MIIA0106

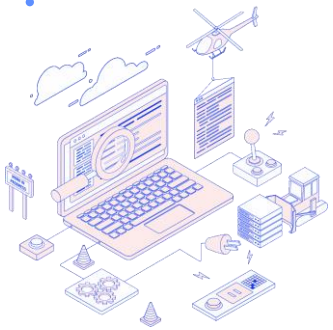
PYTHON AND C PROGRAMMING LANGUAGE

การเขียนโปรแกรมเชิงวัตถุ

Concepts, Inheritance, Polymorphism & STL

อาจารย์ สุทิศ องอาจ

LEC 9

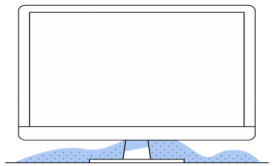


ตารางสอนรายวิชา MIIA0106

Section A

วัน	คาบ	เวลาเรียน	ห้อง
อังคาร (TUE)	คาบ 2	12.30 – 15.00 น.	D604
อังคาร (TUE)	คาบ 3	15.30 – 18.00 น.	MII203

Section B



วัน	คาบ	เวลาเรียน	ห้อง
เสาร์ (SAT)	คาบ 4	16.00 – 18.30 น.	**MII207B**
อาทิตย์ (SUN)	คาบ 4	15.30 – 17.30 น.	MII201

ช่วงสอบ กลางภาค 10 ม.ค – 18 ม.ค 69

สอบประจำภาค วันอาทิตย์ ที่ 15 มีนาคม พ.ศ. 2569

เช้า 09:00-12:00 น.

