Design and Analysis of Data Structures and Algorithms

Warin Wattanapornprom Ph.D.

กฎการฟังบรรยาย

OK

ภาษา C++ พื้นฐาน

- **ตัวแปรแบบอาร์เรย์** (Array) หมายถึงตัวแปรซึ่งมีค่าได้หลายค่าโดยใช้ชื่ออ้างอิงเพียงชื่อเดียว ด้วยการใช้หมายเลขลำดับเป็นตัวจำแนกความแตกต่างของค่าตัวแปรแต่ละตัว ถ้าเราจะเรียกตัวแปรชนิดนี้ว่า "ตัวแปรชุด" ก็เห็นจะไม่ผิดนัก ตัวแปรชนิดนี้มีประโยชน์มาก
- **ลองคิดถึงค่าข้อมูลจำนวน 100 ค่า ที่ต้องการเก็บไว้ในตัวแปรจำนวน 100 ตัว อาจทำให้ต้องกำหนดตัวแปรที่แตกต่างกันมากถึง 100 ชื่อ กรณีอย่างนี้ควรจะทำอย่างไรดี**

ภาษา C++ พื้นฐาน

- **ด้วยการใช้คุณสมบัติอาร์เรย์ เราสามารถนำตัวแปรหลาย ๆ ตัวมาอยู่รวมเป็นชุดเดียวกันได้ และสามารถเรียกใช้ตัวแปรทั้งหมดโดยระบุผ่านชื่อเพียงชื่อเดียวเท่านั้น ด้วยการระบุหมายเลขลำดับ หรือ ดัชนี(index) กำกับตามหลังชื่อตัวแปร** ตัวแปรเพียงชื่อเดียวจึงมีความสามารถเทียบได้กับตัวแปรนับร้อยตัว พันตัว
- อาร์เรย์ใช้จองเนื้อที่ในหน่วยความจำ ตามที่ระบุขนาดหน่วยความจำในตัวแปรอาร์เรย์ ใช้ตัวระบุตำแหน่งเพื่อคำนวณระยะห่างไปของหน่วยความจำ

ภาษา C++ พื้นฐาน

- Array
- Declaring
  - **type arrayName [arraySize];**
- example

```
double balance[10];
```

- Declaring + initialization

```
double balance[5] = {1000.0, 2.0, 3.4, 17.0, 50.0};
```

- A

```
double balance[] = {1000.0, 2.0, 3.4, 17.0, 50.0};
```

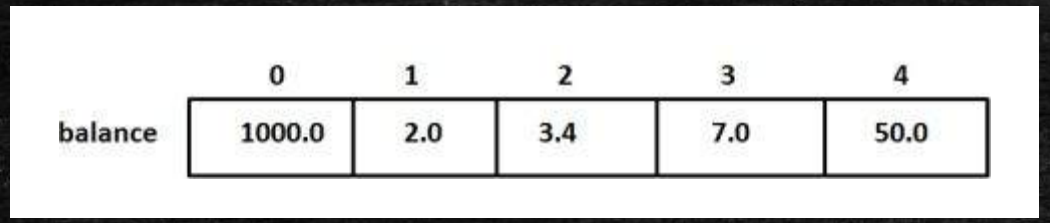
- จำ

```
balance[4] = 50.0;
```

ภาษา C++ พื้นฐาน

- Array

```
double balance[5] = {1000.0, 2.0, 3.4, 17.0, 50.0};
```



ภาษา C++ พื้นฐาน

- Array

```
#include <iostream>
using namespace std;
#include <iomanip>
using std::setw;

void main (){
        int n[10]; // n is an array of 10 integers
   // initialize elements of array n to 0
        for (int i=0;i<10;i++){
                n[i] = i + 100;
        }
        cout << "Element" << setw( 13 ) << "Value" << endl;
        for (int j=0;j<10;j++){
                cout << setw( 7 )<< j << setw( 13 ) << n[j] << endl;
        }
}
```

ภาษา C++ พื้นฐาน

- Array
- Output
  -
```
Element         Value
      0           100
      1           101
      2           102
      3           103
      4           104
      5           105
      6           106
      7           107
      8           108
      9           109
```

ภาษา C++ พื้นฐาน

- Multidimentional Array
- `type name[size1][size2]...[sizeN];`

```
int threedim[5][10][4];
```

```
int x=4;
int y=4;
type arrayName[x][y];
```

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

ภาษา C++ พื้นฐาน

- Multidimentional Array

```
int a[3][4] = {
        {0, 1, 2, 3} ,   /*  row indexed by 0 */
        {4, 5, 6, 7} ,   /*  row indexed by 1 */
        {8, 9, 10, 11}   /*  row indexed by 2 */
};
```

ภาษา C++ พื้นฐาน

- Multidimentional Array

```
#include <iostream>
using namespace std;
void main (){

    int a[5][2]={{0,0},{1,2},{2,4},{3,6},{4,8}};

    for(int i=0;i<5;i++)
       for(int j=0;j<2;j++){
        cout << "a[" << i << "][" << j << "]: ";
        cout << a[i][j]<< endl;
       }
}
```

