2110104: Computer Programming

Map

Dept. of Computer Engineering
Chulalongkorn University

การเก็บข้อมูลใน array หรือ vector

ข้อมูลที่เก็บ กับ หมายเลขอินเด็กซ์ มักมีความสัมพันธ์กัน

จะรู้ว่าเลขแต่ละตัวมีกี่ตัว

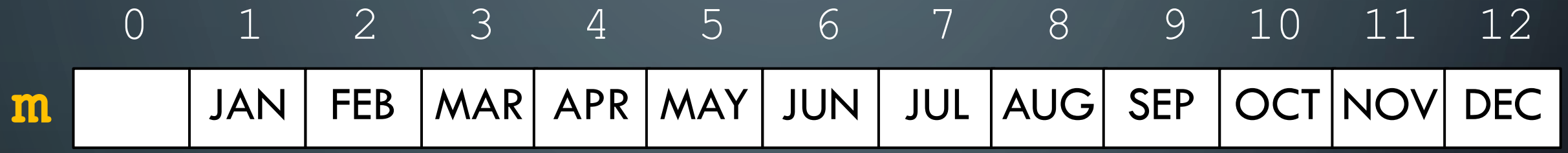
673113318628357 83

ให้เลข `d` ได้ `count[d]` เป็นจำนวนที่ `d` มีในสตริง

int → int

การเก็บข้อมูลใน array หรือ vector

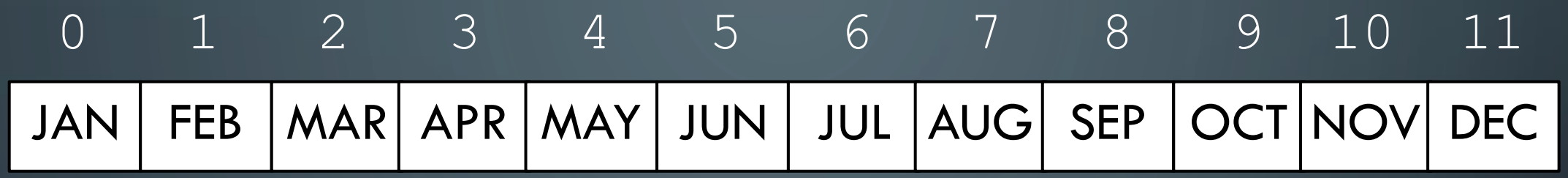
ข้อมูลที่เก็บ กับ หมายเลขอินเด็กซ์ มักมีความสัมพันธ์กัน



ให้เลขเดือน `i` ได้ `m[i]` เป็นชื่อย่อเดือน

int → string

ถ้าต้องการ ชื่อย่อเดือน -> เลขเดือน




```
string M[] = {"JAN", "FEB", "MAR", "APR", "MAY", "JUN",
              "JUL", "AUG", "SEP", "OCT", "NOV", "DEC"};
string mname;
cin >> mname;
for (int n = 0; n<12; ++n)
    if (M[n] == mname)
        cout << n+1 << endl;
        break;
```

มีวิธีดีกว่านี้

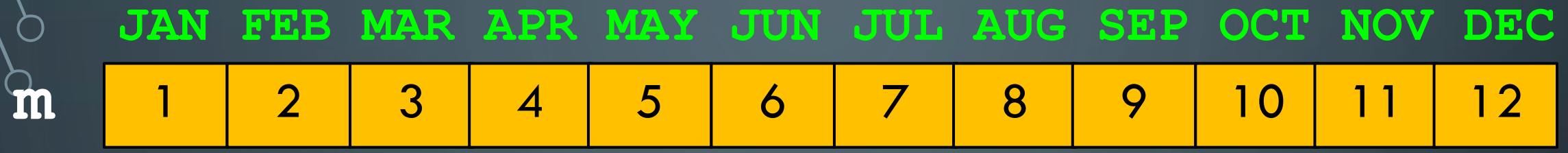
ถ้าต้องการ ชื่อย่อเดือน -> เลขเดือน

string → int

อยากได้


```
m[ name ]
```

ถ้าต้องการ ชื่อย่อเดือน -> เลขเดือน




```
map<string, int> m;
m["JAN"] = 1;
m["FEB"] = 2;
...
m["DEC"] = 12;
string mname;
cin >> mname;
cout << m[ mname ] << endl;
```