ขาดสอบ จะได้เป็น FE

MIIA0106

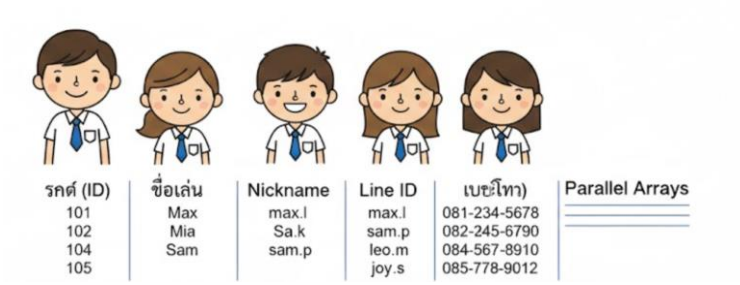
Python and C Programming Language

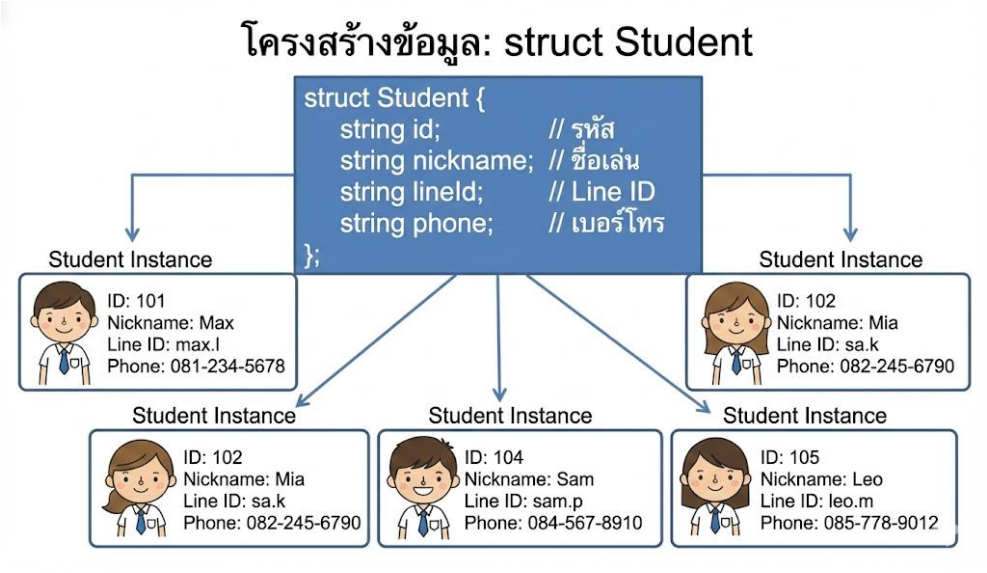
การเขียนโปรแกรมเชิงวัตถุ

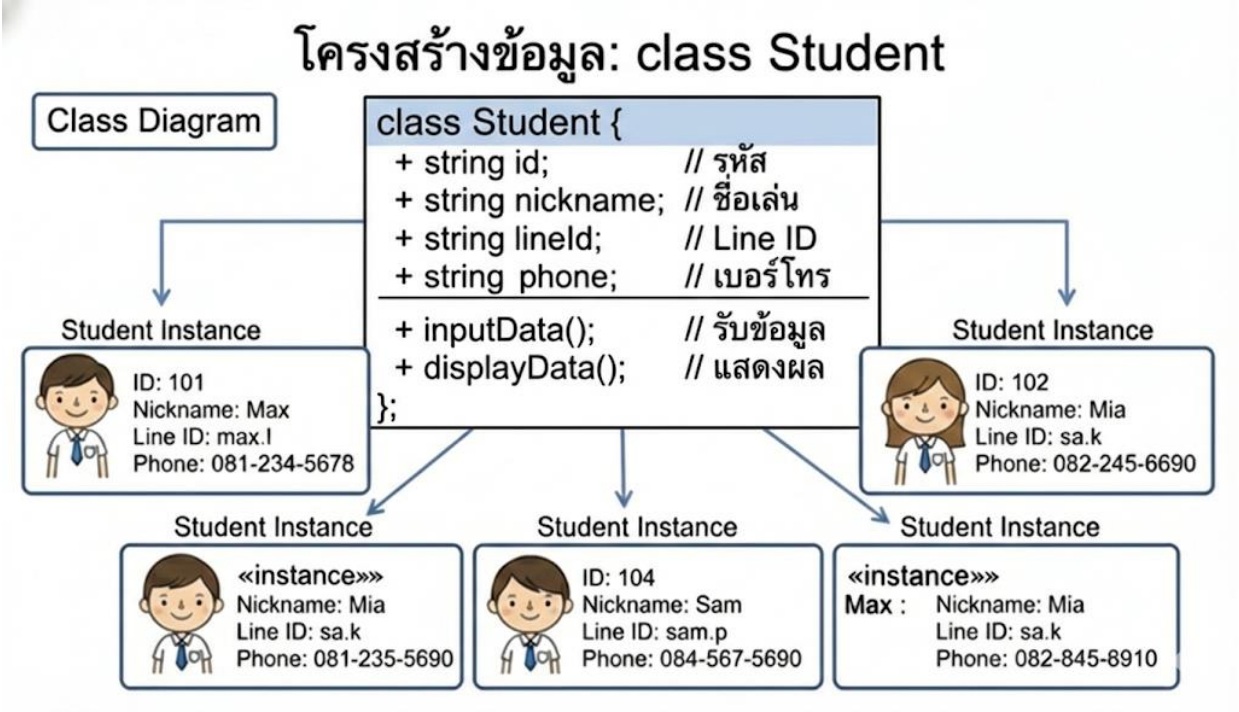
อาจารย์ สุทิศ องอาจ

sutit@mut.ac.th

ตัวแปรเดียว ,Struct ,Arrary และ Class







การประกาศและใช้งาน

การสร้าง Struct

```
struct Student {
    string studentID;
    string nickname;
    string lineID;
    string phoneNumber;
};
```

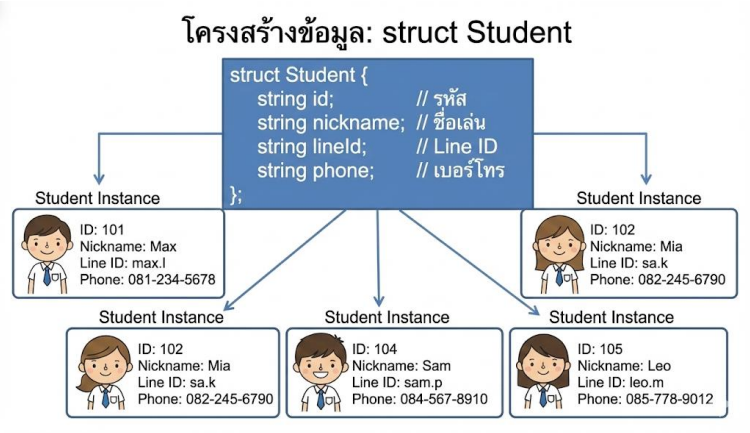
เป็นการกำหนดแม่แบบ (Template) ของข้อมูล

การนำไปใช้

```
int main() {
    // สร้าง Array ของ Struct ขนาด 5
    Student students[5];
}
```

เราจะได้ตัวแปรกลุ่ม students ที่เก็บข้อมูลนักศึกษาได้ 5 คน โดยแต่ละคนมีข้อมูลครบทั้ง 4 อย่าง

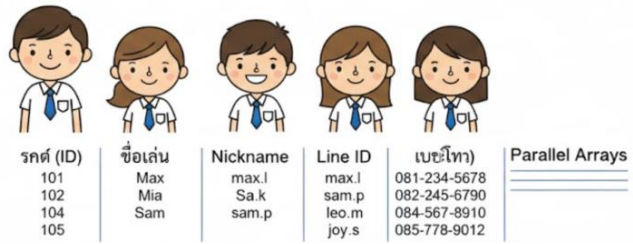
ประกาศตัวแปรโครงสร้างทันที




```
// กำหนดโครงสร้าง struct
struct Student {
    string studentID;
    string nickname;
    string lineID;
    string phoneNumber;
};

int main() {
    Student students[5]; // สร้างอาร์เรย์ของ struct ขนาด 5
```

```
struct Student {
string studentID;
string nickname;
string lineID;
string phoneNumber;
} students[5];  // ประกาศตัวแปรโครงสร้างทันที
```

```
const int SIZE = 5;

    string studentIDs[SIZE];
    string nicknames[SIZE];
    string lineIDs[SIZE];
    string phoneNumbers[SIZE];
```



การเข้าถึงและใช้งานข้อมูล โครงสร้าง และ class

Dot Operator (.)