ภาษา C++ พื้นฐาน

- Multidimentional Array
- Output
- ```
  a[0][0]: 0
  a[0][1]: 0
  a[1][0]: 1
  a[1][1]: 2
  a[2][0]: 2
  a[2][1]: 4
  a[3][0]: 3
  a[3][1]: 6
  a[4][0]: 4
  a[4][1]: 8
  ```
  ```

ภาษา C++ พื้นฐาน

- การบ้าน โบนัส 2 คะแนน ให้เขียนโปรแกรมสร้างแผนที่ขนาด 10x10 ด้วย array แล้วพิมพ์แผนที่ออกมากำหนดให้
  - 0 แทนพื้นดินโล่งๆ
  - 1 แทนต้นไม้
  - 2 แทนฮีโร่
  - 3 แทนมอนสเตอร์
- จงคำนวณระยะทางจาก Hero ไปถึง Monster

ภาษา C++ พื้นฐาน

- 1 แทนต้นไม้
- 2 แทนฮีโร่
- 3 แทนมอนสเตอร์

วัตถุ	X	Y
ต้นไม้	ตำแหน่งใดๆ	ตำแหน่งใดๆ
ฮีโร่	วันเกิด 1 ถึง 7	เลขท้ายรหัสนักศึกษา
มอนสเตอร์	หมายเลข 10 – วันเกิด	รหัสนักศึกษาหลักที่สองจากสุดท้าย

Distance

- การวัดระยะแบบยูคลิด (Euclidean distance)
- การวัดระยะแบบแมนฮัตตัน (Manhattan distance)
- การวัดระยะแบบเชบิเชฟ (Chebychev distance)

http://th.wikipedia.org/wiki/การแบ่งกลุ่มข้อมูล

Euclidean distance

- One dimension

$$d(p, q) = \sqrt{(p - q)^2}.$$

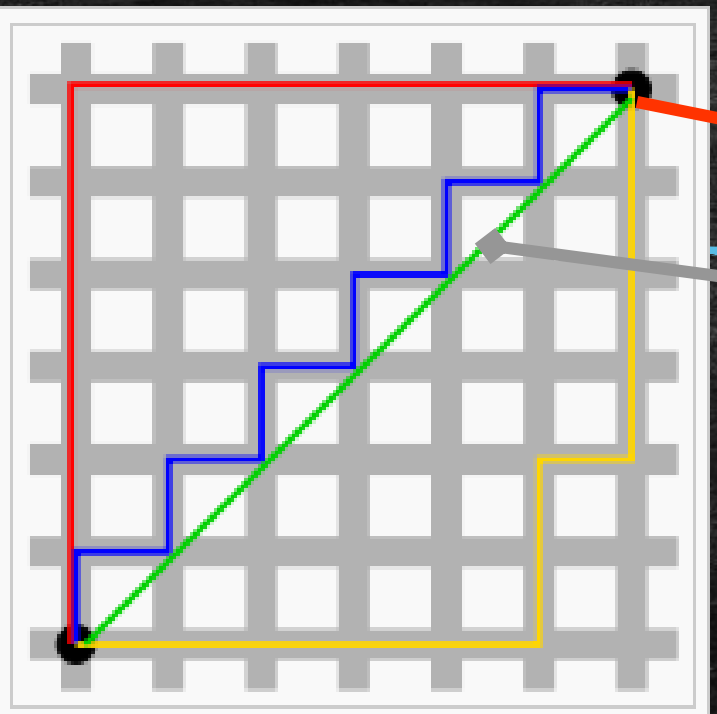
- Two dimensions

$$d(p, q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}.$$

- Higher dimensions

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2}.$$

http://en.wikipedia.org/wiki/Euclidean_distance



Manhattan distance versus Euclidean distance: The red, blue, and yellow lines have the same length (12) in both Euclidean and taxicab geometry. In Euclidean geometry, the green line has length $6\times\sqrt{2} \approx 8.48$, and is the unique shortest path. In taxicab geometry, the green line's length is still 12, making it no shorter than any other path shown.

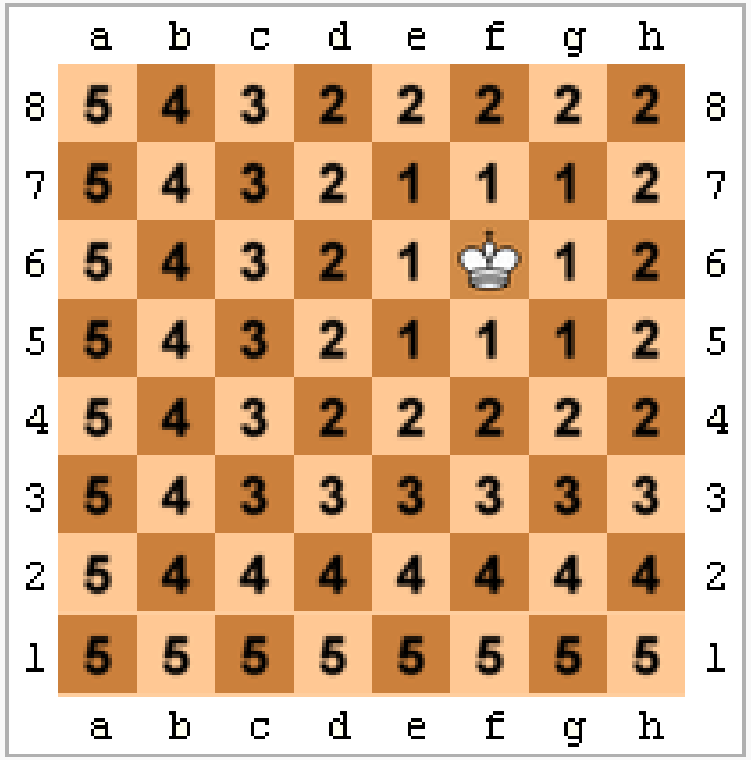
http://en.wikipedia.org/wiki/Manhattan_distance

Taxicab Geometry (Manhattan distance)

(Euclidean distance)

Taxicab geometry, considered by Hermann Minkowski in the 19th century, is a form of geometry in which the usual metric of Euclidean geometry is replaced by a new metric in which the distance between two points is the sum of the (absolute) differences of their coordinates. The taxicab metric is also known as **rectilinear distance**, **L1 distance** or **L1 norm** (see Lp space), **city block distance**, **Manhattan distance**, or Manhattan length, with corresponding variations in the name of the geometry.[1] The last name alludes to the grid layout of most streets on the island of Manhattan, which causes the shortest path a car could take between two points in the city to have length equal to the points' distance in taxicab geometry.

Chebyshev distance



The Chebyshev distance between two spaces on a chess board is the minimum number of moves a king requires to move between them. Above are the Chebyshev distances of each square from the square f6.

In mathematics, **Chebyshev distance** (or **Tchebychev distance**), or L∞ metric[1] is a metric defined on a vector space where the distance between two vectors is the greatest of their differences along any coordinate dimension.[2] It is named after Pafnuty Chebyshev. It is also known as **chessboard distance**, since in the game of chess the minimum number of moves needed by a king to go from one square on a chessboard to another equals the Chebyshev distance between the centers of the squares, if the squares have side length one, as represented in 2-D spatial coordinates with axes aligned to the edges of the board

http://en.wikipedia.org/wiki/Chebyshev_distance

Taxicab Geometry in Game

- OGRE battle / FF Tactics

ภาษา C, C++, Java, C# พื้นฐาน

- ตัวอย่าง เกิดวันอาทิตย์ รหัส xxxxx59