string → int

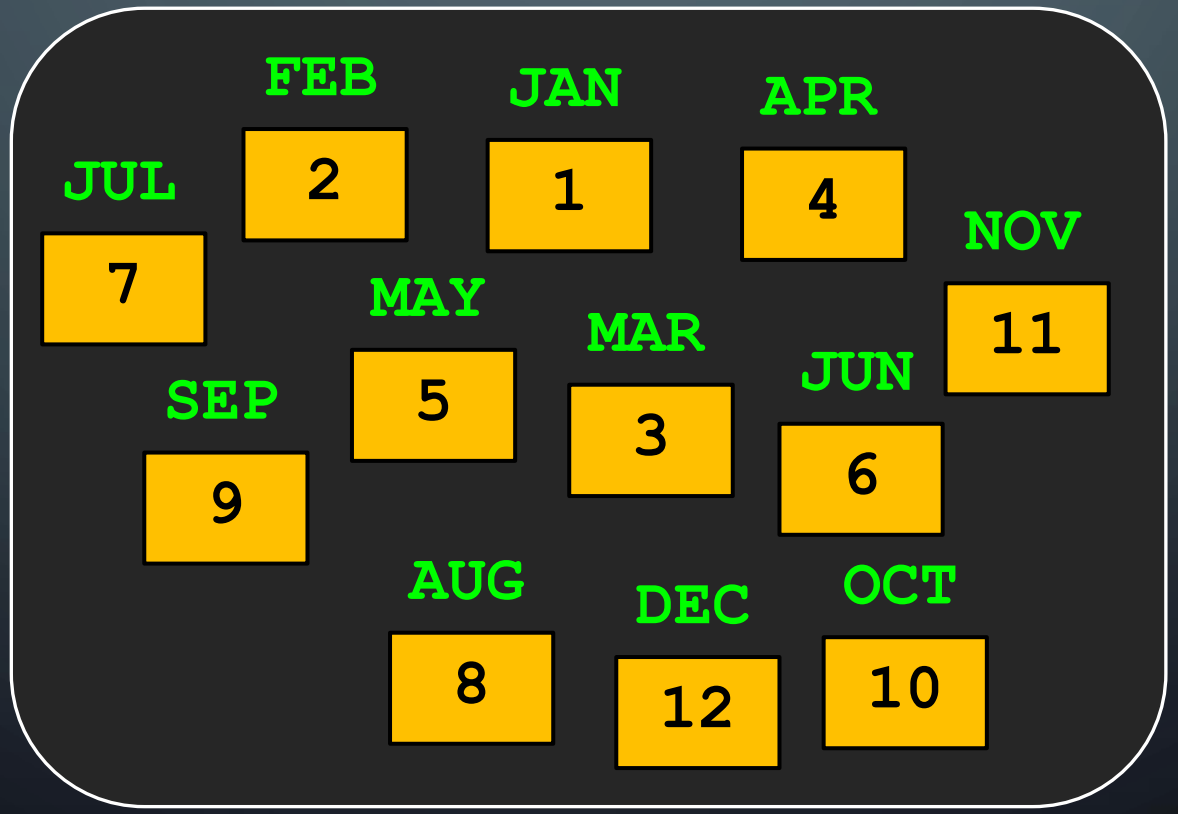
map
dictionary
associative array

ให้ key -> ได้ mapped value

ให้		ได้
ชื่อหุ้น	→	ราคาต่อหน่วย
รหัสนิสิต	→	GPAX
รหัสพนักงาน	→	จำนวนบุตรธิดา
เบอร์โทร	→	ชื่อ
ชื่อ	→	รายการของเบอร์โทร
วันเดือนปี	→	รายชื่อเพื่อนที่เกิดวันที่นี้

map เก็บกลุ่มของคู่ข้อมูล key / mapped value

JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
1	2	3	4	5	6	7	8	9	10	11	12