ใช้เครื่องหมายจุด เพื่อเข้าถึงสมาชิกภายใน Struct

```
struct_var.member
```

การรับค่า (Input)

รับค่าเข้าไปเก็บในตัวแปรย่อยแต่ละตัว

```
cin >> students[i].studentID;
```

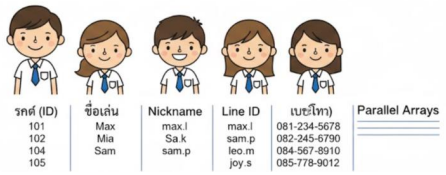
การแสดงผล (Output)

ดึงค่าจากตัวแปรย่อยมาแสดงผล

```
cout << students[i].nickname;
```

ตัวอย่างที่ : ใช้ struct Student → sort ง่าย ดูสะอาด และปลอดภัย

(เป็นบทสรุปจากปัญหา Parallel Arrays)



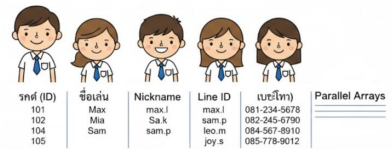

```
struct Student {
    string id;
    string nickname;
    string lineId;
    string phone;
};
```

```
const int SIZE = 5;
Student students[SIZE];

    // รับข้อมูล
  for (int i = 0; i < SIZE; i++) {
        cout << "=== Student " << i + 1 << " ===" << endl;
        cout << "Student ID: ";
        cin >> students[i].id;

        cout << "Nickname: ";
        cin >> students[i].nickname;

        cout << "Line ID: ";
        cin >> students[i].lineId;

        cout << "Phone Number: ";
        cin >> students[i].phone;
        cout << endl;
    }
```

```
const int SIZE = 5;

    string studentIDs[SIZE];
    string nicknames[SIZE];
    string lineIDs[SIZE];
    string phoneNumbers[SIZE];

  // รับข้อมูล
    for (int i = 0; i < SIZE; i++) {
        cout << "=== Student " << i + 1 << " ===" << endl;

        cout << "Student ID: ";
        cin >> studentIDs[i];

        cout << "Nickname: ";
        cin >> nicknames[i];

        cout << "Line ID: ";
        cin >> lineIDs[i];

        cout << "Phone Number: ";
        cin >> phoneNumbers[i];

        cout << endl;
  }
```

ตัวอย่างที่ : ใช้ struct Student → sort ง่าย ดูสะอาด และปลอดภัย

(เป็นบทสรุปจากปัญหา Parallel Arrays)




```
// sort ตาม ID (น้อย → มาก)
 for (int i = 0; i < SIZE - 1; i++) {
       for (int j = 0; j < SIZE - 1 - i; j++) {
           if (students[j].id > students[j + 1].id) {
               swapStudent(students[j], students[j + 1]); //1 บรรทัด
    }
  }
 }
```

```
// ===== Sort ตาม ID (น้อย → มาก)=====
for (int i = 0; i < SIZE - 1; i++) {
    for (int j = 0; j < SIZE - 1 - i; j++) {
         if (studentIDs[j] > studentIDs[j + 1]) {
            // สลับ ID
            swapString(studentIDs[j], studentIDs[j + 1]);
            // สลับข้อมูลที่เกี่ยวข้องทั้งหมด
            swapString(nicknames[j], nicknames[j + 1]);
            swapString(lineIDs[j], lineIDs[j + 1]);
            swapString(phoneNumbers[j], phoneNumbers[j + 1]);
            }
     }
}
```

```
// สามารถใช้ swap  ที่เป็น library  ได้เลย แต่ขอเขียนฟังก์ชัน swap  เองเพื่อความ
เข้าใจ
void swapString(string& a, string& b) {
   string temp = a;
   a = b;
   b = temp;
}
```

ทำไมต้อง OOP? (Why OOP?)

Structural Programming

- เน้นขั้นตอนกระบวนการ (Process-oriented)
- แก้ปัญหาเฉพาะด้าน เหมาะกับโปรแกรมขนาดเล็ก
- ยากต่อการนำกลับมาใช้ใหม่ (Not Reusable)

Object-Oriented Programming

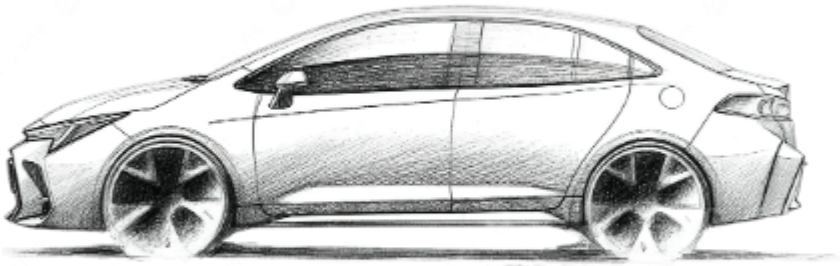
- มองปัญหาเป็น "วัตถุ" (Object) ที่ทำงานร่วมกัน
- เหมาะกับซอฟต์แวร์ขนาดใหญ่ (Large Scale)
- โค้ดมีความยืดหยุ่น นำกลับมาใช้ใหม่ได้ง่าย
- ง่ายต่อการ Debug และดูแลรักษา

