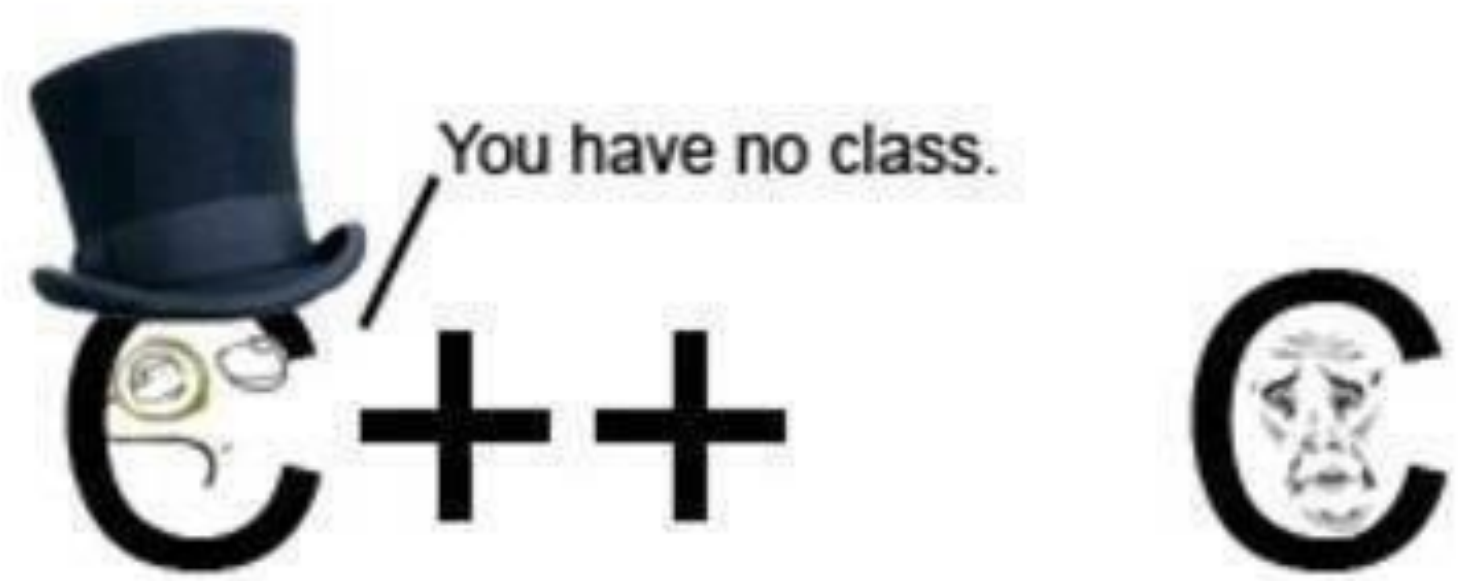
OOP & data struct

2.Class concept (Encapsulation)

BY SOMSIN THONGKRAIRAT


You have no class.
C++
C

What is Object-oriented programming (OOP)

Programming paradigm using “OBJECT” concept

รูปแบบหนึ่งของการเขียนโปรแกรมที่มองทุกอยางเป็น “OBJECT”

Most popular of Programming agreement (2022)

เป็นรูปแบบที่ใช้กันมากที่สุด (2565)

Data type

Int – integer (จำนวนเต็ม)

Float – floating point (ทศนิยม)

String – text (ตัวอักษร)

Example

```
int     my_integer          = 10;
int     my_integer2         = 20;
float   my_floating_point   = 3.14159;.
```

Variable type? ชนิดของตัวแปร?

