2110104: Computer Programming

Tuple & Pair

Dept. of Computer Engineering
Chulalongkorn University

ที่เก็บข้อมูล

- ข้อมูลพื้นฐาน : int, float, char, bool, ...
- กลุ่มข้อมูล
  - เก็บข้อมูลชนิดเดียวกัน
    - แบบมีเลขลำดับ : string, array, vector, ...
    - แบบไม่มีเลขลำดับ : set, map, ...
  - เก็บข้อมูลต่างชนิดกัน
    - จำนวนคงตัว : tuple, pair, ...

ถ้าต้องการเก็บข้อมูลย่อยหลายตัวที่สัมพันธ์กัน

- ถ้าข้อมูลย่อย ๆ เป็นชนิดเดียวกัน -> ใช้ array ได้
- เช่น เก็บ วัน เดือน ปี เกิด ในอาเรย์ 3 ช่อง

```
int main() {
    int bd1[3], bd2[3];
    cin >> bd1[0] >> bd1[1] >> bd1[2];
    cin >> bd2[0] >> bd2[1] >> bd2[2];
    cout << older(bd1, bd2);
}
```

```
bool older(int dmy1[], int dmy2[]) {
    if (dmy1[2] == dmy2[2]) {
        if (dmy1[1] == dmy2[1])
            return dmy1[0] < dmy2[0];
        return dmy1[1] < dmy2[1];
    }
    return dmy1[2] < dmy2[2];
}
```

ถ้าต้องการเก็บข้อมูลย่อยหลายตัวที่สัมพันธ์กัน

- ถ้าข้อมูลย่อย ๆ เป็น/ไม่เป็นชนิดเดียวกัน
- เช่น รหัสสินค้า, ราคา, จำนวนสินค้าในคลัง
  - tuple, pair, struct, class,

ไม่สอนในวิชานี้

ใช้ tuple เก็บข้อมูลย่อย ๆ ที่สัมพันธ์กัน (จำนวนคงตัว)

- เช่น รหัสสินค้า (string), ราคา (float), จำนวนในคลัง (int)

```
#include <tuple>
```

```
tuple<string, float, int> t1 = {"sd0123", 225.50, 250};
tuple<string, float, int> t2("sd0123", 225.50, 250);
tuple<string, float, int> t3 = make_tuple("sd0123",
                                          225.50, 250);

cout << get<2>(t3) << endl;  // 250
get<1>(t3) = 218.0;
cout << get<1>(t3) << endl;
```

ใช้ตัวแปรแทนตำแหน่งไม่ได้

```
int k = 2;
get<k>(t3) = 7; //WRONG
```

การเปรียบเทียบ tuple

```
#include <iostream>
#include <tuple>

using namespace std;

int main() {
  tuple<string, int, int> t1 = make_tuple("AB", 12, 25);
  tuple<string, int, int> t2 = make_tuple("B",   8, 12);
  tuple<string, int, int> t3 = make_tuple("B",   8, 99);

  cout << (t1 < t2) << endl;  // 1
  cout << (t2 < t3) << endl;  // 1
}
```

เปรียบเทียบจากซ้ายไปขวา

ถ้าเท่ากัน ค่อยเปรียบเทียบตัวย่อยถัดไป

หา tuple ที่มีค่ามากสุดใน array

```cpp
#include <iostream>
#include <tuple>
using namespace std;
string to_str(tuple<int,int> & e) {
    return "[" + to_string(get<0>(e)) + ", " +
                 to_string(get<1>(e)) + "]";
}
int main() {
    const int N = 4;
    tuple<int,int> t[N] = {{2,3}, {5,1}, {4,3}, {5,2}};
    for (int i=0; i<N; ++i)
        cout << to_str(t[i]) << endl;
    tuple<int,int> maxt = t[0];
    for (int i=1; i<N; ++i)
        if (t[i] > maxt) maxt = t[i];
    cout << "max = " << to_str(maxt) << endl;
}
```

```
[2, 3]
[5, 1]
[4, 3]
[5, 2]
max = [5, 2]
```

ตัวอย่าง: ใครเป็นพี่

Input	Output
Jane 23 3 2543 Kate 9 6 2544	Jane
Jib 2 7 2545 Mobile 10 7 2545	Jib
Noey 10 12 2532 Jam 10 12 2532	Noey Jam

```cpp
#include <iostream>
#include <tuple>
using namespace std;

int main() {
    string name1, name2;
    int d1, d2, m1, m2, y1, y2;
    cin >> name1 >> d1 >> m1 >> y1;
    cin >> name2 >> d2 >> m2 >> y2;
    tuple<int,int,int> t1 = make_tuple(y1,m1,d1);
    tuple<int,int,int> t2 = make_tuple(y2,m2,d2);
    if (t1 < t2)
        cout << name1 << endl;
    else if (t2 < t1)
        cout << name2 << endl;
    else
        cout << name1 << ' ' << name2 << endl;
}
```

