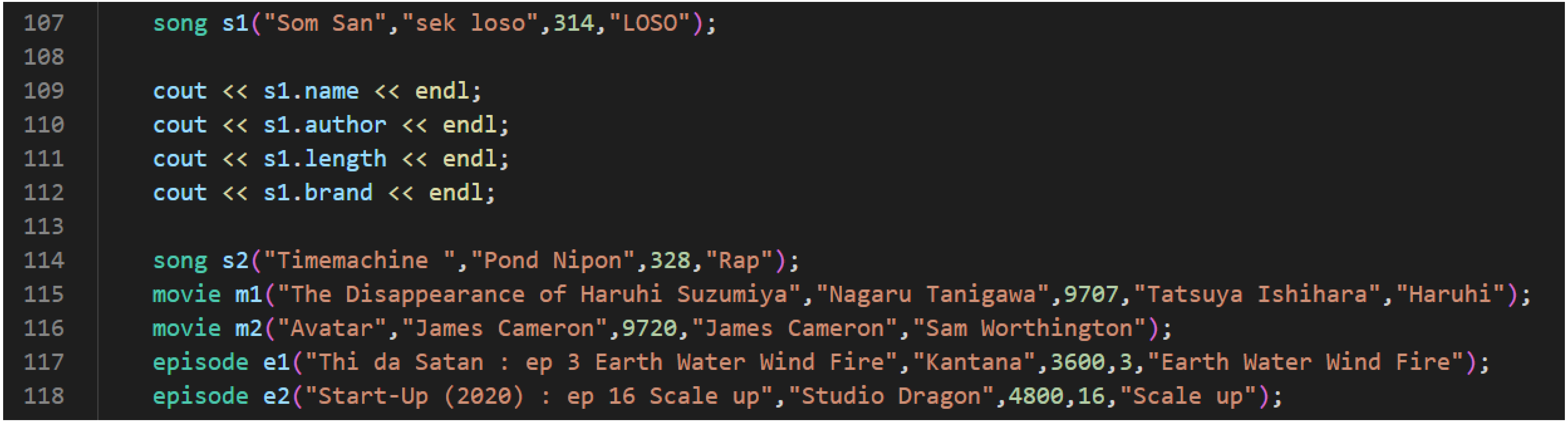
Base constructor another



```
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;
}
```



```cpp
107     song s1("Som San","sek loso",314,"LOSO");
108 
109     cout << s1.name << endl;
110     cout << s1.author << endl;
111     cout << s1.length << endl;
112     cout << s1.brand << endl;
113 
114     song s2("Timemachine ","Pond Nipon",328,"Rap");
115     movie m1("The Disappearance of Haruhi Suzumiya","Nagaru Tanigawa",9707,"Tatsuya Ishihara","Haruhi");
116     movie m2("Avatar","James Cameron",9720,"James Cameron","Sam Worthington");
117     episode e1("Thi da Satan : ep 3 Earth Water Wind Fire","Kantana",3600,3,"Earth Water Wind Fire");
118     episode e2("Start-Up (2020) : ep 16 Scale up","Studio Dragon",4800,16,"Scale up");
```

Result :

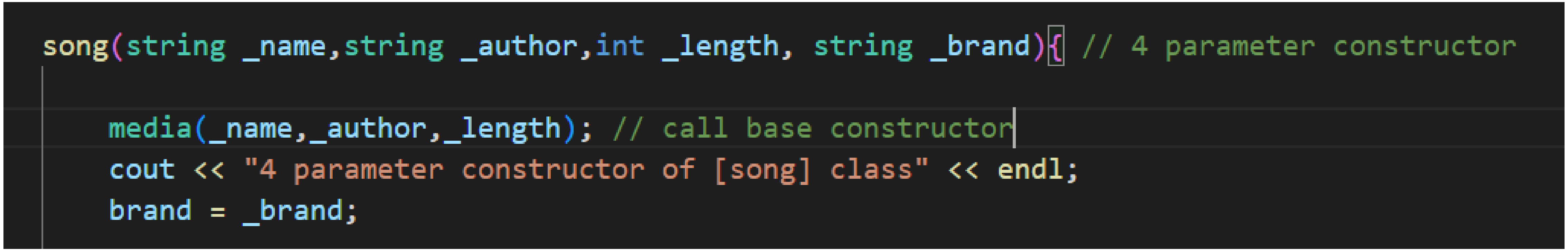
```
3 parameter constructor of [media] class (107)    3 parameter constructor of [media] class (115)
4 parameter constructor of [song] class (107)     5 parameter constructor of [movie] class (115)
Som San (109)                                     3 parameter constructor of [media] class (116)
sek loso (110)                                    5 parameter constructor of [movie] class (116)
314 (111)                                         3 parameter constructor of [media] class (117)
LOSO (112)                                        5 parameter constructor of [episode] class (117)
3 parameter constructor of [media] class (114)    3 parameter constructor of [media] class (118)
4 parameter constructor of [song] class (114)     5 parameter constructor of [episode] class (118)
```

