**STL 기초 – 바인더**

IsMulti 예제

#include <iostream>

#include <vector>

#include <algorithm>

#include <functional>

using namespace std;

struct IsMulti : public binary\_function<int,int, bool> {

bool operator()(int a,int b) const {

return (a % b == 0);

}

};

void main()

{

IsMulti IM;

if (IM(6,3)) { cout << "6은 3의 배수이다." << endl; }

if (IM(9,2)) { cout << "9는 2의 배수이다." << endl; }

}

* 두 개의 인수를 전달받음으로써 일반성을 확보한 것은 좋은데 이렇게 되면 단항 조건자를 요구하는 find\_if와는 함께 사용할 수 없다

bind1st(이항 객체, 고정값)

bind2nd(이항 객체, 고정값)

* bind1st는 첫 번째 인수를 고정하며 bind2nd는 두 번째 인수를 고정한다
* bind1st는 잘 사용하지 않음

bind2nd 예제

#include <iostream>

#include <vector>

#include <algorithm>

#include <functional>

using namespace std;

struct IsMulti : public binary\_function<int,int, bool> {

bool operator()(int a,int b) const {

return (a % b == 0);

}

};

void main()

{

int ari[]={1,2,3,4,5,6,7,8,9,10};

vector<int> vi(&ari[0],&ari[10]);

vector<int>::iterator it;

for (it=vi.begin();;it++) {

it=find\_if(it, vi.end(), bind2nd(IsMulti(),3));

if (it==vi.end()) break;

cout<< \*it << "이(가) 있다" << endl;

}

}

it=find\_if(it, vi.end(), bind2nd(greater<int>(),5));

it=find\_if(it, vi.end(), bind2nd(less\_equal<int>(),5));

* greater는 두 값을 비교하여 앞의 값이 뒤의 값보다 더 큰지를 조사하는 이항 조건자