

# C++ STL

vector, list, string

@arbuztw

# Standard Template Library

- 常用到的資料結構都要自己寫嗎?
  - linked list
  - map
  - ...

# Standard Template Library

- 常用到的資料結構都要自己寫嗎?
  - linked list
  - map
  - ...
- 常用到的演算法也要自己寫嗎?
  - 二分搜尋
  - 排序
  - 全排列

# Standard Template Library

- 1990年代成為C++ Standard的一部分
- 包含演算法(algorithm)、容器(container)、迭代器(iterator)...
- 用C++ Template(樣版)來實作

# 泛型程式設計 (Generic Programming)

- 問題: 給一個int陣列，請找出最小值。
  - 實做 `int findMax(int arr[], int size)`

# 泛型程式設計 (Generic Programming)

- 問題: 給一個int陣列，請找出最小值。
  - 實做 `int findMax(int arr[], int size)`
- 給一個double陣列，請找出最小值。
- 給一個char陣列，請找出最小值。
- 給一個...陣列，請找出最小值。

# 泛型程式設計 (Generic Programming)

- 問題: 給一個int陣列，請找出最小值。
  - 實做 `int findMax(int arr[], int size)`
- 給一個double陣列，請找出最小值。
- 給一個char陣列，請找出最小值。
- 給一個...陣列，請找出最小值。
- 要寫 $N$ 次一樣的函數?

# Template

- 如果能這樣該有多好

```
T findMax(T arr[], int size) {  
    T mx = arr[0];  
    for (int i = 1; i < size; i++)  
        if (arr[i] > mx)  
            mx = arr[i];  
    return mx;  
}
```

- 令 `T = int, double, char ...`



# Template

- 如果能這樣該有多好

```
T findMax(T arr[], int size) {  
    T mx = arr[0];  
    for (int i = 1; i < size; i++)  
        if (arr[i] > mx)  
            mx = arr[i];  
    return mx;  
}
```

- 令 `T = int, double, char ...`
- C++ Template!
- 「可重用性(reusibility)」的表現

# vector

- 動態伸縮的陣列，可以快速存取任意元素。
- 常用函數

```
#include <vector>

std::vector<T> vec;           // T為一個type (e.g. int/char...)
std::vector<T> vec(n);        // 長度為n的vector
std::vector<T> vec(n, x);     // 長度為n且初始化為x的vector
vec[i];                       // i-th element
vec.push_back(element);       // 新增element到vector尾端
vec.pop_back();               // 移除
vec.size();                   // 回傳vector長度
vec.clear();                   // 清空vector
```

- Reference:  
<http://www.cplusplus.com/reference/vector/vector/>

# vector範例

```
#include <iostream>
#include <vector>

int main() {
    std::vector<int> vec;

    vec.push_back(3);
    vec.push_back(2);
    vec.pop_back();
    vec.push_back(1);

    std::cout << "size = " << vec.size();
    std::cout << ", element: " << vec[0] << ", " << vec[1] << std::endl;

    return 0;
}
```

```
size = 2, element: 3, 1
```

# list

- double linked-list，可以快速插入/刪除元素。
- 常用函數

```
#include <list>

std::list<T> xs;
xs.push_front(element);
xs.push_back(element);
xs.pop_front();
xs.pop_back();
xs.insert(iterator, element); //在iterator前插入element
xs.erase(iterator);          //刪除iterator指向的元素
```

- Reference:  
<http://www.cplusplus.com/reference/list/list/>

# iterator (迭代器)

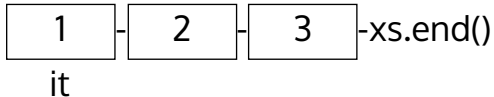
- 類似指標，指向容器(container)中的元素
- 無需關心容器底層的實作
- Example

```
std::list<int> xs;  
std::list<int>::iterator it;  
  
for (it = xs.begin(); it != xs.end(); it++)  
    std::cout << *it << std::endl;
```