Constructor (from encapsulation)

- function ที่ถูกเรียกใช้ทุกครั้งที่มีการสร้าง object
  - ใช้เพื่อการกำหนดค่าเริ่มต้นและตั้งค่าก่อนใช้ตัวแปร
  - สร้าง method ที่ชื่อเหมือน struct หรือ class
  - สามารถมี parameter ได้
  - Default constructor คือ constructor ที่ไม่รับ parameter หรือ มีแต่ default parameter
  - Default constructor จะถูก call เสมอหากไม่มี การ call constructor อื่น

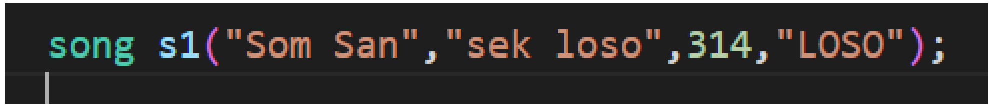
Alternative Base constructor (work but not recommend)

Call Base constructor directly



```
song(string _name,string _author,int _length, string _brand){ // 4 parameter constructor

    media(_name,_author,_length); // call base constructor
    cout << "4 parameter constructor of [song] class" << endl;
    brand = _brand;
```

Code :



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

Result :

default constructor of [media] class

3 parameter constructor of [media] class

4 parameter constructor of [song] class

Quiz

Inheritance Modifier

`private :`

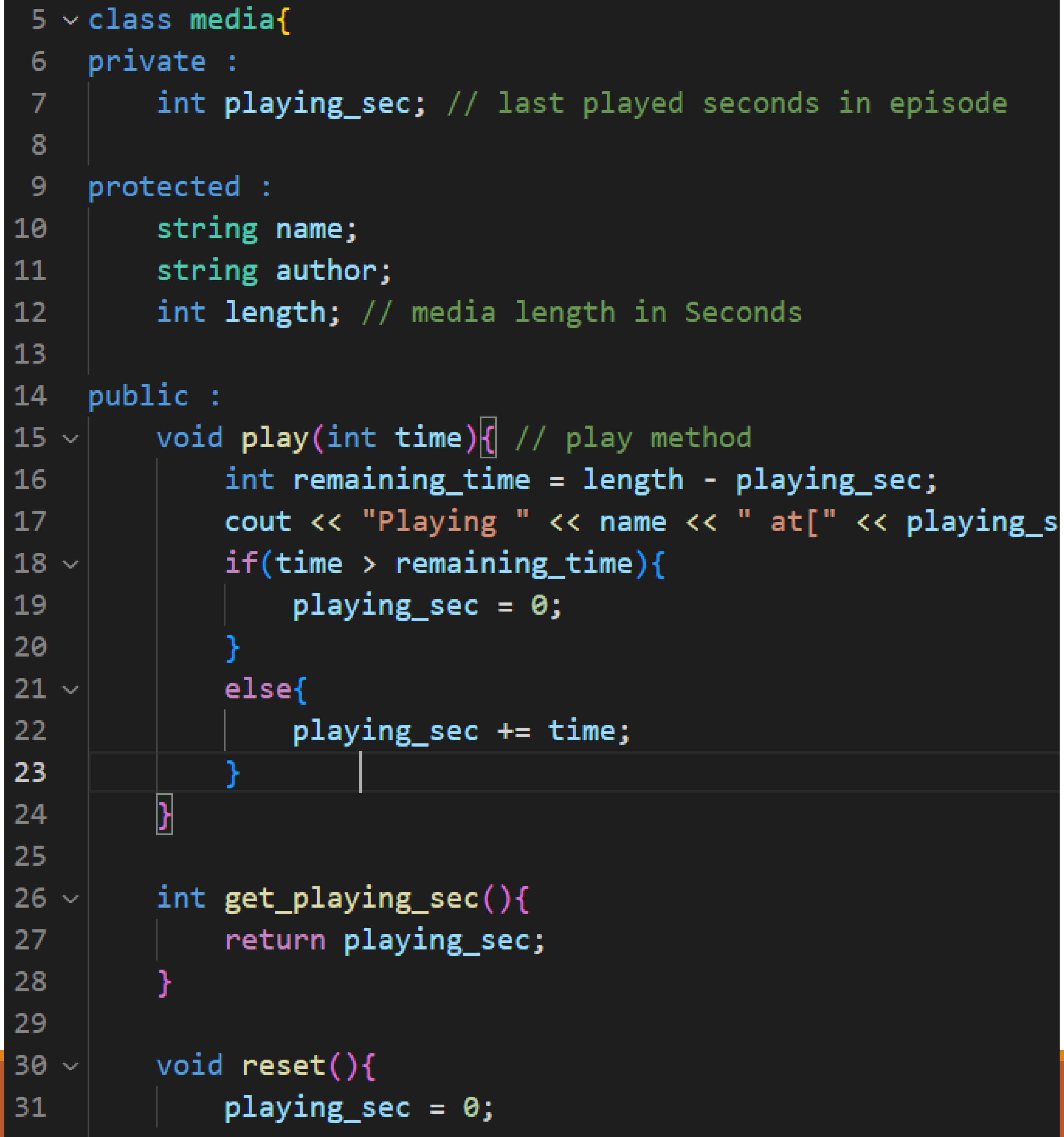
- accessible only in the same class , เข้าถึงได้เฉพาะใน class เดียวกัน