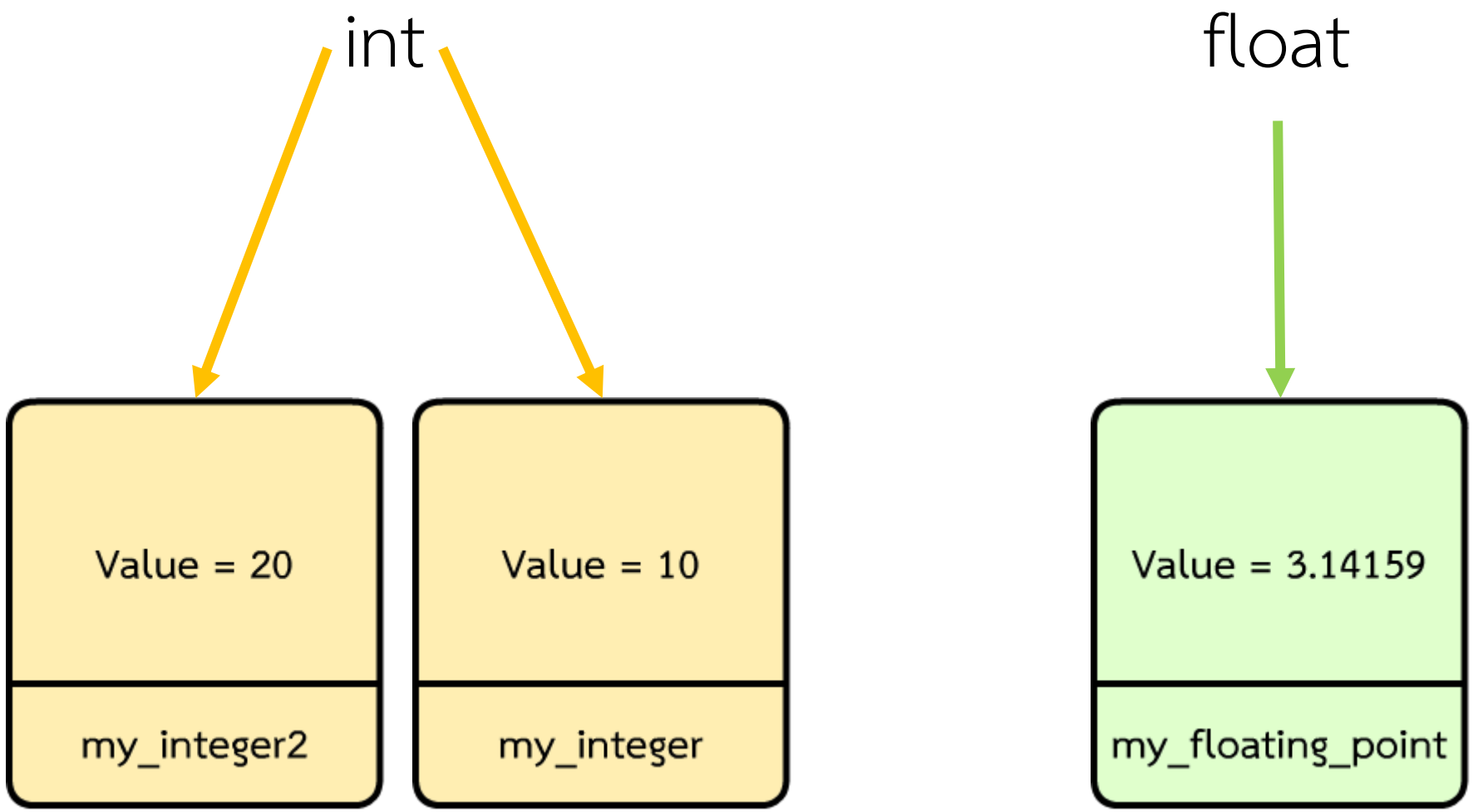
Variable name? ชื่อของตัวแปร?

Variable value? ค่าของตัวแปร?

Example

```
int     my_integer          = 10;
int     my_integer2         = 20;
float   my_floating_point   = 3.14159;
```

Variable type ชนิดตัวแปร	Variable name ชื่อตัวแปร	Variable Value ค่า


int
float
Value = 20
my_integer2
Value = 10
my_integer
Value = 3.14159
my_floating_point

Struct (previous class)

```
Anime Type
```

full_name = "Haruhi" author = "Nagaru Tanigawa" total_episode = 24 length_per_episode = 1200	full_name = "One Piece" author = "Eiichiro Oda" total_episode = 1045 length_per_episode = 900	full_name = "Spy x Family Part 1" author = "Tatsuya Endo" total_episode = 12 length_per_episode = 1440
a1	a2	a4