```
- 02000000000 //Hero(1,9)
  0000000000
  0000000000
  0000000000
  0000000003 //Monster(9,5)
  0000000000
  00000100000 //Tree(r,r)
  0000000000
  0000000000
  0000000000
  Taxicab = abs(9-1)+abs(5-9)=12
  Euclidean = sqrt(pow(9-1,2)+pow(5-9,2))=22.9
  Chebyshev = if(abs(9-1)>abs(5-9))
                {d=abs(9-1)}
              else{d=abs(5-9)}=8
```

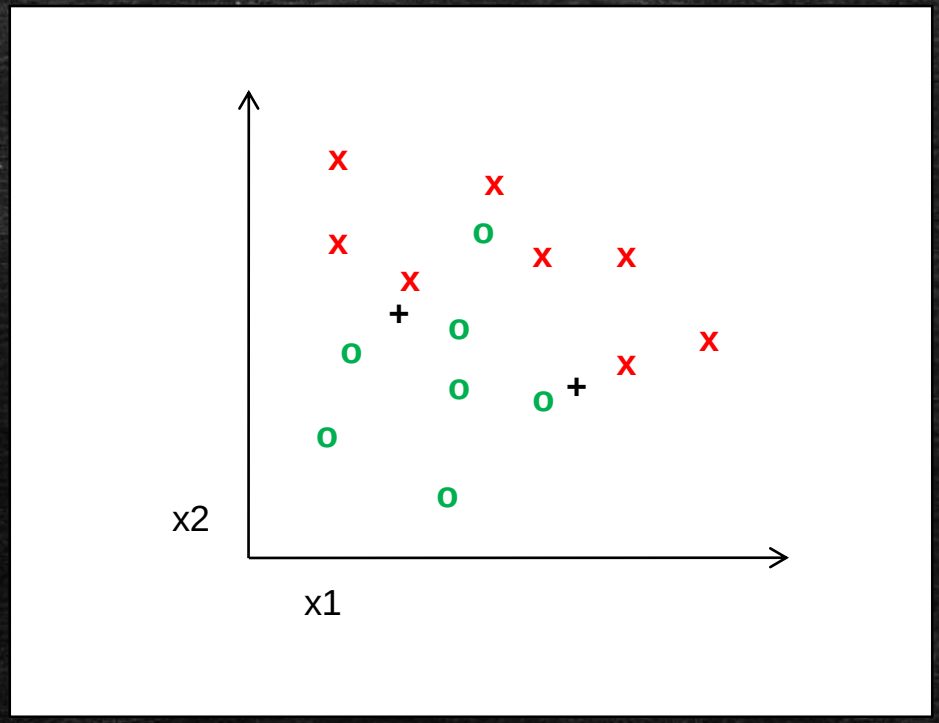
วัตถุ	X	Y
ต้นไม้	Rand(9)	Rand(9)
ฮีโร่	1	9
มอนสเตอร์	10-1=9	5

Simple recommender system with KNN

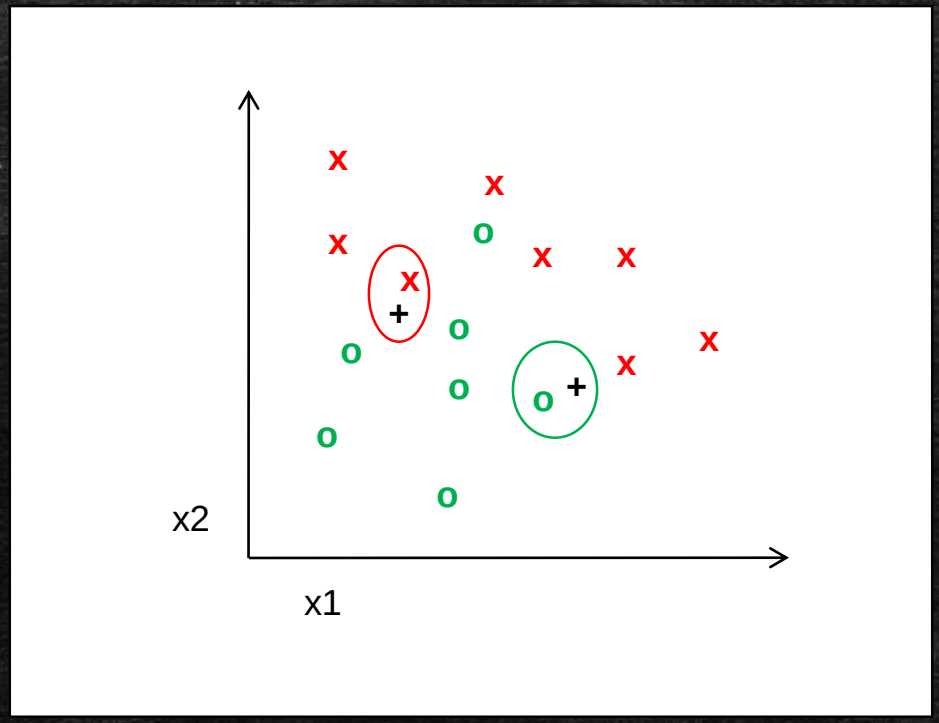
No	Feature1	Feature2	Feature3	...	FeatureN	Class
1	4	3	2		4	X
2	5	6	5		4	X
3	2	3	7		6	Y
4	5	5	2		7	Y
5	4	9	7		8	X
6	2	7	2		2	Y

No	Feature1	Feature2	Feature3	...	FeatureN	Class
1	3	5	2		4	?
2	6	6	5		1	?
3	7	3	4		4	?

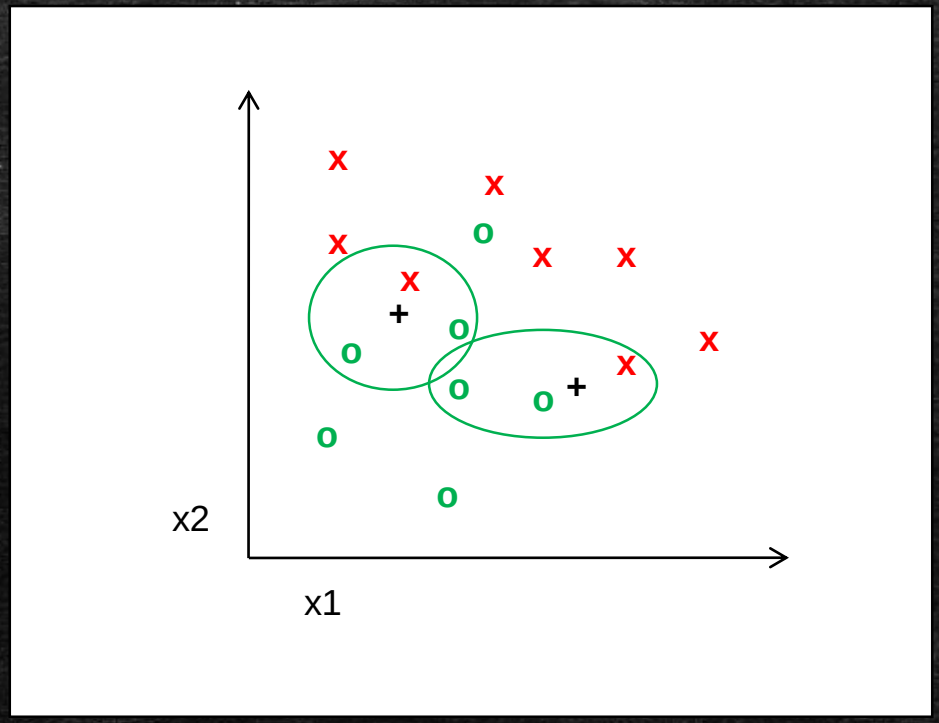
K-nearest neighbor



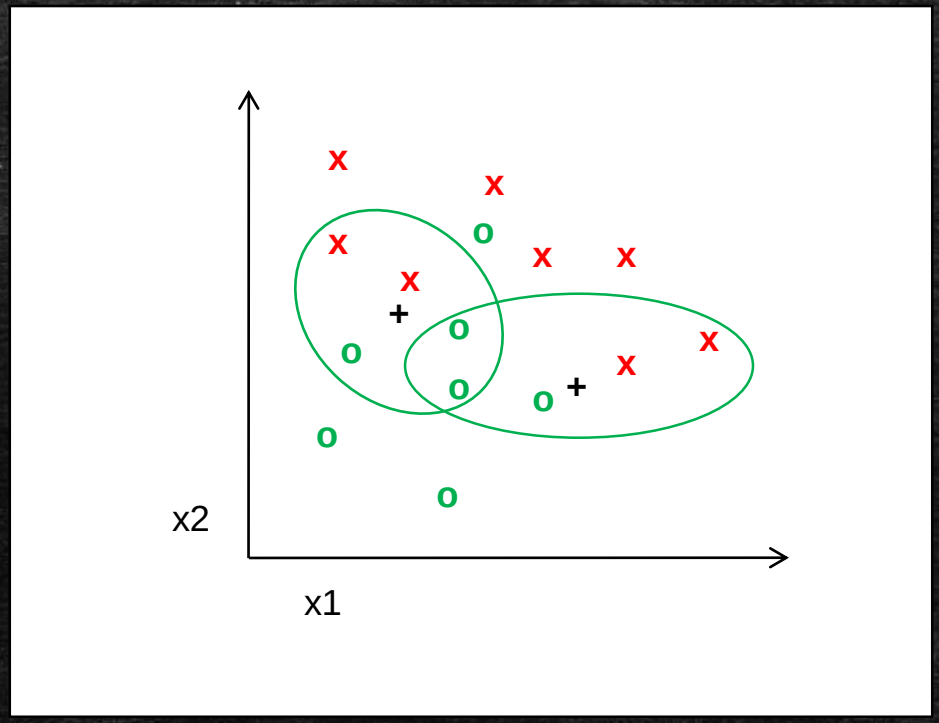
1-nearest neighbor



3-nearest neighbor



5-nearest neighbor



การบ้าน 5 คะแนน