`protected :`

- accessible in same class and Derived or child class
- เข้าถึงจาก class เดียวกัน และ class ที่สืบทอดไป (class ลูกๆ)

`public :`

- accessible everywhere, เข้าถึงได้จากทุกที่



```
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;
```

- No one can edit playing sec directly
- ไม่มีใครสามารถเปลี่ยน playing sec โดยตรงได้

- Name ,author ,length สามารถเข้าถึงผ่านทาง class song ,movie ,episode ได้
- Name ,author ,length is accessible via song ,movie ,episode class

- Play(int) , get_playing_sec(), reset can be accessed from everywhere
- Play(int) , get_playing_sec(), reset สามารถเข้าถึงได้จากทุกที่

C++ Syntax (from above slide)

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

    // class component

}
```

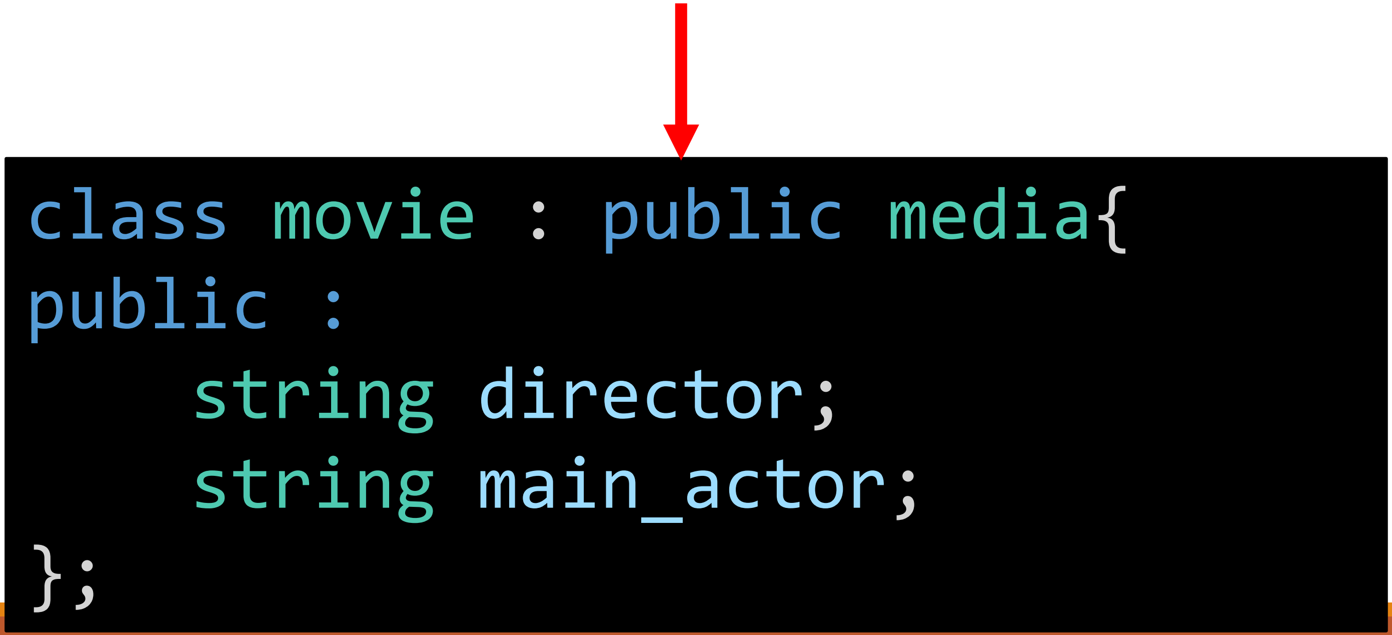
```
class movie : public media{
public :
    string director;
    string main_actor;
};
```

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

Common use Inheritance

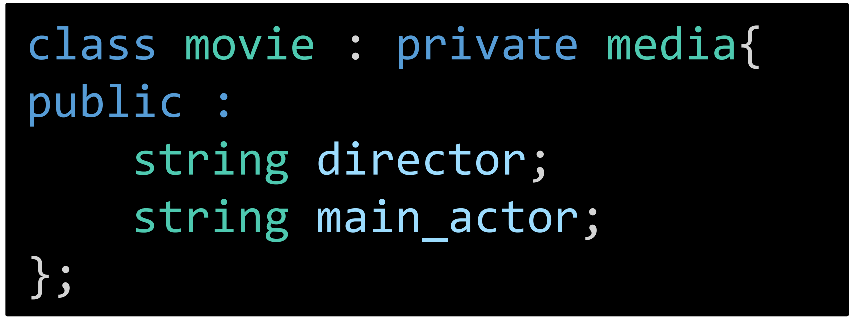
- **Public** inheritance is the most common usage