Type Declaration

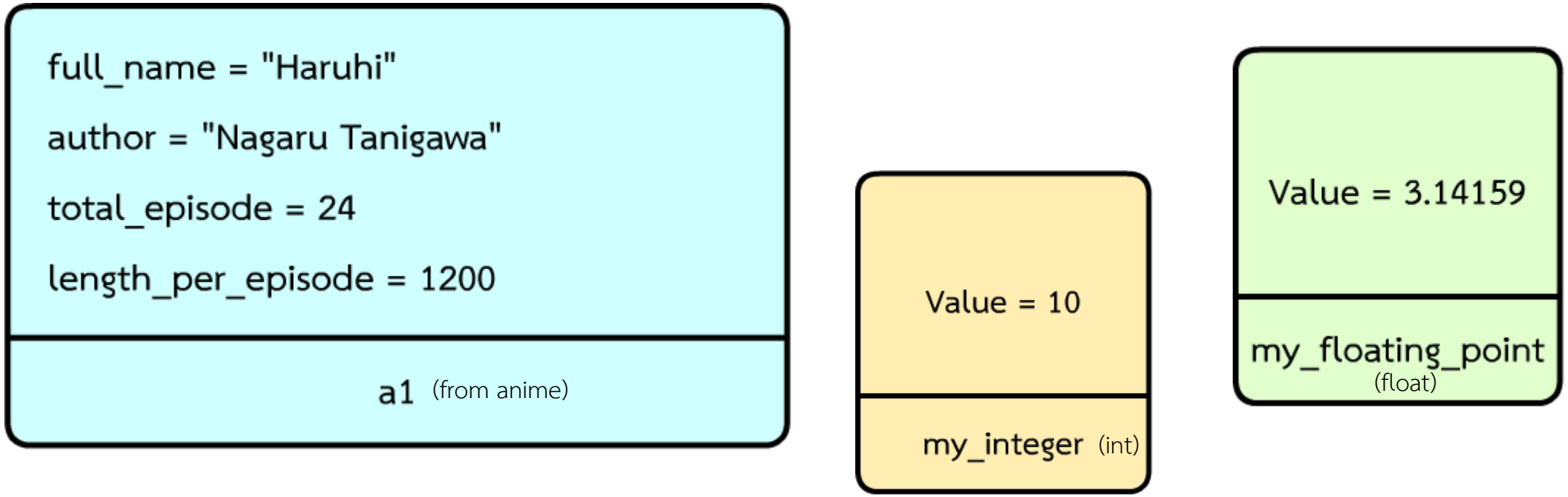
```
// struct declaration
struct anime{
    string full_name;
    string author;
    int total_episode;
    int length_per_episode;
};
```

Variable Declaration

```
// variable declaration
anime a1,a2,a3,a4;
a1.full_name = "The Melancholy of Haruhi Suzumiya";
a1.author = "Nagaru Tanigawa";
a1.total_episode = 24;
a1.length_per_episode = 1200;
```

Object Concept

จำลองค่าสถานะต่างๆ ไว้ในตัวแปร และเรียกสิ่งนั้นว่า Object (ตัวแปร)



Object Concept

- Class คือ blueprint หรือต้นแบบในการสร้าง object (data type)
- Object (instance) คือ ตัวแทนที่ถูกสร้างขึ้นมาจาก class (ตัวแปร)
- Attributes คือ ข้อมูลหรือสถานะของ object นั้นๆ (ตัวแปรใน struct)
- Method คือ การทำงานของ object (คำสั่ง หรือ function) ***

*แตกต่างจาก procedural อย่างชัดเจน

Example for modeling class

กำหนดให้เขียนโปรแกรมเพื่อจัดเก็บ ชื่อ ผู้แต่ง จำนวนตอน ความยาว ของ series หรือ anime (กำหนดให้ ความยาวของแต่ละตอนเท่ากัน)

- เก็บตอนล่าสุดและวินาที่ล่าสุดที่ดูเพื่อ resume การดูได้

- มี function play คือ กำหนดวินาทีที่จะดู คำนวณตอนที่ถูกดูออกมา (cout ออกมา)

