#include <iostream>

class DisjointSets {

public:

int s[256];

DisjointSets() { for (int i = 0; i < 256; i++) s[i] = -1; }

int find(int i);

};

/\* Modify the find() method below to

\* implement path compression so that

\* element i and all of its ancestors

\* in the up-tree refer directly to the

\* root after that initial call to find()

\* returns.

\*/

/\*int DisjointSets::find(int i) {

if ( s[i] < 0 ) {

return i;

} else {

s[i] = s[s[i]];

return find(s[i]);

}

}\*/// this does not work

**int** DisjointSets**::**find(**int** i) {

**if** ( s[i] **<** 0 ) {

**return** i;

} **else** {

**int** root **=** find(s[i]);

s[i] **=** root;

**return** root;

}

}

int main() {

DisjointSets d;

d.s[1] = 3;

d.s[3] = 5;

d.s[5] = 7;

d.s[7] = -1;

std::cout << "d.find(3) = " << d.find(3) << std::endl;

std::cout << "d.s(1) = " << d.s[1] << std::endl;

std::cout << "d.s(3) = " << d.s[3] << std::endl;

std::cout << "d.s(5) = " << d.s[5] << std::endl;

std::cout << "d.s(7) = " << d.s[7] << std::endl;

return 0;

}