- การใช้งาน **Public** inheritance คือรูปแบบที่ใช้งานกันมากที่สุดและปกติที่สุด



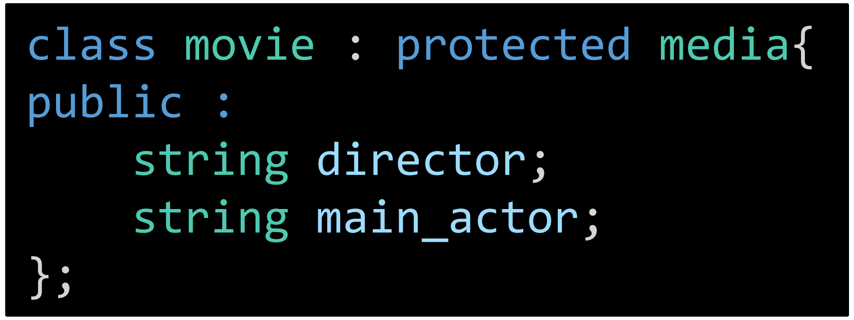
```
class movie : public media{
public :
    string director;
    string main_actor;
};
```

Private and Protected inheritance



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

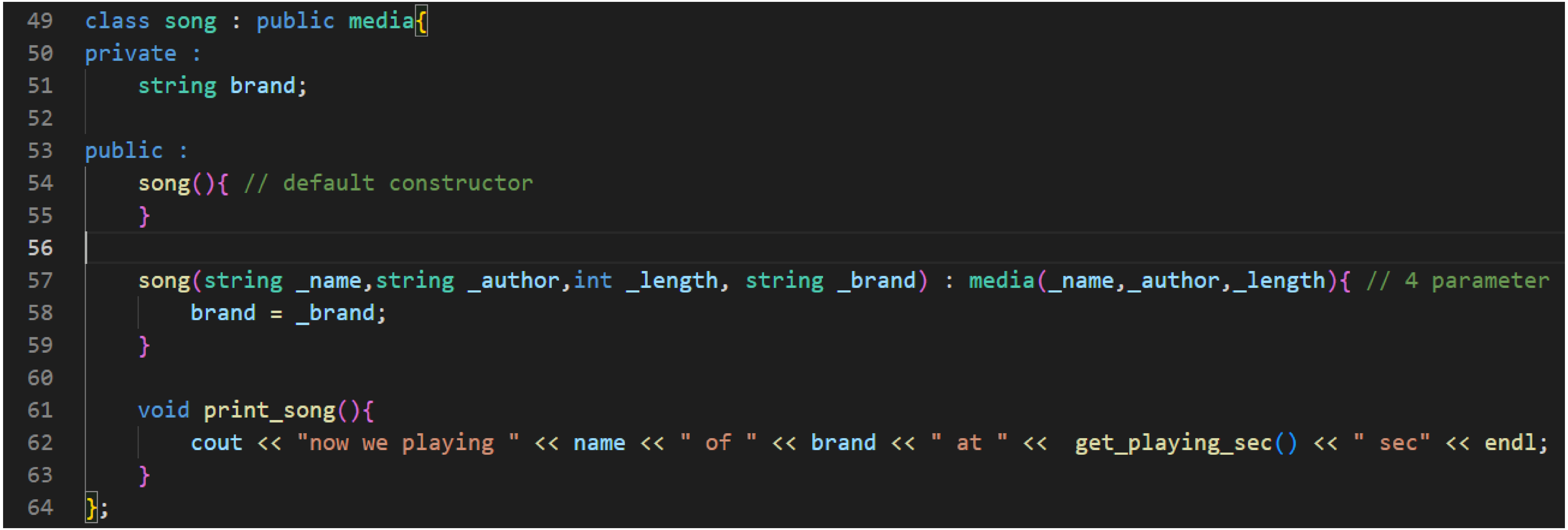
public	attribute **become** private
protected	attribute **become** private
private	attribute **become** private



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

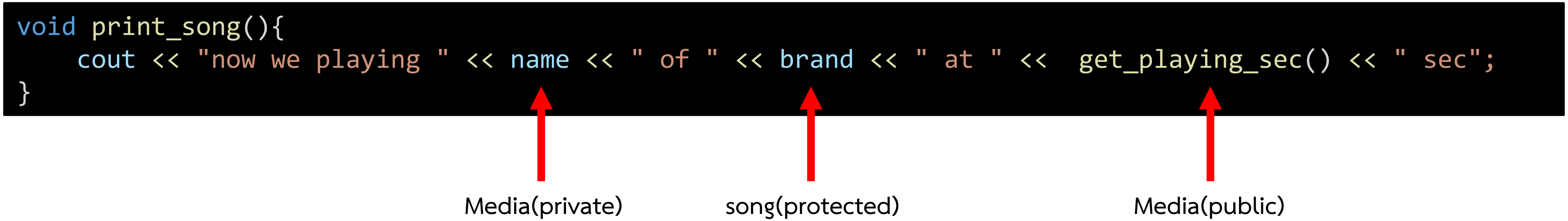
public	attribute **become** protected
protected	attribute **become** protected
private	attribute **become** private

Modifier example



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

public :
    song(){ // default constructor
    }

    song(string _name,string _author,int _length, string _brand) : media(_name,_author,_length){ // 4 parameter
        brand = _brand;
    }

    void print_song(){
        cout << "now we playing " << name << " of " << brand << " at " <<  get_playing_sec() << " sec" << endl;
    }
};
```