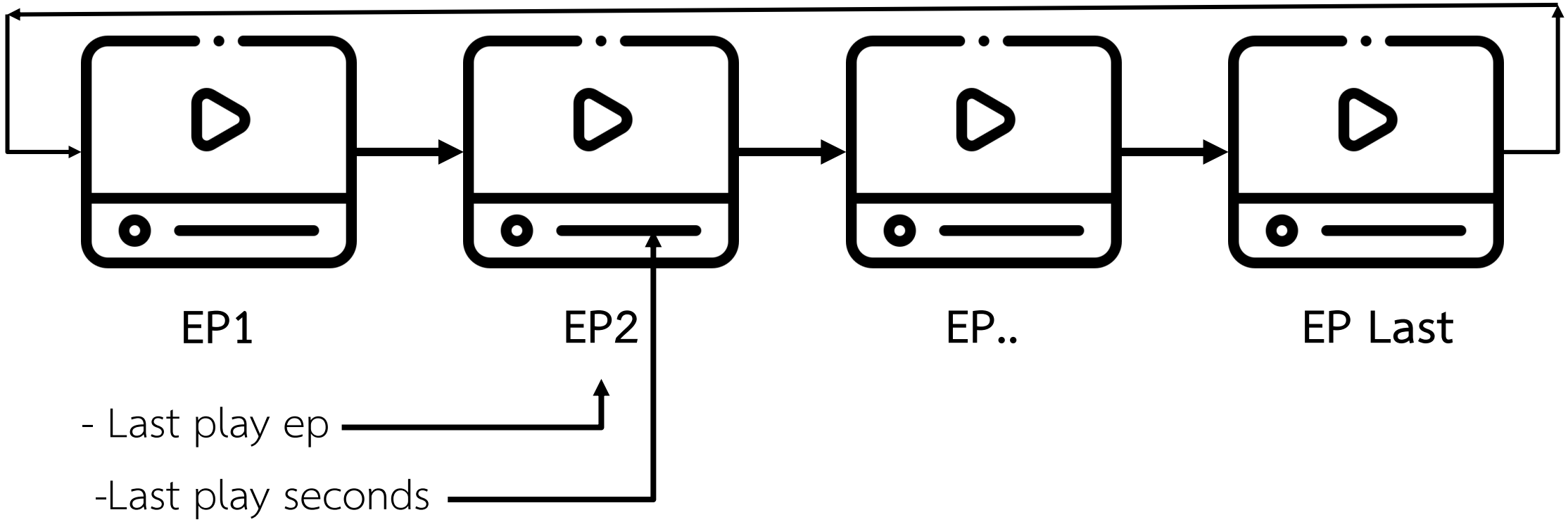
- หากเล่นจนจบตอนให้หยุดเล่นและเปลี่ยนตอนล่าสุดเป็นตอนถัดไป (ไม่ต้องเล่นต่อ) หากจบทุกตอนให้ไปเริ่มต้นที่ตอนแรก

- Last play ep

-Last play seconds

Example for modeling class

Series or Anime (Same length)



Example for modeling class (Attributes)

```
4
5   // struct declaration
6   struct anime{
7       string full_name;
8       string author;
9       int total_episode;
10      int length_per_episode; // Average running time in Seconds
11
12      int playing_episode; // last played episode
13      int playing_sec; // last played seconds in episode
14  };
15
```

*From recent LAB

Example for modeling class (Method)

```
// function definition
void play(anime *a,int time){
    int remaining_time = a->length_per_episode - a->playing_sec;
    if(time > remaining_time){ // next ep
        cout << "playing " << a->full_name << " EP." << a->playing
        a->playing_episode++;
        if(a->playing_episode >= a->total_episode){
            a->playing_episode = 1;
        }
        a->playing_sec = 0;
    }
    else{
        cout << "playing " << a->full_name << " EP." << a->playing
        a->playing_sec += time;
    }
}
```

```
// variable declaration
anime a1,a2;
a1.full_name = "The Melancholy of Haruhi Suzumiya";
a1.author = "Nagaru Tanigawa";
a1.total_episode = 24;
a1.length_per_episode = 1200;
a1.playing_episode = 1;
a1.playing_sec = 0;

a2.full_name = "Spy x Family Part 1";
a2.author = "Tatsuya Endo";
a2.total_episode = 12;
a2.length_per_episode = 1440;
a2.playing_episode = 1;
a2.playing_sec = 0;
```

output

```
play(&a1,500);
play(&a1,99999);
play(&a1,99999);
play(&a2,100);
play(&a2,100);
play(&a2,100);
play(&a2,99999);
play(&a1,500);
play(&a1,99999);
```