- เขียนโปรแกรมหา Class จากการวัดระยะ Euclidean Distance เลือกโหวตจากตัวที่ใกล้ที่สุดสามตัว หรือจำนวนเลขคี่

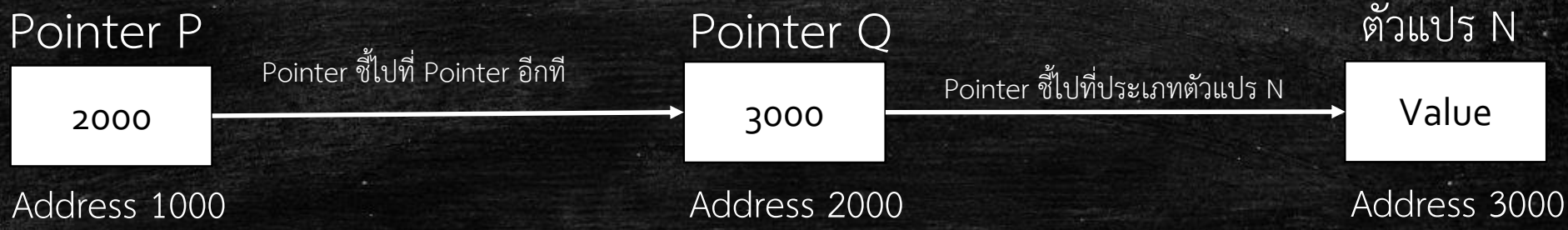
Pointer

- Pointer (ตัวชี้) เป็นตัวแปรชนิดพิเศษในภาษา C ทำหน้าที่เก็บตำแหน่งที่อยู่ (Address) ของตัวแปรชนิดอื่นๆ ที่อยู่ในหน่วยความจำ แทนที่จะเก็บข้อมูลเหมือนกันตัวแปรพื้นฐานชนิดอื่นๆ
- ตัวแปร pointer มีลักษณะคล้ายตัวแปรตารางอาร์เรย์ แต่ที่แตกต่างกันคือ ตัวแปรตารางอาร์เรย์จะเก็บเฉพาะค่าต่างๆ ที่เป็นชนิดกันเดียวกับตัวแปรอาเรย์ แต่ตัวแปร pointer จะเก็บเฉพาะค่าตำแหน่ง Address ตัวแปรเท่านั้น
- Array เองก็สามารถสร้างจาก pointer ได้โดยการชี้ไปที่ตำแหน่งที่ต้องการแล้วจองขนาดของตัวแปรคูณด้วยขนาดของ Array
- Array ใช้ประโยชน์จาก index addressing mode
  ส่วน Pointer เป็นการใช้ประโยชน์จาก indirect addressing mode
- ตัวแปร Pointer ในภาษา C++ จะมีเครื่องหมายดอกจันทร์ * (Asterisk) นำหน้าตัวแปร

Pointer

- ประเภทของ Pointer
  - Direct Pointer

  - Indirect Pointer



ภาษา C, C++, Java, C# พื้นฐาน

- **Address**
  - **Address of variables**