Code method :



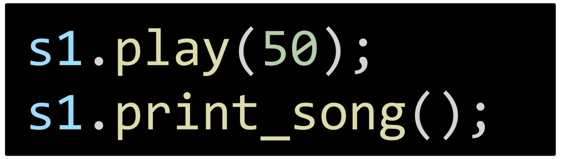
```
void print_song(){
    cout << "now we playing " << name << " of " << brand << " at " <<  get_playing_sec() << " sec";
}
```

Media(private)

song(protected)

Media(public)

Code main :



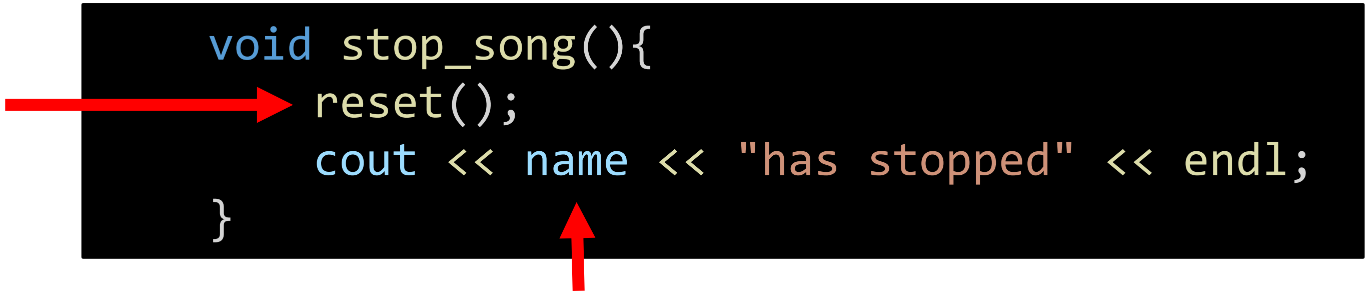
```
s1.play(50);
s1.print_song();
```

Result :

Playing Som San at[0] : [314 sec] remaining

now we playing Som San of LOSO at 50sec

Another example

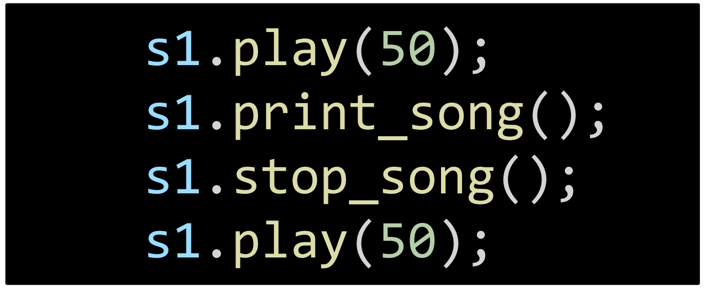


```
    void stop_song(){
        reset();
        cout << name << "has stopped" << endl;
    }
```

Media(public)

Media(private)

Code main :