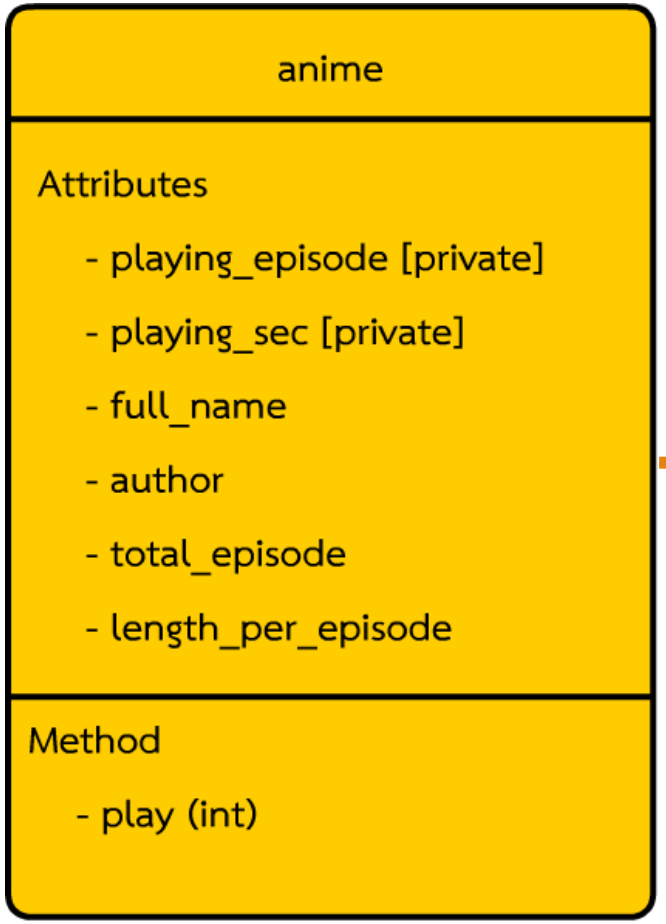
```
playing The Melancholy of Haruhi Suzumiya EP.1 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.1 [700 sec]
playing The Melancholy of Haruhi Suzumiya EP.2 [1200 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [1140 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [700 sec]
```

Object Concept

- Class : Anime
- Object (instance) : a1,a2
- Attributes (property) : full_name, author, total_ep, etc.
- Method : play(time)

*แตกต่างจาก procedural อย่างชัดเจน

Class or Blueprint



Object

a1,a2...

Class

anime

Attributes

- playing_episode [private]
- playing_sec [private]
- full_name
- author
- total_episode
- length_per_episode

Method

- play (int)

a1 (anime)

Attributes

- playing_episode = 1
- playing_sec = 0
- full_name = The Melancholy of Haruhi Suzumiya
- author = Nagaru Tanigawa
- total_episode = 24
- length_per_episode = 1200

Method

- play (int)

Object

a2 (anime)

Attributes

- playing_episode = 1
- playing_sec = 0
- full_name = Spy x Family Part 1
- author = Tatsuya Endo
- total_episode = 12
- length_per_episode = 1440

Method

- play (int)

Object

Type Declaration

(Class)

```
// struct declaration
struct anime{
    string full_name;
    string author;
    int total_episode;
    int length_per_episode;
};
```

Variable Declaration

(Object)

```
// variable declaration
anime a1,a2,a3,a4;
a1.full_name = "The Melancholy of Haruhi Suzumiya";
a1.author = "Nagaru Tanigawa";
a1.total_episode = 24;
a1.length_per_episode = 1200;
```

Let’s make it OOP

Class name

Attributes

Method

```
struct anime{
    string full_name;
    string author;
    int total_episode;
    int length_per_episode; // Average running time in Seconds

    int playing_episode; // last played episode
    int playing_sec; // last played seconds in episode

    void play(int time){
        int remaining_time = length_per_episode - playing_sec;
        if(time > remaining_time){ // next ep
            cout << "playing " << full_name << " EP." << playing_episode <<" [" << remaining_time <<" sec]" << endl;
            playing_episode++;
            if(playing_episode >= total_episode){
                playing_episode = 1;
            }
            playing_sec = 0;
        }
        else{
            cout << "playing " << full_name << " EP." << playing_episode <<" [" << time <<" sec]" << endl;
            playing_sec += time;
        }
    }

};
```

Object

```
// variable declaration
anime a1("The Melancholy of Haruhi Suzumiya","Nagaru Tanigawa",24,1200);
anime a2("Spy x Family Part 1","Tatsuya Endo",12,1440);
if(true){
    anime a3("detective conan","Gosho Aoyama",1067,1200);
    a3.play(20);
}
//a1.playing_episode = -2;
a1.select_episode(3);
a1.select_episode(-2);
```