```
#include <iostream>
using namespace std;
void main (){
    int  var1;
    float var2[10];
    cout << "Address of var1 variable:";
    cout << &var1 << endl;
    cout << "Address of var2 variable:";
    cout << &var2 << endl;
}
```

- **Address of var1 variable: 0xbfebd5c0**
  Address of var2 variable: 0xbfebd5b6

ภาษา C, C++, Java, C# พื้นฐาน

- Pointer

```
int    *ip;    // pointer to an integer
double *dp;    // pointer to a double
float  *fp;    // pointer to a float
char   *ch     // pointer to character
```

ภาษา C, C++, Java, C# พื้นฐาน

- Pointer

```
#include <iostream>
using namespace std;
void main (){
   int  var = 20;
   int  *ip;         // pointer variable
   ip = &var;        // store address of var
   cout << "Value of var variable:";
   cout << var << endl;
   cout << "Address stored in ip variable:";
   cout << ip << endl;
   cout << "Value of *ip variable:";
   cout << *ip << endl;
}
```

ภาษา C, C++, Java, C# พื้นฐาน

- Pointer
  - Output

```
Value of var variable: 20
Address stored in ip variable: 0xbfc601ac
Value of *ip variable: 20
```

ภาษา C, C++, Java, C# พื้นฐาน

- Pointer to pointer

```cpp
#include <iostream>
using namespace std;
int main (){
   int  var;
   int  *ptr;
   int  **pptr;
   var = 3000;
   ptr = &var; //the address of var
   pptr = &ptr; // the address of ptr

   cout << "Value of var :" << var << endl;
   cout << "Value available at *ptr :" << *ptr << endl;
   cout << "Value available at **pptr :" << **pptr << endl;
}
```

ภาษา C, C++, Java, C# พื้นฐาน

- Pointer to pointer
  - **Output**
  - **Value of var :3000**
    Value available at *ptr :3000
    Value available at **pptr :3000

ภาษา C, C++, Java, C# พื้นฐาน

- Pointer