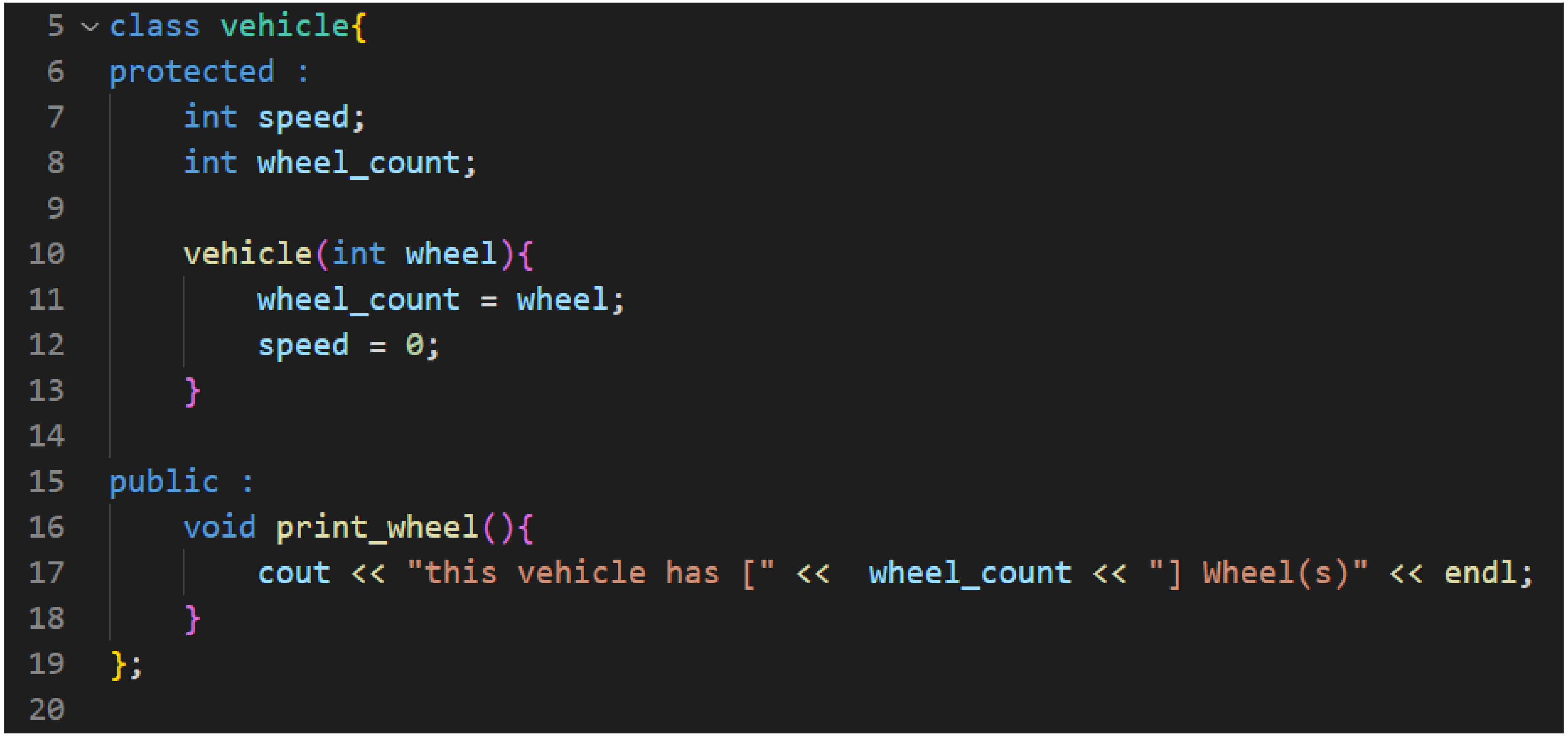
```
    s1.play(50);
    s1.print_song();
    s1.stop_song();
    s1.play(50);
```

Result :

```
Playing Som San at[0] : [314 sec] remaining
now we playing Som San of LOSO at 50 sec
Som San has stopped
Playing Som San at[0] : [314 sec] remaining
```

Move on from media

Base class :