Same result

```
a1.play(500);      // playing The Melancholy of Haruhi Suzumiya EP.1 [500 sec]
a1.play(500);      // playing The Melancholy of Haruhi Suzumiya EP.1 [500 sec]
a1.play(99999);    // playing The Melancholy of Haruhi Suzumiya EP.1 [700 sec]
a1.play(99999);    // playing The Melancholy of Haruhi Suzumiya EP.2 [1200 sec]
a2.play(100);      // playing Spy x Family Part 1 EP.1 [100 sec]
a2.play(100);      // playing Spy x Family Part 1 EP.1 [100 sec]
a2.play(100);      // playing Spy x Family Part 1 EP.1 [100 sec]
a2.play(99999);    // playing Spy x Family Part 1 EP.1 [1140 sec]
a1.play(500);      // playing The Melancholy of Haruhi Suzumiya EP.3 [500 sec]
a1.play(99999);    // playing The Melancholy of Haruhi Suzumiya EP.3 [700 sec]
```

Method

a1.play() method using a1 attributes

a2.play() method using a2 attributes

function a1.play() ใช้ attributes ของ a1 ทั้งหมด

function a2.play() ใช้ attributes ของ a2 ทั้งหมด

Called variable

```
// variable declaration
anime a1,a2;
a1.full_name = "The Melancholy of Haruhi Suzumiya";
a1.author = "Nagaru Tanigawa";
a1.total_episode = 24;
a1.length_per_episode = 1200;
a1.playing_episode = 1;
a1.playing_sec = 0;

a2.full_name = "Spy x Family Part 1";
a2.author = "Tatsuya Endo";
a2.total_episode = 12;
a2.length_per_episode = 1440;
a2.playing_episode = 1;
a2.playing_sec = 0;
```

```
a1.play(500);
a1.play(500);
a1.play(99999);
a1.play(99999);
a2.play(100);
a2.play(100);
a2.play(100);
a2.play(99999);
a1.play(500);
a1.play(99999);
```

```
a1.play(500);
a1.play(500);
a1.play(99999);
a1.play(99999);
```

a1.length_per_episode

a1.playing_sec

```
void play(int time){
    int remaining_time = length_per_episode - playing_sec;
    if(time > remaining_time){ // next ep
        cout << "playing " << full_name << " EP." << playing_ep
        playing_episode++;
        if(playing_episode >= total_episode){
            playing_episode = 1;
        }
        playing_sec = 0;
    }
    else{
        cout << "playing " << full_name << " EP." << playing_ep
        playing_sec += time;
    }
}
```

a1.full_name

Difference

```
// function definition
void play(anime *a,int time){
    int remaining_time = a->length_per_episode - a->pl
    if(time > remaining_time){ // next ep
        cout << "playing " << a->full_name << " EP." <
        a->playing_episode++;
        if(a->playing_episode >= a->total_episode){
            a->playing_episode = 1;
        }
        a->playing_sec = 0;
    }
    else{
        cout << "playing " << a->full_name << " EP." <
        a->playing_sec += time;
    }
}
```

Procedural (function)

```
void play(int time){
    int remaining_time = length_per_episode - playing_sec;
    if(time > remaining_time){ // next ep
        cout << "playing " << full_name << " EP." << playing
        playing_episode++;
        if(playing_episode >= total_episode){
            playing_episode = 1;
        }
        playing_sec = 0;
    }
    else{
        cout << "playing " << full_name << " EP." << playing
        playing_sec += time;
    }
}
```

OOP (method)

Constructor (initial variable)

```
// variable declaration
anime a1,a2;
a1.full_name = "The Melancholy of Haruhi Suzumiya";
a1.author = "Nagaru Tanigawa";
a1.total_episode = 24;
a1.length_per_episode = 1200;
a1.playing_episode = 1;
a1.playing_sec = 0;
```

Constructor

- 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

Constructor syntax

Method that have the same name with class without return type

Method ที่มีชื่อเหมือน Class และไม่มี return type

In C++

Constructor EX.

```
6   struct anime{
7
8       anime(){
9
10          playing_episode = 1;
11          playing_sec = 0;
12      }
13
```

Constructor EX.

```
// variable declaration
anime a1,a2;
a1.full_name = "The Melancholy of Haruhi Suzumiya";
a1.author = "Nagaru Tanigawa";
a1.total_episode = 24;
a1.length_per_episode = 1200;

cout << a1.playing_episode << endl;
cout << a2.playing_episode << endl;

a2.full_name = "Spy x Family Part 1";
a2.author = "Tatsuya Endo";
a2.total_episode = 12;
a2.length_per_episode = 1440;
```

Result

```
1
1
```

Constructor

*Default constructor จะถูก call เสมอหากไม่มี การ call constructor อื่น

Constructor with parameter

```
41      anime(){
42          cout << "this is default constructor" << endl;
43
44          playing_episode = 1;
45          playing_sec = 0;
46      }
47
48      anime(string _name,string _author,int _ep,int length){
49          cout << "this is constructor for " << _name << endl;
50
51          full_name = _name;
52          author = _author;
53          total_episode = _ep;
54          length_per_episode = length;
55
56          playing_episode = 1;
57          playing_sec = 0;
58      }
59
```