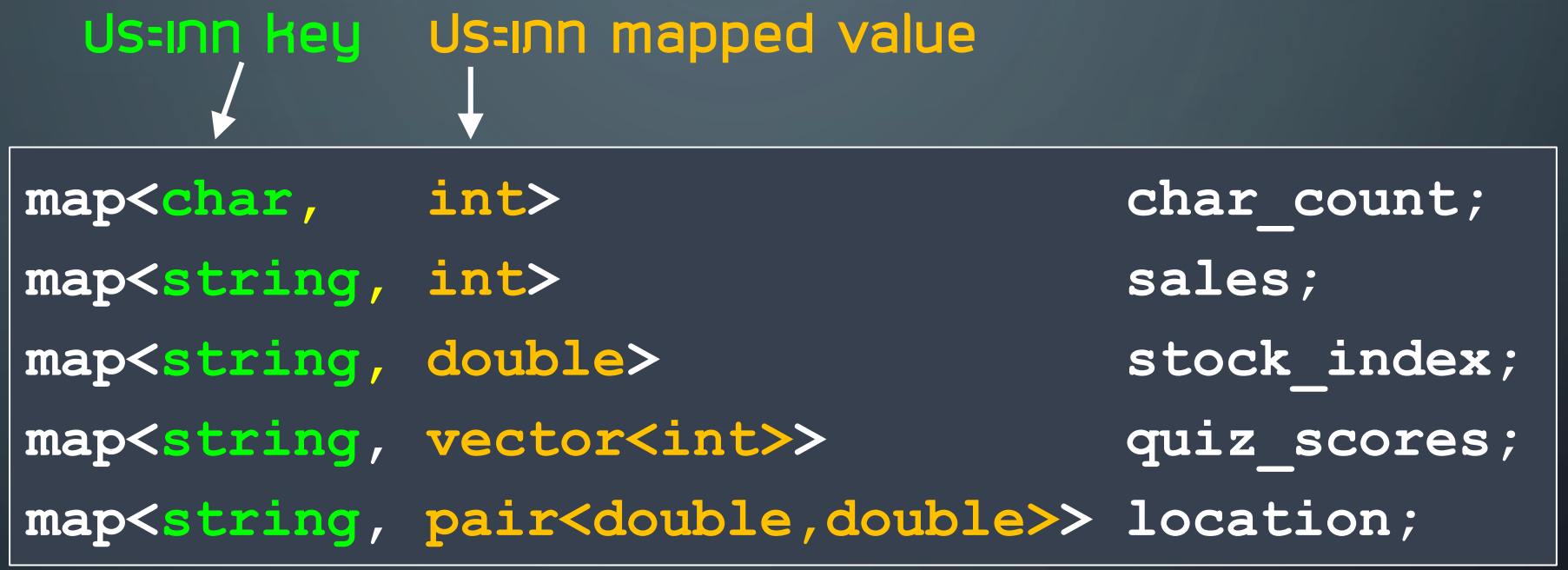
map : กลุ่มของคู่ข้อมูล key / mapped value

- key เป็นประเภทเดียวกัน
- mapped value เป็นประเภทเดียวกัน
- key ต้องมีค่าต่างกันหมด (ไม่ซ้ำกัน)
- เพิ่ม / ลบข้อมูลได้ระหว่างใช้งาน
- เก็บแล้ว เปลี่ยนค่า key ไม่ได้ แต่เปลี่ยน value ได้
- เข้าใช้ข้อมูลผ่าน iterator จะได้เรียงลำดับตาม key

```
#include <map>
```

การสร้าง map

- ต้องกำหนดว่า key กับ mapped value เป็นประเภทใด



แบบนี้ได้ map ว่าง ๆ ยังไม่มีข้อมูล

ตัวอย่าง map

Key	Mapped Value
ชื่อหุ้น	ราคาต่อหน่วย
รหัสนิสิต	GPAX
รหัสพนักงาน	จำนวนบุตรธิดา
เบอร์โทร	ชื่อ
ชื่อ	รายการของเบอร์โทร
วันเดือนปี	รายชื่อเพื่อนที่เกิดวันที่นี้

```
map<string, double>
map<string, double>
map<string, int>
map<string, string>
map<string, vector<string>>
map<tuple<int,int,int>,
    set<string>>
```

การให้ค่าเริ่มต้นกับ map

```
map<string,double> stock_index = {
                        {"SET", 1534.81},
                        {"Hang Seng", 18949.94},
                        {"Nikkei", 31524.22}
                    };
```

key

mapped value

```
// copy จาก stock_index
map<string,double> m1(stock_index);
auto  m2(stock_index);
```

map เก็บคู่ข้อมูลเสมือนเรียงลำดับตาม key

ลำดับของคู่ข้อมูลที่กำหนดตอนสร้าง
หรือเพิ่มในภายหลัง ไม่สำคัญ

```
map<int,int> m1 = { {1,2}, {4,5}, {9,6} };
map<int,int> m2 = { {9,6}, {1,2}, {4,5} };
cout << (m1 == m2) << endl;  // 1

map<int,int> m3 = { {9,0}, {1,2}, {4,5} };
cout << (m2 == m3) << endl;  // 0
```