Input	Output
Jane 23 3 2543 Kate 9 6 2544	Jane
Jib 2 7 2545 Mobile 10 7 2545	Jib
Noey 10 12 2532 Jam 10 12 2532	Noey Jam

ถ้าจะใช้ tuple เก็บแค่ 2 ช่อง -> ใช้ pair ดีกว่า

```
#include <utility>
```

```
pair<string, int> p1 = {"sd0123", 256};
pair<string, int> p2("sd0123", 256);
pair<string, int> p3 = make_pair("sd0123", 256);

cout << p2.first << ", " p2.second << endl;
p2.second = 218;
cout << p3.first << ", " p3.second << endl;
```

ใช้ .first .second ในการหยิบสมาชิกของ pair

การเปรียบเทียบ pair

```
#include <iostream>
#include <utility>

using namespace std;

int main() {
  pair<string, int> p1 = make_pair("AB", 2);
  pair<string, int> p2 = make_pair("B",  8);
  pair<string, int> p3 = make_pair("B",  9);

  cout << (p1 < p2) << endl;  // 1
  cout << (p2 < p3) << endl;  // 1
}
```

เปรียบเทียบ .first ก่อน

ถ้าเท่ากัน ค่อยเปรียบเทียบ .second

หา pair ที่มีค่ามากสุดใน array

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

string to_str(pair<int,int> & e) {
    return "[" + to_string(e.first) + ", " +
                 to_string(e.second) + "]";
}
int main() {
    const int N = 4;
    pair<int,int> p[N] = {{2,3}, {5,1}, {4,3}, {5,2}};
    for (int i=0; i<N; ++i)
        cout << to_str(p[i]) << endl;
    pair<int,int> maxp = p[0];
    for (int i=1; i<N; ++i)
        if (p[i] > maxp) maxp = p[i];
    cout << "max = " << to_str(maxp) << endl;
}
```

```
[2, 3]
[5, 1]
[4, 3]
[5, 2]
max = [5, 2]
```

ตัวอย่าง: upgrade

Input	Output
9 9999 B+ 1111 F 2222 A 3333 B 4444 B 5555 D 6666 C 7777 D 8888 A 9999 8888 6666	9999 A 1111 F 2222 A 3333 B 4444 B 5555 D 6666 C+ 7777 D 8888 A

```cpp
#include <iostream>
#include <utility>

using namespace std;

string G[] = {"F", "D", "D+", "C",
              "C+", "B", "B+", "A", "A"};

string upgrade(string g) {
    for (int i=0; i<9; ++i) {
        if (g == G[i]) return G[i+1];
    }
    return g;
}
```

```
int main() {
    int n;
    cin >> n;
    pair<string,string> grades[n];
    string sid, grade;
    for (int i=0; i<n; ++i) {
        cin >> sid >> grade;
        grades[i] = make_pair(sid, grade);
    }
    while (cin >> sid) {
        for (int i=0; i<n; ++i) {
            if (grades[i].first == sid) {
                grades[i].second = upgrade(grades[i].second);
                break;
            }
        }
    }
    for (int i=0; i<n; ++i)
        cout << grades[i].first << ' ' <<grades[i].second<< endl;
}
```

Input

```
9
9999 B+
1111 F
2222 A
3333 B
4444 B
5555 D
6666 C
7777 D
8888 A
9999 8888 6666
```

ใช้ tuple หรือ pair เพื่อการคืนหลายค่าจากฟังก์ชัน

```
pair<float,float> qroots(float a, float b, float c) {
    // return roots of ax^2 + bx + c = 0
    float t = b*b - 4*a*c;
    float r1 = (-b - sqrt(t)) / (2*a);
    float r2 = (-b + sqrt(t)) / (2*a);
    return make_pair(r1, r2);
}

int main() {
    pair<float,float> roots = qroots(1, 3, -10);
    cout << roots.first << ", " << roots.second; // -5, 2
}
```

$x^2 + 3x - 10 = 0$

$(x+5)(x-2) = 0$

tuple / pair : สรุป

- ใช้ tuple กับ pair เมื่อต้องการรวมกลุ่มข้อมูลที่มีความสัมพันธ์กันไว้ด้วยกัน (aggregate data)
- ข้อมูลย่อย ๆ ไม่จำเป็นต้องเป็นชนิดเดียวกัน
- หยิบสมาชิกของ tuple ด้วย get<0>(t), get<1>(t), ...
- หยิบสมาชิกของ pair ด้วย p.first กับ p.second