Default constructor

Constructor with argument

Call constructor with parameter

```
anime a1("The Melancholy of Haruhi Suzumiya","Nagaru Tanigawa",24,1200);

anime a2;
a2.full_name = "Spy x Family Part 1";
a2.author = "Tatsuya Endo";
a2.total_episode = 12;
a2.length_per_episode = 1440;
```

Call default constructor

Result

```
this is constructor forThe Melancholy of Haruhi Suzumiya
this is default constructor
playing The Melancholy of Haruhi Suzumiya EP.1 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.1 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.1 [200 sec]
playing The Melancholy of Haruhi Suzumiya EP.2 [1200 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [1140 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [700 sec]
```

Quiz

CONSTRUCTOR

Variable modifiers

- Limit access for methods and attributes
  - public, private, protected (c++)
  - For prevent irregular variable change from outside class
  - For check valid command or not
  - ***more secure?***

Variable modifiers

- เพื่อจำกัดการเข้าถึง methods and attributes ใน object
  - public, private, protected (c++)
  - ป้องกันการเปลี่ยนแปลงตัวแปรที่ทำให้โปรแกรมทำงานผิดพลาด
  - ตรวจสอบความถูกต้องของคำสั่งนั้น
  - ***ความปลอดภัย?***

Consider this case

```
// variable declaration
anime a1("The Melancholy of Haruhi Suzumiya","Nagaru Tanigawa",24,1200);
a1.playing_episode = -2;
```

Result T_T

```
playing The Melancholy of Haruhi Suzumiya EP.-2 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.-2 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.-2 [200 sec]
playing The Melancholy of Haruhi Suzumiya EP.-1 [1200 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [1140 sec]
playing The Melancholy of Haruhi Suzumiya EP.0 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.0 [700 sec]
```

How to prevent it?

- Create get set method to manage attributes in class

- สร้าง method get set เพื่อจัดการ attributes ภายใน class

```
void select_episode(int _ep){
    if(_ep <= 0) return;
    if(_ep > total_episode) return;

    playing_episode = _ep;
    playing_sec = 0;
}
```

```
// variable declaration
anime a1("The Melancholy of Haruhi Suzumiya","Nagaru Tanigawa",24,1200);
//a1.playing_episode = -2;
a1.select_episode(3);
a1.select_episode(-2);
```

Result

```
playing The Melancholy of Haruhi Suzumiya EP.3 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [200 sec]
playing The Melancholy of Haruhi Suzumiya EP.4 [1200 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [1140 sec]
playing The Melancholy of Haruhi Suzumiya EP.5 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.5 [700 sec]
```

Variable modifiers

```
string full_name;
string author;
int total_episode;
int length_per_episode;

int playing_episode; //
int playing_sec; // last
```

No need to accessed by outside

Make it Private and access via method!

Variable modifiers

Public เข้าถึงได้จากทุกที่ ไม่จำกัด

Private เข้าถึงได้เพียงแค่ใน class เดียวกัน

Protected เข้าถึงได้จาก class ที่สืบทอด (inheritance) ไป

Syntax

```
struct or class name{
public :
    public property1;
    public property2;
    public property3;
    public method1();
    public method2();

private :
    private property1;
    private property2;
    private property3;
    private method1();
    private method2();
}
```

Public section

Private section

Variable modifiers Ex.

```
struct anime{
private :
    int playing_episode; // last played episode
    int playing_sec; // last played seconds in episode

public :
    string full_name;
    string author;
    int total_episode;
    int length_per_episode; // Average running time in Seconds

    void play(int time){ // play method
        int remaining_time = length_per_episode - playing_sec;
        if(time > remaining_time){ // next ep
            cout << "playing " << full_name << " EP." << playing_ep
            playing_episode++;
            if(playing_episode >= total_episode){
                playing_episode = 1;
            }
            playing_sec = 0;
        }
        else{
```

Public section

Private section

Variable modifiers Ex.

```
// variable declaration
anime a1("The Melancholy of Haruhi Suzumiya","Nagaru Tanigawa",24,1200);
anime a2("Spy x Family Part 1","Tatsuya Endo",12,1440);
//a1.playing_episode = -2;
a1.select_episode(3); // pass
a1.select_episode(-2); // pass
cout << a2.full_name << endl; // pass
a1.playing_episode--; // inaccessible
a2.playing_sec = -36.33; // inaccessible
a2.total_episode += 1; // pass
a2.author = "aabbbb"; // pass
```