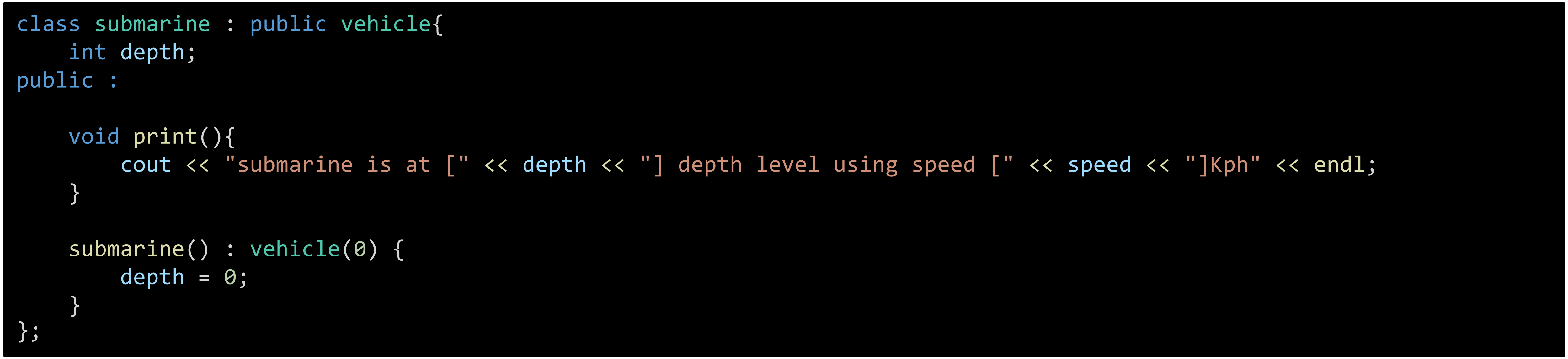


```
class vehicle{
protected :
    int speed;
    int wheel_count;

    vehicle(int wheel){
        wheel_count = wheel;
        speed = 0;
    }

public :
    void print_wheel(){
        cout << "this vehicle has [" <<  wheel_count << "] Wheel(s)" << endl;
    }
};
```