คลาส และ วัตถุ (Class & Object)

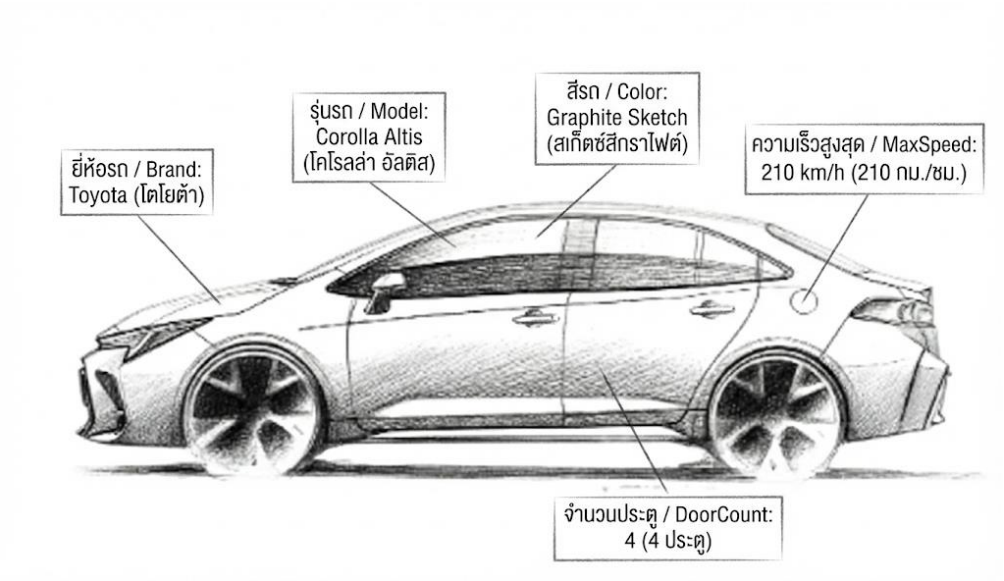
แนวคิดหลักคือการแยกแยะระหว่าง "พิมพ์เขียว" และ "สิ่งของจริง"

- **Class (คลาส)**: คือแบบแปลนหรือพิมพ์เขียว (Blueprint) ที่กำหนดคุณสมบัติ (Data) และพฤติกรรม (Method)
- **Object (วัตถุ)**: คือสิ่งที่ถูกสร้างขึ้นจากคลาส (Instance) เพื่อใช้งานจริง สามารถมีได้หลายชิ้นจากพิมพ์เขียวเดียวกัน

```
class Car { ... }; // Blueprint
Car myToyota; // Object
```