- (Invalidation)在erase或vector push\_back後，記憶體位置可能會改變，因此不保證原先的iterator仍會指到對的位置
- 更多invalidation: **Iterator invalidation rules**

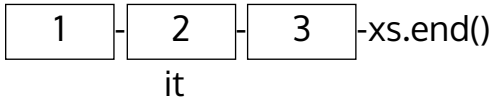
# list範例

```
std::list<int> xs;  
std::list<int>::iterator it;  
  
for (int i = 1; i <= 3; i++)  
    xs.push_back(i);  
  
it = xs.begin();  
/*it++;  
xs.insert(it, 6);  
xs.insert(it, 7);  
it--;  
xs.erase(it);*/
```



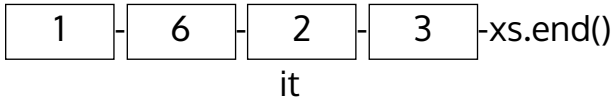
# list範例

```
std::list<int> xs;  
std::list<int>::iterator it;  
  
for (int i = 1; i <= 3; i++)  
    xs.push_back(i);  
  
it = xs.begin();  
it++;  
/*xs.insert(it, 6);  
xs.insert(it, 7);  
it--;  
xs.erase(it);*/
```



# list範例

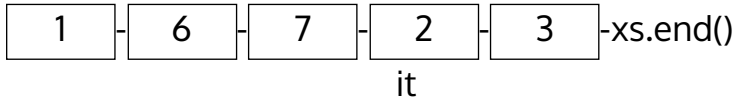
```
std::list<int> xs;  
std::list<int>::iterator it;  
  
for (int i = 1; i <= 3; i++)  
    xs.push_back(i);  
  
it = xs.begin();  
it++;  
xs.insert(it, 6);  
/*xs.insert(it, 7);  
it--;  
xs.erase(it);*/
```





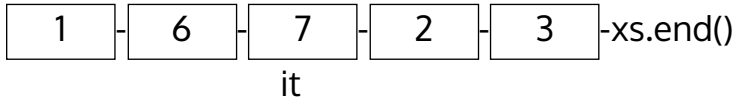
# list範例

```
std::list<int> xs;  
std::list<int>::iterator it;  
  
for (int i = 1; i <= 3; i++)  
    xs.push_back(i);  
  
it = xs.begin();  
it++;  
xs.insert(it, 6);  
xs.insert(it, 7);  
/*it--;  
xs.erase(it);*/
```



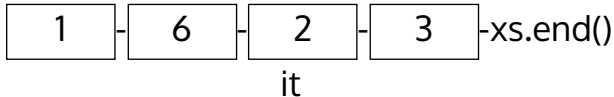
# list範例

```
std::list<int> xs;  
std::list<int>::iterator it;  
  
for (int i = 1; i <= 3; i++)  
    xs.push_back(i);  
  
it = xs.begin();  
it++;  
xs.insert(it, 6);  
xs.insert(it, 7);  
it--;  
/*xs.erase(it);*/
```



# list範例

```
std::list<int> xs;  
std::list<int>::iterator it;  
  
for (int i = 1; i <= 3; i++)  
    xs.push_back(i);  
  
it = xs.begin();  
it++;  
xs.insert(it, 6);  
xs.insert(it, 7);  
it--;  
it = xs.erase(it);
```



# string

- 字串物件
- 常用函數

```
#include <string>

std::string str1, str2;
str1[i];
str1 + str2;
str1 == str2;
str1.length(); str1.size();
str1.substr(0, 3);
str1.replace(0, 3, str2);
str1.reverse();
str1.find(str2);
str1.c_str();
```

- Reference:  
<http://www.cplusplus.com/reference/string/string/>

# string範例

```
#include <iostream>
#include <string>

using std::string;

int main() {
    string str1, str2;
    size_t k;

    str1 = string("hello");
    str1 += string(" world");
    str2 = string("ello");

    k = str1.find(str2);

    std::cout << str2 << " is at index " << k << " of " << str1;

    return 0;
}
```

```
ello is at index 1 of hello world
```