เพิ่มคู่ข้อมูล : m[key] = mapped_value

```
map<string,double> stock_index;
stock_index["SET"] = 1534.81;
stock_index["Hang Seng"] = 18949.94;
stock_index["Nikkei"] = 31524.22;
stock_index["SET"] = 1530.23; // เปลี่ยน value ได้
```

map ก็มี iterator ให้ใช้

- m.begin(), m.end(), ++itr, itr++, --itr, itr--
- m.begin() คือตำแหน่งของคู่ข้อมูลที่มีค่าคีย์น้อยสุด
- ถ้าเท่ากับ m.end() แสดงว่า ข้อมูลหมดแล้ว
- เข้าใช้ข้อมูลด้วย *itr จะได้เป็น pair
  - (*itr).first คือ key
  - (*itr).second คือ mapped value
- ทำ itr + k และ itr - k ไม่ได้

```
map<int,int> m = { {9,11}, {3,13}, {1,12} };
auto itr = m.begin();
cout << (*(++itr)).second << endl;  // 13
```

ใช้ range-based for loop ได้

```
map<int,int> m = { {9,11}, {1,12}, {3,13} };
for (auto itr = m.begin(),end=m.end(); itr != end; ++itr) {
    auto & kv = *itr;  // ใช้ & จะได้ไม่ต้อง copy pair
    ...
}
```

```
for (auto & kv : s) {  // ใช้ & จะได้ไม่ต้อง copy pair
    ...
}
```

เพิ่ม itr : เลื่อนไปข้อมูลที่มีคีย์มากขึ้น ๆ ๆ

```
map<int,int> m = { {9,11}, {1,12}, {3,13} };
for (auto itr = m.begin(),end=m.end(); itr != end; ++itr) {
    auto & kv = *itr;  // ใช้ & จะได้ไม่ต้อง copy pair
    cout << kv.first << " : " << kv.second << endl;
}
```

```
1 : 12
3 : 13
9 : 11
```

ลด itr : เลื่อนไปข้อมูลที่มีคีย์น้อยลง ๆ ๆ

```
map<int,int> m = { {9,11}, {1,12}, {3,13} };
for (auto itr = m.end(); itr != m.begin();) {
    auto & kv = *(--itr);
    cout << kv.first << " : " << kv.second << endl;
}
```

```
9 : 11
3 : 13
1 : 12
```

การค้นข้อมูล : m.find(k)

คืน iterator ที่ชี้ pair ที่มีคีย์เท่ากับ k
แต่ถ้าไม่มีคีย์ k ใน m จะคืน m.end()

```
map<int,int> m = { {9,11}, {1,12}, {3,13} };
cout << (m.find(1) != m.end()) << endl;  // 1
cout << (m.find(2) != m.end()) << endl;  // 0
```

มีแต่ค้น key
ไม่มีบริการค้น mapped value

อย่าวนหาคีย์เองใน map

```
map<int,string> m;
...
auto i=m.begin();
for (auto e=m.end(); i!=e; ++i)
    if ((*i).first == key) break;
cout << (i!=m.end() ? (*i).second : "?");
```

ช้า

```
map<int,string> m;
...
auto i = m.find(key);
cout << (i!=m.end() ? (*i).second : "?");
```

เร็ว

ตัวอย่าง : tsort(v)

```
9 1 100 5 1 1 2 3 3 100 5 9 1 4 3
```

```
1 : 4
2 : 1
3 : 3
4 : 1
5 : 2
9 : 2
100 : 2
```

```
1 1 1 1 2 3 3 3 4 5 5 9 9 100 100
```

ตัวอย่าง : tsort(v)

```
void tsort(vector<int> & v) {
    map<int,int> m;
    for (auto e : v) {
        auto it = m.find(e);
        if (it == m.end())
            m[e] = 1;
        else
            ++m[e];  // ++((*it).second)
    }
    v.clear();
    for (auto & kv : m)
        while(kv.second--)
            v.push_back(kv.first);
}
```

ลบแบบที่ 1 (ลบตรงไหน) : m.erase(itr)