Destructor

- Create method that has ‘~’ symbol follow by class name
- สร้าง method ที่ชื่อขึ้นต้นด้วยเครื่องหมาย ‘~’ และตามด้วยชื่อ Class

```
~anime(){
    cout << full_name << "has destroyed" << endl;
}
```

```
this is constructor forThe Melancholy of Haruhi Suzumiya
this is constructor forSpy x Family Part 1
playing The Melancholy of Haruhi Suzumiya EP.3 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [200 sec]
playing The Melancholy of Haruhi Suzumiya EP.4 [1200 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [1140 sec]
playing The Melancholy of Haruhi Suzumiya EP.5 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.5 [700 sec]
Spy x Family Part 1has destroyed
The Melancholy of Haruhi Suzumiyahas destroyed
```

When will variable destroy?

- when it out of scope {}

```
// variable declaration
anime a1("The Melancholy of Haruhi Suzumiya","Nagaru Tanigawa
anime a2("Spy x Family Part 1","Tatsuya Endo",12,1440);
if(true){
    anime a3("detective conan","Gosho Aoyama",1067,1200);
    a3.play(20);
}
//a1.playing_episode = -2;
a1.select_episode(3);
a1.select_episode(-2);


a1.play(500);      // playing The Melancholy of Haruhi Suzumi
a1.play(500);      // playing The Melancholy of Haruhi Suzumi
a1.play(99999);    // playing The Melancholy of Haruhi Suzumi
a1.play(99999);    // playing The Melancholy of Haruhi Suzumi
a2.play(100);      // playing Spy x Family Part 1 EP.1 [100 se
a2.play(100);      // playing Spy x Family Part 1 EP.1 [100 se
a2.play(100);      // playing Spy x Family Part 1 EP.1 [100 se
a2.play(99999);    // playing Spy x Family Part 1 EP.1 [1140 s
a1.play(500);      // playing The Melancholy of Haruhi Suzumi
```

```
this is constructor forThe Melancholy of Haruhi Suzumiya
this is constructor forSpy x Family Part 1
this is constructor fordetective conan
playing detective conan EP.1 [20 sec]
detective conan has destroyed
playing The Melancholy of Haruhi Suzumiya EP.3 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [500 sec]
playing The Melancholy of Haruhi Suzumiya EP.3 [200 sec]
playing The Melancholy of Haruhi Suzumiya EP.4 [1200 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
playing Spy x Family Part 1 EP.1 [100 sec]
```

What is class

- struct with default public field!

```
6   struct anime{
7   private :
8       int playing_episode; // last played episode
9       int playing_sec; // last played seconds in episode
10
11  public :
12      string full_name;
13      string author;
14      int total_episode;
15      int length_per_episode; // Average running time in Secon
16
17      void play(int time){ // play method
18          int remaining_time = length_per_episode - playing_se
19          if(time > remaining_time){ // next ep
20              cout << "playing " << full_name << " EP." << pla
```

```
6   class anime{
7       int playing_episode; // last played episode
8       int playing_sec; // last played seconds in episode
9
10  public :
11      string full_name;
12      string author;
13      int total_episode;
14      int length_per_episode; // Average running time in Second
15
16      void play(int time){ // play method
17          int remaining_time = length_per_episode - playing_sec
18          if(time > remaining_time){ // next ep
19              cout << "playing " << full_name << " EP." << play
```

Conclude

- what is class
- constructor & destructor
- modifier (public private)

Another Example

```
Class Car
Attribute :
    string name;
    float acceleration;
    float deceleration;
    float speed;
Method :
    void print()
    void speed_up()
    void speed_down()
```

```
car c1("ae86",1.5);
car c2;
car c3("Honda wave",0.2);
car c4("tesla model X",9.92);
```

```
cout << "speeding up" << endl;
for(int i=0;i<10;i++){
    c1.speed_up(); c2.speed_up(); c3.speed_up(); c4.speed_up();
}

c1.print(); c2.print(); c3.print(); c4.print();

cout << "slowing down" << endl;
c1.speed_down(); c2.speed_down(); c3.speed_down(); c4.speed_down();

c1.print(); c2.print(); c3.print(); c4.print();
```

Quiz (what is output)

LAB