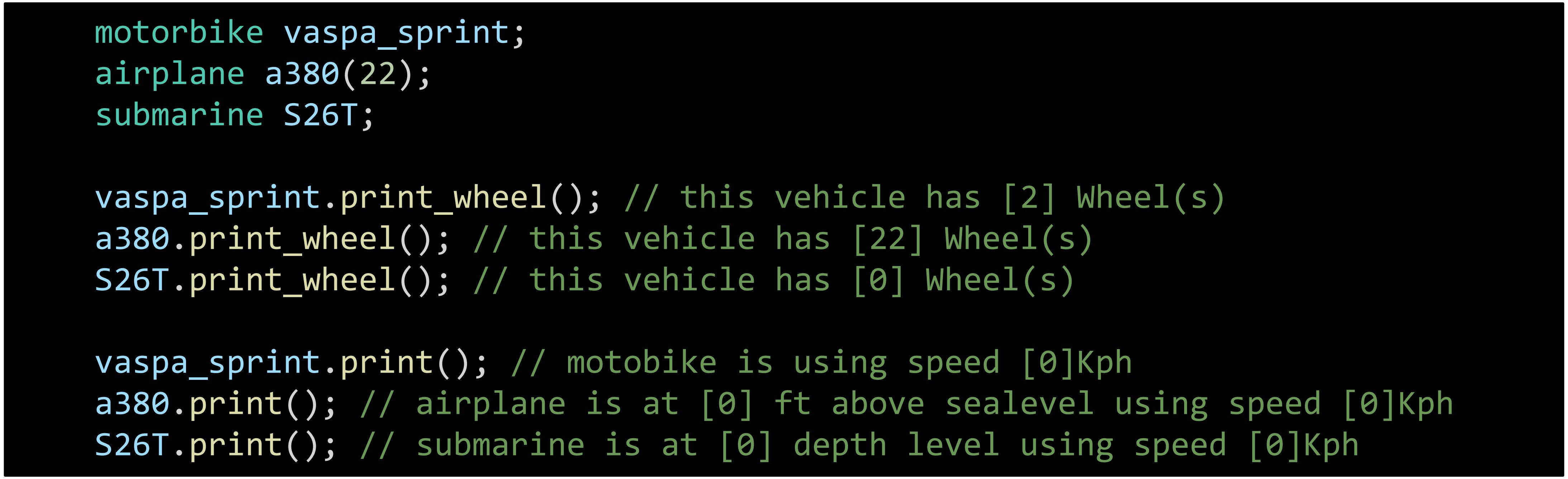
3_6_vehicle_class

```
class motorbike : public vehicle{
public :
    motorbike() : vehicle(2) {

    }

    void print(){
        cout << "motobike is using speed [" << speed << "]Kph" << endl;
    }
};
```

```
class airplane : public vehicle{
    int altitude;
public :

    void print(){
        cout << "airplane is at [" << altitude << "] ft above sealevel using speed [" << 
speed << "]Kph" << endl;
    }

    airplane(int wheel) : vehicle(wheel) {
        altitude = 0;
    }
};
```



```cpp
class submarine : public vehicle{
    int depth;
public :

    void print(){
        cout << "submarine is at [" << depth << "] depth level using speed [" << speed << "]Kph" << endl;
    }

    submarine() : vehicle(0) {
        depth = 0;
    }
};
```



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

    vaspa_sprint.print_wheel(); // this vehicle has [2] Wheel(s)
    a380.print_wheel(); // this vehicle has [22] Wheel(s)
    S26T.print_wheel(); // this vehicle has [0] Wheel(s)

    vaspa_sprint.print(); // motobike is using speed [0]Kph
    a380.print(); // airplane is at [0] ft above sealevel using speed [0]Kph
    S26T.print(); // submarine is at [0] depth level using speed [0]Kph
```

Result :

```
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

ข้อกำหนด

- ห้ามเปลี่ยน modifier ของไฟล์ LAB

- ห้ามเปลี่ยน main

- นอนนั้นทำได้ทุกอย่าง เขียนเพิ่มที่ไหนของไฟล์ก็ได้

Global variable , เพิ่ม constructor ได้