- ลบคู่ข้อมูลใน m ที่ itr ชี้
- คืน iterator ที่ชี้ pair "ถัดไป"
  ("ถัดไป" ของ e คือข้อมูลที่มีคีย์น้อยสุด
  ที่มากกว่าคีย์ของ e)

```
map<int,int> m = { {9,11}, {1,12}, {3,13} };
auto itr = m.begin();              // ชี้ที่ {1,12}
++itr;                             // ชี้ที่ {3,13}
itr = m.erase(itr);                // ลบ {3,13}, itr ชี้ที่ {9,11}
cout << (*itr).first << endl;      // 9
```

ลบแบบที่ 2 (ลบอะไร) : m.erase(k)

- ลบคู่ข้อมูลที่มีคีย์ k ใน m ออก
- ถ้ามีคีย์ k ให้ลบ คืน 1 ถ้าไม่มี คืน 0

```
map<int,int> m = { {9,11}, {1,12}, {3,13} };
int r;
r = m.erase(3);          // r = 1
r = m.erase(99);         // r = 0
for (auto & kv : m)
    cout << kv.first << " : " << kv.second << endl;
```

1 : 12

9 : 11

คีย์ใน map เก็บแล้ว เป็น read-only

```
map<int,string> m = {{4,"three"}, {5,"five"}};
auto itr = m.begin();
(*itr).second = "four"; // OK
(*itr).first = 3;       // WRONG (read only)
```

```
map<int,string> m = {{4,"three"}, {6,"five"}};
for (auto & p : m) {
    p.first -= 1;  // WRONG (read only)
}
```

บริการอื่น ๆ ของ map

- `m.empty()` ถ้าไม่มีข้อมูลคืนจริง ไม่งั้นคืนเท็จ
- `m.clear()` ลบข้อมูลทั้งหมด เหลือ 0 ตัว
- และอีกหลายบริการ (ที่ขอไม่ครอบคลุมในวิชานี้)

https://cplusplus.com/reference/map/map/

ตัวอย่าง: นับโหวตเฉพาะสมาชิกในวง

```
map<string, int> votes = {{"Jisoo",0},
                          {"Jennie",0},
                          {"Rose",0},
                          {"Lisa",0}};

for (string name; cin >> name;) {
    auto itr = votes.find(name);
    if (itr != votes.end())
        ++((*itr).second);
}
for (auto & kv : votes) {
    cout << kv.first << " --> " << kv.second << endl;
}
```

Input	Output
Lisa Lisa Lisa Jennie Lisa Jennie Aum Jisoo Lisa	Jennie --> 2 Jisoo --> 1 Lisa --> 5 Rose --> 0

ตัวอย่าง: นับโหวตทุกคน

```
map<string, int> votes;
for (string name; cin >> name;) {
    auto itr = votes.find(name);
    if (itr != votes.end())
        votes[name]++; //++((*itr).second);
    else
        votes[name] = 1;
}
for (auto & kv : votes) {
    cout << kv.first << " --> " << kv.second << endl;
}
```

Input	Output
Lisa Lisa Lisa Jennie Lisa Jennie Aum Jisoo Lisa	Aum --> 1 Jennie --> 2 Jisoo --> 1 Lisa --> 5

ตัวอย่าง: ค่าเหยียบแผ่นดิน

Input	Output
6 BKK TH DMK TH CNX TH PAR FR LHR UK NRT JP 4 TH 300 FR 2800 UK 2800 JP 3500 NRT LHR DMK CNX BKK PAR BKK	6200

รหัสสนามบิน รหัสประเทศ

รหัสประเทศ ค่าเหยียบแผ่นดิน

ลำดับสนามบินที่แวะผ่าน

```
 NRT   LHR   DMK   CNX   BKK   PAR   BKK
  JP    UK    TH    TH    TH    FR    TH
   0  2800   300     0     0  2800   300

                                    6200
```

```
int main() {
    map<string, string> country;  // <airport, country>
    string airport, ct;
    int n;
    cin >> n;
    while (n--) {
        cin >> airport >> ct;
        country[airport] = ct;
    }
    map<string, int> fee;         // <country, fee>
    int f;
    cin >> n;
    while (n--) {
        cin >> ct >> f;
        fee[ct] = f;
    }
```

Input

```
6
BKK TH
DMK TH
CNX TH
PAR FR
LHR UK
NRT JP
4
TH 300
FR 2800
UK 2800
JP 3500
```

```
int main() {
    map<string, string> country; // <airport, country>
    ...
    map<string, int> fee;        // <country, fee>
    ...
    int total = 0;
    string last;
    cin >> last;
    while (cin >> airport) {
        if (country[airport] != country[last])
            total += fee[country[airport]];
        last = airport;
    }
    cout << total;
}
```