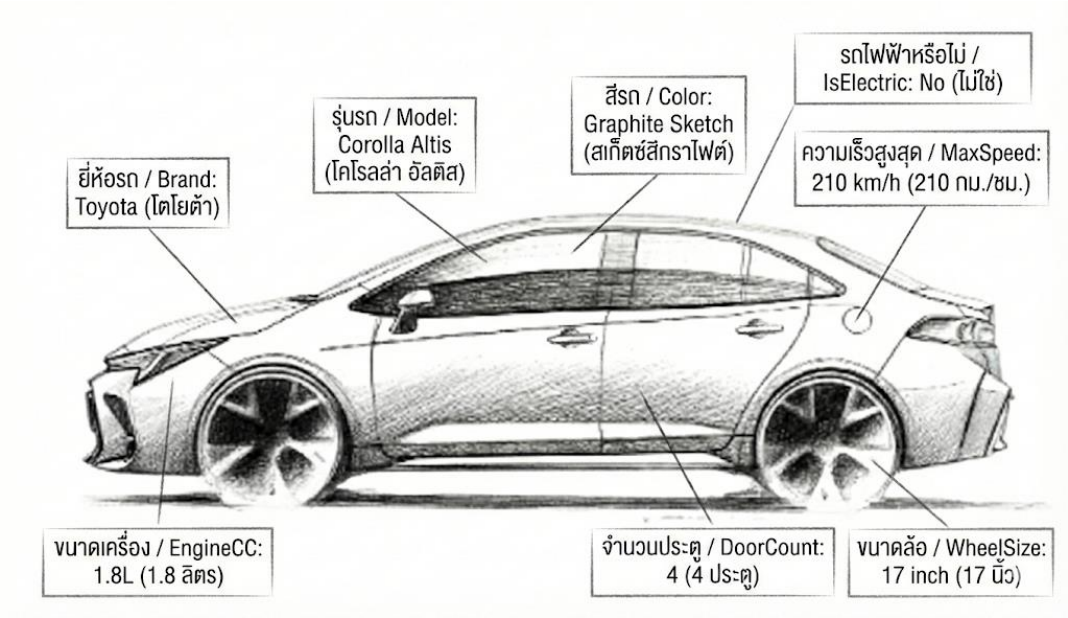


Class Car:Properties



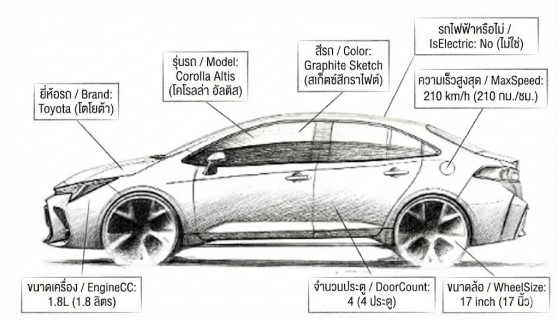

```
class Car
{
public:
    string brand;      // ยี่ห้อ
    string model;      // รุ่น
    string color;      // สี
    int maxSpeed;     // ความเร็วสูงสุด (km/h)
    int doorCount;    // จำนวนประตู
};
```

Class Car: Properties




```
#include <iostream>
#include <string>
using namespace std;

class Car
{
public:
    // Properties หลัก
    string brand;        // ยี่ห้อรถ
    string model;        // รุ่นรถ
    string color;        // สีรถ

    // Properties เพิ่มจากที่เลือก
    int wheelSize;       // ขนาดล้อ (นิ้ว)
    string fuelType;    // ประเภทน้ำมัน (Gas, Electric, Hybrid)
    int engineCC;       // ขนาดเครื่องยนต์ (cc)
    bool isElectric;    // รถไฟฟ้าหรือไม่
};
```

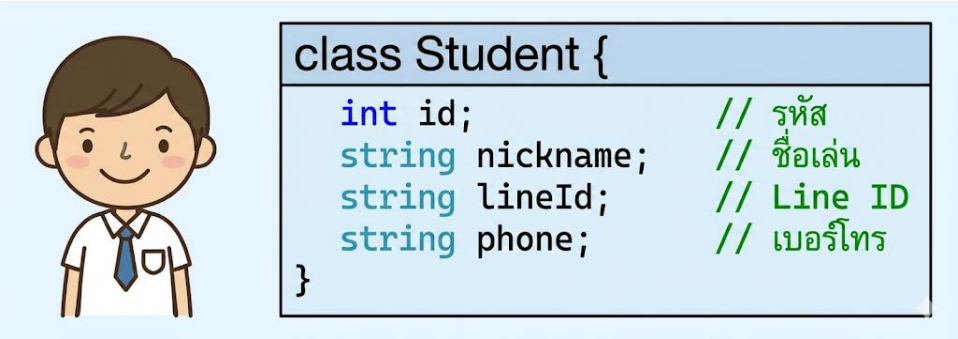
Class Car :Method




```
int main()
{
    Car c1;

    c1.brand = "Toyota";
    c1.model = "Corolla";
    c1.color = "White";
    c1.wheelSize = 17;
    c1.fuelType = "Gas";
    c1.engineCC = 1800;
    c1.isElectric = false;

    c1.printInfo();

    return 0;
}
```

```
#include <iostream>
#include <string>
using namespace std;

class Car
{
public:
    // Properties หลัก
    string brand;       // ยี่ห้อรถ
    string model;       // รุ่นรถ
    string color;       // สีรถ

    // Properties เพิ่มจากที่เลือก
    int wheelSize;      // ขนาดล้อ (นิ้ว)
    string fuelType;   // ประเภทน้ำมัน (Gas, Electric, Hybrid)
    int engineCC;      // ขนาดเครื่องยนต์ (cc)
    bool isElectric;   // รถไฟฟ้าหรือไม่

    // Method แสดงข้อมูลรถ
    void printInfo()
  {
        cout << "Brand     : " << brand << endl;
        cout << "Model     : " << model << endl;
        cout << "Color     : " << color << endl;
        cout << "WheelSize : " << wheelSize << " inch" << endl;
        cout << "FuelType  : " << fuelType << endl;
        cout << "EngineCC  : " << engineCC << " cc" << endl;
        cout << "Electric  : " << (isElectric ? "Yes" : "No") << endl;
  }
};
```

Class Student:Properties




```
#include <iostream>
#include <string>
using namespace std;

class Student
{
public:
    // Properties (ตัวแปรของคลาส)
    int id;                // รหัส
    string nickname;      // ชื่อเล่น
    string lineId;        // Line ID
    string phone;         // เบอร์โทร

};
```

Class Student:Properties

```
#include <iostream>
#include <string>
using namespace std;

class Student
{
public:
    // Properties (ตัวแปรของคลาส)
    int id;                 // รหัส
    string nickname;        // ชื่อเล่น
    string lineId;          // Line ID
    string phone;           // เบอร์โทร
    int height;             // ความสูง (cm)
    int weight;             // น้ำหนัก (kg)
    string hairColor;       // สีผม
    bool longHair;          // ผมยาว (true / false)

};
```