```
#include <iostream>
using namespace std;
int main ()
{
    int  arr[5]={2,1,3,5,4};
    int  *ptr;
    int  maxi=0;
    for(int i=0;i<5;i++){
        if(arr[i]>maxi){
                maxi=arr[i];
                ptr=&arr[i];        }}
        cout << "Max Value of arr :" << *ptr << endl;
        *ptr = 6;
        for(int j=0;j<5;j++){
        cout << "Array of index :"<<j <<"=" << arr[j] << endl;}
    return 0;
}
```

Structure

ข้อมูลพื้นฐาน (premitive data type) เช่น int, unsigned int, char, float, double, long เป็นต้น

ข้อมูลซับซ้อน (complex data type) ได้แก่ array, structure & union

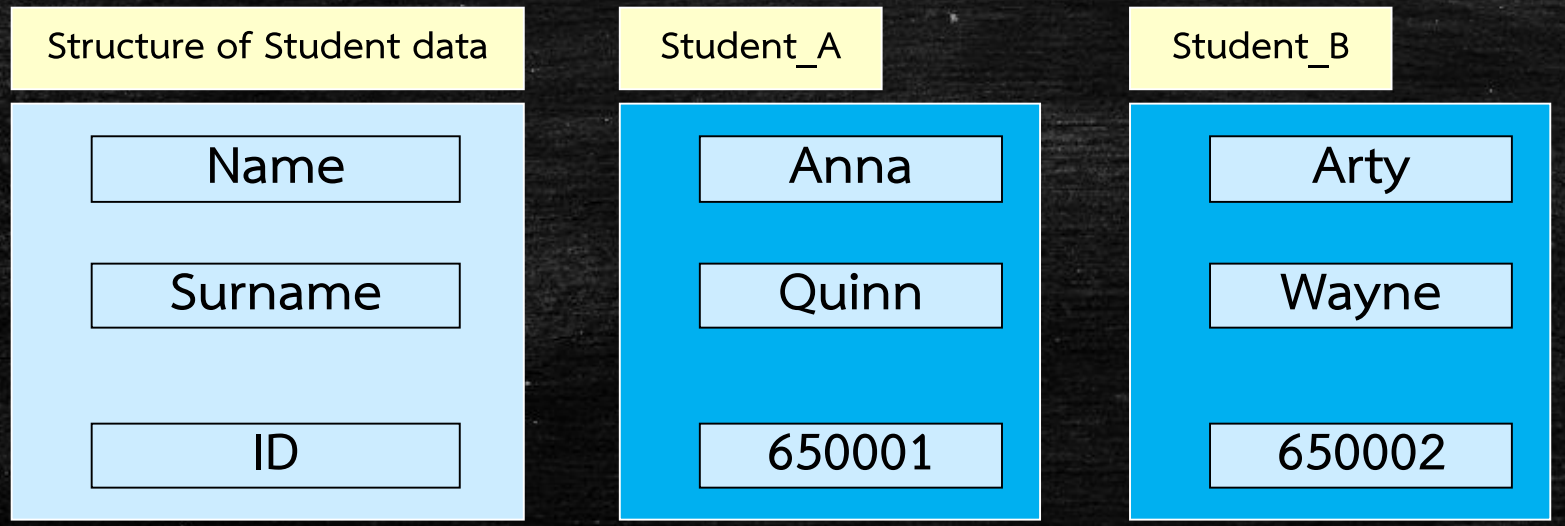
Structure หรือกลุ่มข้อมูลชนิดโครงสร้าง (Structure / struct) เป็นการประกาศหน่วยของข้อมูลใหม่ที่เกิดจากการรวมกลุ่มของข้อมูล เป็นโครงสร้าง โดยข้อมูลซึ่งเป็นสมาชิกของโครงสร้างใหม่ อาจมีหลายตัว และเป็น*ชนิดเดียวกันหรือต่างชนิด*กันก็ได้ เช่น มีสมาชิกเป็นจำนวนเต็ม ทศนิยม และอักขระ ได้

Structure อาจมีสมาชิกที่เป็น Array หรือแม้แต่ Structure เองก็ยังได้

Structure

ประโยชน์สตรัคเจอร์ (Structure)

- **เพื่อกำหนดหน่วยข้อมูลใหม่ ให้เหมาะสมกับข้อมูลที่ต้องการเก็บ**
  - เช่น การกำหนดหน่วยข้อมูลนักเรียน ที่ประกอบด้วยสมาชิกเป็น *ชื่อและนามสกุลที่เป็น* `string (char [ ])` กับ*รหัสนักศึกษาที่เป็นตัวเลข* `(int)` ซึ่งเราสามารถกำหนดให้ตัวแปรนักเรียน A กับ B มีโครงสร้างแบบหน่วยข้อมูลที่สร้างขึ้นนี้ได้ดังรูป



Structure

Structure definiton & declaration

- การประกาศตัวแปรของกลุ่มข้อมูลชนิดโครงสร้าง structure มี 2 ส่วน

    1. *Structure definition* การนิยามกลุ่มข้อมูลที่สร้างใหม่ ว่ามีสมาชิกอะไรบ้าง เป็นชนิดใด ข้อมูลย่อยในสตรัคเจอร์เรียกว่า **Field**

    2. *Structure declaration* ประกาศ*ตัวแปร*สำหรับกลุ่มข้อมูลที่สร้างขึ้นมา

```
struct name
   {      type varname1;
          type varname2[size of array];
          ……
          type varname n;
   } structurevarname;
```

Structure

```
struct address
{    char name[30];
     char detail[50];
     int  age;
     char telephone[10];
};
```

สมาชิกข้อมูลภายในมี 4 ตัว

ชื่อของ Structure

```
struct student
{    char name[30];
     char surname[50];
     int  Id;
};
```

ภาษา C, C++, Java, C# พื้นฐาน

- Data Structure