Input

```
...

NRT LHR DMK CNX BKK PAR BKK
```

ตัวอย่าง: Phone Book

Input	Output
092-923-2929 Jane 089-248-0013 Aom 089-228-9292 Jane 061-029-3939 Ben 062-283-0001 Jane 02-222-4598 Ben	Aom: 089-248-0013 Ben: 061-029-3939 02-222-4598 Jane: 092-923-2929 089-228-9292 062-283-0001

หนึ่งหมายเลขเป็นของหนึ่งคนเท่านั้น
ไม่มีชื่อคนซ้ำ

```
map<string, vector<string>> phonebooks;
string name, phone_no;
while (cin >> phone_no) {
    cin >> name;
    if (phonebooks.find(name) == phonebooks.end())
        phonebooks[name] = vector<string>();
    phonebooks[name].push_back(phone_no);
}
for (auto & kv : phonebooks) {
    cout << kv.first << ": ";
    for (auto & phone_no : kv.second)
        cout << phone_no << ' ';
    cout << endl;
}
```

Input

```
092-923-2929 Jane
089-248-0013 Aom
089-228-9292 Jane
061-029-3939 Ben
062-283-0001 Jane
02-222-4598  Ben
```

map : สรุป

- เหมาะกับการค้นหา mapped value จาก key
- iterator วิ่งตามคีย์จากค่าน้อยไปมาก (หรือมากมาน้อย)
- แต่ห้ามเปลี่ยนค่าของคีย์
- ค้นได้เร็ว เพิ่มได้เร็ว ลบได้เร็ว
  - m.find(k), m[k]=v, m.erase(k), m.erase(itr)
    ใช้เวลาแปรตาม $\log_2$(ปริมาณข้อมูล)
    (จะได้เรียนในวิชาโครงสร้างข้อมูล)

	pair	tuple	vector	set	map
งานที่ใช้	ลำดับข้อมูลย่อย 2 ตัว (ต่างประเภทกันได้)ที่ประกอบกันเป็นกลุ่มข้อมูลที่เกี่ยวข้องกัน	ลำดับข้อมูลย่อยหลายตัว (ต่างประเภทกันได้) ที่ประกอบกันเป็นกลุ่มข้อมูลที่เกี่ยวข้องกัน	รายการของข้อมูลประเภทเดียวกัน ซ้ำได้ ข้อมูลแต่ละตัวมีเลขอินเด็กซ์กำกับ	กลุ่มข้อมูลที่มีค่าต่าง ๆ กัน (ไม่มีค่าซ้ำ) ข้อมูลไม่มีเลขอินเด็กซ์	กลุ่มของคู่ข้อมูล key และ mapped value ที่ไม่มีค่า key ซ้ำกัน
การเข้าใช้ข้อมูล	`p.first` `p.second`	`get<0>(t)` `get<1>(t) ...`	`v[0], v[1], ...` `v[k]`หรือผ่าน iterator	ผ่าน iterator	`m[key]` ได้ mapped val หรือผ่าน iterator
iterator	ไม่มี	ไม่มี	มี `++, --, +k, -k` ได้เรียงตามลำดับจากซ้ายไปขวา	มี `++` กับ `--` ได้เรียงลำดับตามค่าของข้อมูล	มี `++` กับ `--` ได้เรียงลำดับตามค่าของคีย์
range-based for loop	ไม่มี	ไม่มี	ได้เรียงตามลำดับจากซ้ายไปขวา	ได้เรียงตามลำดับจากค่าน้อยไปมาก	ได้เรียงตามลำดับค่าของคีย์จากน้อยไปมาก
การค้น	ไม่มี	ไม่มี	`find(itr1,itr2,e)` เวลาแปรตามปริมาณข้อมูล	`s.find(e)` ทำงานเร็ว	`m.find(e)` ทำงานเร็ว
การสร้าง	`make_pair(f,s)`	`make_tuple(...)`	initializer list `{3,1,2,2}`	initializer list `{3,1,2}`	initializer list `{{3,1},{2,9}}`
การเพิ่ม	ไม่มี	ไม่มี	`v.push_back(e)` `v.insert(itr,e)`	`s.insert(e)`	`m[k] = mapped_val`
การลบ	ไม่มี	ไม่มี	`v.pop_back()` `v.erase(itr)`	`s.erase(itr)` `s.erase(e)`	`m.erase(itr)` `m.erase(key)`