Class Student: `Method`

Class Student: Method

```
    // Method 1 : รับข้อมูลนักเรียน
    void input()
  {
        cout << "Student ID: ";
        cin >> id;

        cout << "Nickname: ";
        cin >> nickname;

        cout << "Line ID: ";
        cin >> lineId;

        cout << "Phone: ";
        cin >> phone;

        cout << "Height (cm): ";
        cin >> height;

        cout << "Weight (kg): ";
        cin >> weight;

        cout << "Hair Color: ";
        cin >> hairColor;

        cout << "Long Hair (1 = Yes, 0 = No): ";
        cin >> longHair;
  }

};
```

```
    // Method 2 : แสดงข้อมูลนักเรียน
    void print()
  {
        cout << "\n----- Student Info -----" << endl;
        cout << "ID        : " << id << endl;
        cout << "Nickname  : " << nickname << endl;
        cout << "Line ID   : " << lineId << endl;
        cout << "Phone     : " << phone << endl;
        cout << "Height    : " << height << " cm" << endl;
        cout << "Weight    : " << weight << " kg" << endl;
        cout << "HairColor : " << hairColor << endl;
        cout << "LongHair  : " << (longHair ? "Yes" : "No") << endl;
  }

    // Method 3 : คำนวณ BMI อย่างง่าย
    double calcBMI()
  {
        double h = height / 100.0;   // แปลง cm → m
        return weight / (h * h);
  }
};
```

Class Student: Method

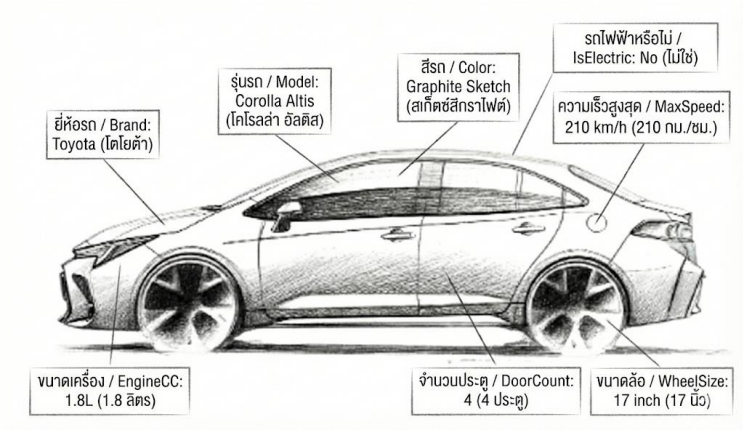
```
int main()
{
    Student s1;

    s1.input();          // รับข้อมูล
    s1.print();          // แสดงข้อมูล

    cout << "BMI : " << s1.calcBMI() << endl;

    return 0;
}
```

```
class Student
{
public:
    // Properties
    int id;              // รหัส
    // Method 1 : รับข้อมูลนักเรียน
 void input()
  {
   //
  }
    // Method 2 : แสดงข้อมูลนักเรียน
 void print()
  {
   //
  }
    // Method 3 : คำนวณ BMI อย่างง่าย
 double calcBMI()
  {
        double h = height / 100.0;   // แปลง cm → m
        return weight / (h * h);
  }
};

int main()
{
    Student s1;

    s1.input();          // รับข้อมูล
    s1.print();          // แสดงข้อมูล

    cout << "BMI : " << s1.calcBMI() << endl;

    return 0;
}
```

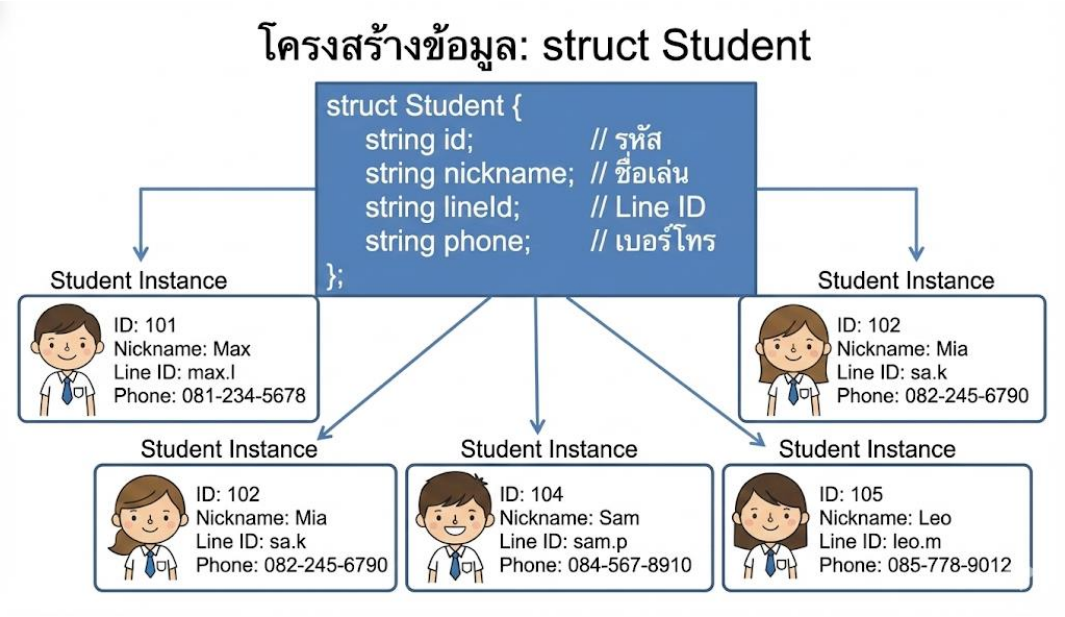
โจทย์:

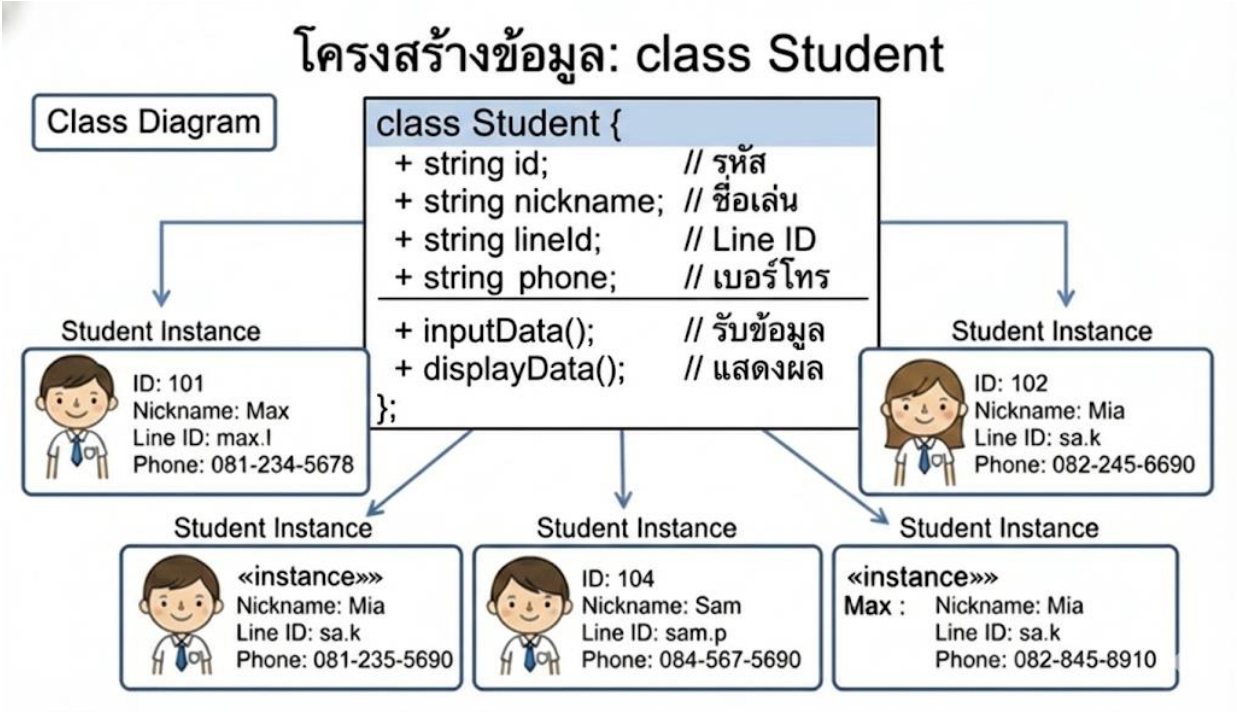
โจทย์: ลองมองไปรอบตัว แล้วระบุสิ่งที่เป็น "Object" มา 2 อย่าง พร้อมบอกว่ามันมีข้อมูล (Data) และพฤติกรรม (Behavior) อะไรบ้าง?

(ตัวอย่าง:รถ และ นักศึกษา)



Struct and Class





Struct =>> Class

```
struct Student {
    string id;
    string nickname;
    string lineId;
    string phone;
};
```

```
const int SIZE = 5;
Student students[SIZE];

   // รับข้อมูล
 for (int i = 0; i < SIZE; i++) {
       cout << "=== Student " << i + 1 << " ===" << endl;
       cout << "Student ID: ";
       cin >> students[i].id;

       cout << "Nickname: ";
       cin >> students[i].nickname;

       cout << "Line ID: ";
       cin >> students[i].lineId;

       cout << "Phone Number: ";
       cin >> students[i].phone;
       cout << endl;
   }
```

```
class Student {
public:
    string studentID;
    string nickname;
    string lineID;
    string phoneNumber;
};
```

```
const int SIZE = 5;
Student students[SIZE];
```

ต้องเขียนแบบไหน

Struct =>> Class

```
struct Student {
    string id;
    string nickname;
    string lineId;
    string phone;
};
```

```
class Student {
public:
    string studentID;
    string nickname;
    string lineID;
    string phoneNumber;
};
```

```
void printStudents(Student students[], int size) {
    for (int i = 0; i < size; i++) {
        cout << "ID       : " << students[i].id << endl;
        cout << "Nickname : " << students[i].nickname << endl;
        cout << "Line ID  : " << students[i].lineId << endl;
        cout << "Phone    : " << students[i].phone << endl;
        cout << "----------------------\n";
    }
}
```