```
struct Books{
                char  title[50];
                char  author[50];
                char  subject[100];
                int   book_id;
}book;
```

ภาษา C, C++, Java, C# พื้นฐาน

- Data Structure

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

int main( ){
   struct Books Book1;        // Declare Book1 of type Book
   struct Books Book2;        // Declare Book2 of type Book
   strcpy( Book1.title, "Learn C++ Programming");
   strcpy( Book1.author, "Chand Miyan");
   strcpy( Book1.subject, "C++ Programming");
   Book1.book_id = 6495407;
   strcpy( Book2.title, "Telecom Billing");
   strcpy( Book2.author, "Yakit Singha");
   strcpy( Book2.subject, "Telecom");
   Book2.book_id = 6495700;
   cout << "Book 1 title :" << Book1.title <<endl;
   cout << "Book 1 author :" << Book1.author <<endl;
   cout << "Book 1 subject :" << Book1.subject <<endl;
   cout << "Book 1 id :" << Book1.book_id <<endl;
   cout << "Book 2 title :" << Book2.title <<endl;
   cout << "Book 2 author :" << Book2.author <<endl;
   cout << "Book 2 subject :" << Book2.subject <<endl;
   cout << "Book 2 id :" << Book2.book_id <<endl;
   return 0;
}
```

ภาษา C, C++, Java, C# พื้นฐาน

- Output

```
Book 1 title : Learn C++ Programming
Book 1 author : Chand Miyan
Book 1 subject : C++ Programming
Book 1 id : 6495407
Book 2 title : Telecom Billing
Book 2 author : Yakit Singha
Book 2 subject : Telecom
Book 2 id : 6495700
```

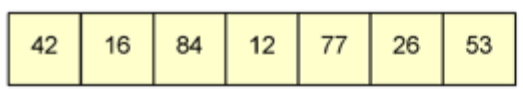
ภาษา C, C++, Java, C# พื้นฐาน

```
#include <iostream>
#include <cstring>
using namespace std;
void printStudent( struct Students student );

struct Students{
   char  name[50];
   int   score;
};

int main( ){
          struct Students student[10];
          strcpy( student[0].name, "John");
          student[0].score = 95;
          strcpy( student[1].name, "Mary");
          student[1].score = 90;
          printStudent( student[0] );
          printStudent( student[1] );
      return 0;
     }

     void printStudent( struct Students student ){
          std::cout << "Name :" << student.name ;
          std::cout << " Score :" << student.score <<endl;
     }
```

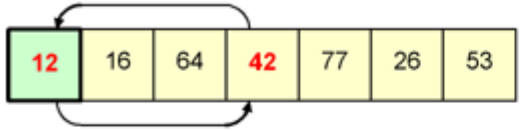
โบนัส 2 คะแนน พยายามใช้ structure

- กำหนดให้มีโครงสร้างข้อมูลนักศึกษา 10 คน
- จงเขียน function ดังนี้
  - MaxStudent คนที่ได้คะแนนสูงสุด
  - MinStudent คนที่ได้คะแนนต่ำสุด
  - AvrScore คะแนนเฉลี่ย
  - ModeScore ฐานนิยม
  - MedianScore มัธยฐาน
  - SDScore ค่าเบี่ยงเบนมาตรฐาน
- จงแสดงเกรดของนักศึกษาที่ได้ดังนี้
  - คะแนนมากกว่า avr+2*SD ได้เกรด A
  - คะแนนต่ำกว่า A มากกว่า avr+SD ได้เกรด B
  - คะแนนต่ำกว่า B มากกว่า avr ได้เกรด C
  - คะแนนต่ำกว่า C มากกว่า avr-SD ได้เกรด D
  - ต่ำกว่านั้นได้ F

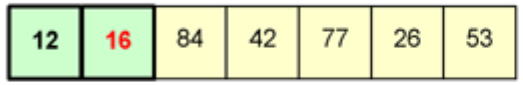
การเรียงลำดับอย่างง่าย



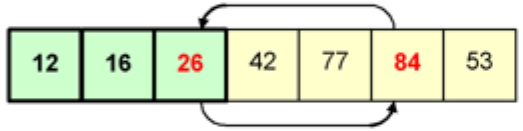
The array, before the selection sort operation begins.



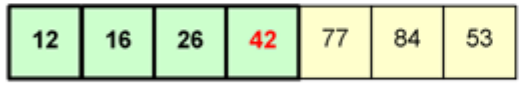
The smallest number (**12**) is swapped into the first element in the structure.



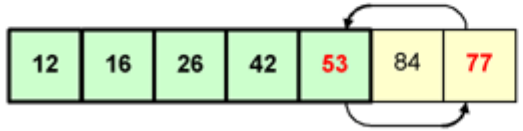
In the data that remains, **16** is the smallest; and it does not need to be moved.



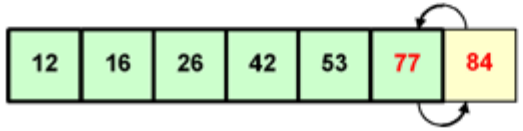
26 is the next smallest number, and it is swapped into the third position.



42 is the next smallest number; it is already in the correct position.



53 is the smallest number in the data that remains; and it is swapped to the appropriate position.