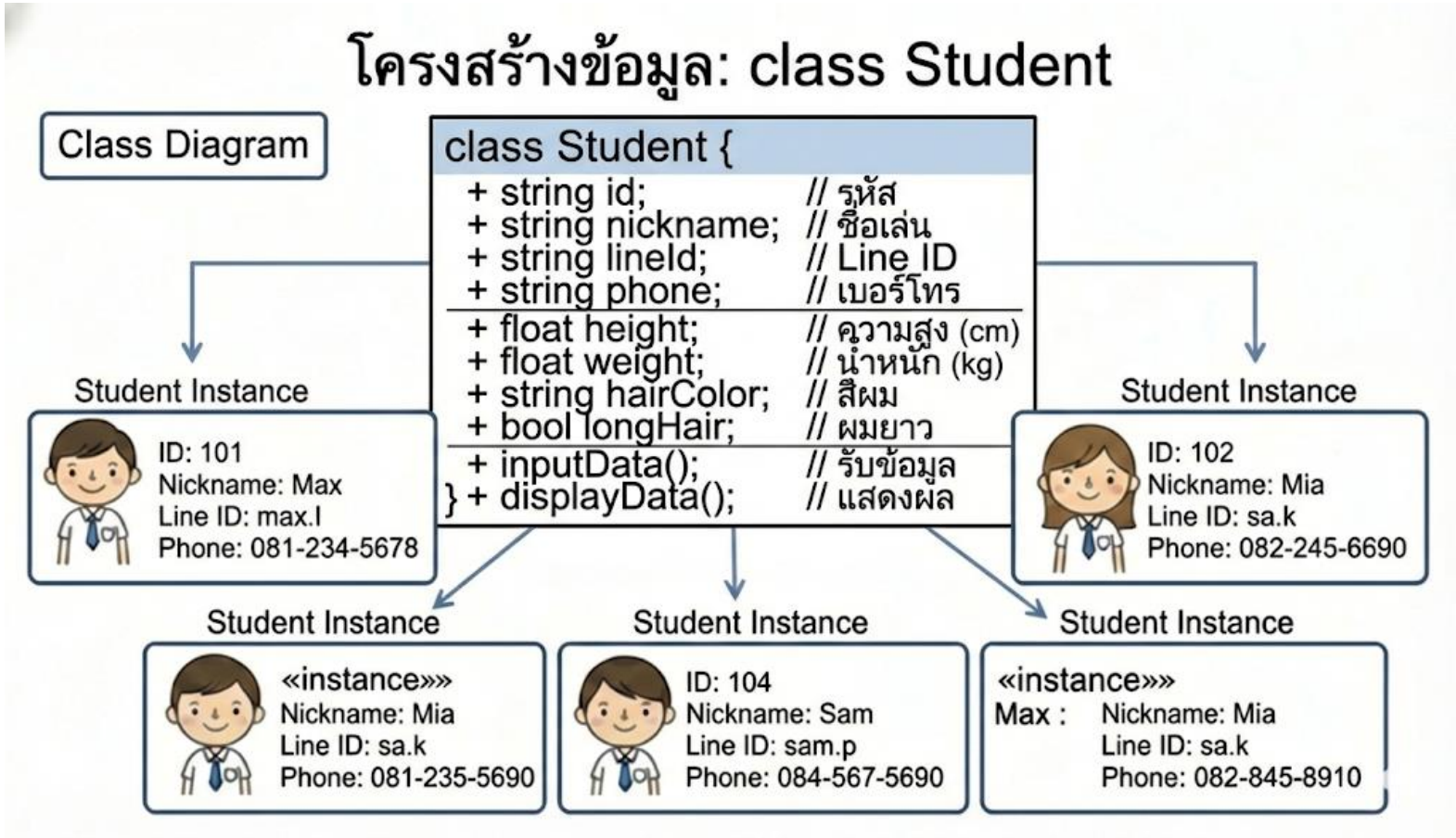
Struct =>> Class

```
struct Student {
    string id;
    string nickname;
    string lineId;
    string phone;
};
```

```
void printStudents(Student students[], int size) {
    for (int i = 0; i < size; i++) {
        cout << "ID       : " << students[i].id << endl;
        cout << "Nickname : " << students[i].nickname << endl;
        cout << "Line ID  : " << students[i].lineId << endl;
        cout << "Phone    : " << students[i].phone << endl;
        cout << "----------------------\n";
    }
}
```

```
class Student {
public:
    string id;
    string nickname;
    string lineId;
    string phone;

    void print() {
        cout << "ID       : " << id << endl;
        cout << "Nickname : " << nickname << endl;
        cout << "Line ID  : " << lineId << endl;
        cout << "Phone    : " << phone << endl;
        cout << "----------------------\n";
    }
};
```

Struct =>> Class



Sutit Ongart
Sutit@mut.ac.th
sutit.ongart@gmail.com

END