Of the two remaining data items, **77** is the smaller; the items are swapped. *The selection sort is now complete.*

พื้นฐานทางคณิตศาสตร์

เนื้อหาด้านอัลกอริทึมจำเป็นต้องใช้พื้นฐานทางคณิตศาสตร์ โดย ส่วนใหญ่จะเน้นไปที่หลักการของ Discrete Mathematic
ส่วนหนึ่งที่มีการใช้งานกันบ่อยประกอบด้วย

- เลขยกกำลังและฟังก์ชั่น Exponential
- ฟังก์ชั่น Logarithm
- ฟังก์ชั่น Factorial
- การ Modular

เลขยกกำลังและฟังก์ชั่น Exponential

ฟังก์ชั่น Exponential เป็นส่วนหนึ่งของเลขยกกำลัง และมีคุณสมบัติหลายข้อที่ต้องอาศัยพื้นฐานของเลขยกกำลังมาเป็นองค์ประกอบ

ทฤษฎีที่เกี่ยวข้องกับฟังก์ชัน Exponential

ถ้า a, b เป็นจำนวนจริงบวก โดยที่ $a \neq 1, b \neq 1$ และ x, y เป็นจำนวนจริง, ตัวแปร หรือนิพจน์ทางคณิตศาสตร์

กฎของเลขยกกำลัง (Exponent Laws)

- $a^x a^y = a^{x+y}$
- $(a^x)^y = a^{xy}$
- $(ab)^x = a^x b^x$
- $(a/b)^x = a^x / b^x$
- $a^m / a^n = a^{m-n}$
- $a^{-x} = 1/ax$
- $a^0 = 1$

$a^x = a^y$ *ก็ต่อเมื่อ* $x = y$

ถ้า $x \neq 0$ *แล้ว* $a^x = a^y$ *ก็ต่อเมื่อ* $a = b$

ฟังก์ชั่น Logarithms

ฟังก์ชั่น Logarithm เป็นส่วนกลับกันของ Exponential

โดเมนของ ฟังก์ชั่น Exponential กลายเป็นเรนจ์ (Range) ของฟังก์ชั่น Logarithms ในทางกลับกันเรนจ์ของฟังก์ชั่น Exponential จะกลายเป็นโดเมนของฟังก์ชั่น Logarithm

กำหนดให้ a, x เป็นจำนวนจริง โดยที่ $a > 0, x > 0$ และ $a \neq 1$ แล้ว เป็นฟังก์ชัน Logarithms ก็ต่อเมื่อ

$$y = log_a x \text{ เมื่อ } x = a^y$$

ทฤษฎีที่เกี่ยวข้องกับฟังก์ชัน Logarithms

กำหนดให้ M, N, a และ b เป็นจำนวนจริงบวก
โดยที่ $a \neq 1, b \neq 1$และ n เป็นจำนวนจริง

- $log_a MN = log_a M + log_a N$
- $log_a M/N = log_a M - log_a N$
- $log_a 1 = 0$
- $log_a M^n = nlog_a M$
- $log_a a = 1$
- $log_a M = log_a N$ ก็ต่อเมื่อ $M = N$
- $log_a M = log_b M \,/\, log_b a$
- $a\, log_a M = M$
- $log_{a^n} M = 1/n\, log_a M$
- $\log_{1/a} M = -log_a M$
- $log_a M = 1/\, log_M a$ เมื่อ $M \neq 1$

ฟังก์ชั่น Factorial

ค่า Factorial n (n Factorial) หมายถึง ผลคูณของจำนวนเต็มบวกตั้งแต่ 1 ถึง n เมื่อ n เป็นจำนวนเต็มบวกใดๆ เช่น

$8!$ มีค่า เท่ากับ $8\ x\ 7\ x\ 6\ x\ 5\ x\ 4\ x\ 3\ x\ 2\ x\ 1\ =\ 40320$

หรือ factorial n คือ ผลคูณของจำนวนเต็มบวก n กับจำนวนที่ลดลงจาก n ทีละ 1 จนกระทั่งถึง 1 นั่นเอง โดยสัญลักษณ์ที่กำหนด คือ $n!$

$$n! = n \times (n-1) \times (n-2) \times (n-3) \times \ldots \times 2 \times 1$$

หรือ

$$n! = 1 \times 2 \times 3 \times \ldots \times (n-1) \times n$$

การ Modular

การ Modular จะคล้ายกันกับการหาร (Division) แต่จะแตกต่างจากการหารตรงที่ผลลัพธ์ของการคำนวณ โดยที่ Modular เป็นเศษที่เหลือจากการหาร

กำหนดให้ a เป็นจำนวนเต็ม และ m เป็นจำนวนเต็มบวกแล้ว ค่า $a \ mod \ n$ มีค่าเท่ากับเศษที่เหลือจากการหาร a ด้วย m

เช่น

$7/3 = 2.333$ $7 \div 3 = 2$ $7\%3 = 1$

ลำดับและอนุกรม

ลำดับ (Sequence) คือ ฟังก์ชั่นที่มีโดเมนเป็นเซตของจำนวนเต็มบวกเขียนแทนด้วย n แบ่งเป็น 2 ประเภท คือ

1. ลำดับจำกัด คือ ลำดับที่มีโดเมนเป็นเซตของจำนวนเต็มบวก n ตัวแรก เช่น $1, 2, 3, 4, \ldots, n$
2. ลำดับอนันต์ (Infinite Sequence) คือ ลำดับที่มีโดเมนเป็นเซตของ จำนวนเต็มบวก จนกระทั่งถึงอนันต์ เช่น $1, 2, 3, 4, \ldots$ (ไม่ทราบพจน์สุดท้าย)

อนุกรม (Series) คือ ผลรวมของพจน์ทุกพจน์ของลำดับนั้นๆ ใช้สัญลักษณ์ $\sum$ ตัวการแทนสัญลักษณ์เขียนได้ดังนี้

$$1 + 2 + 3 + 4 + \cdots + n = \sum n$$