Initializer lists

When we want to initialize a vector, we had to do the following:

// C++03

std::vector v;

v.push\_back(1);

v.push\_back(2);

v.push\_back(3);

v.push\_back(4);

Now, with C++ 11, we can do this:

// C++11

std::vector v = { 1, 2, 3, 4 };

C++11 binds the concept to a template, called std::initializer\_list. This allows constructors and other functions to take initializer-lists as parameters:

#include <iostream>

#include <vector>

class MyNumber

{

public:

MyNumber(const std::initializer\_list<int> &v;) {

for (auto itm : v) {

mVec.push\_back(itm);

}

}

void print() {

for (auto itm : mVec) {

std::cout << itm << " ";

}

}

private:

std::vector<int> mVec;

};

int main()

{

MyNumber m = { 1, 2, 3, 4 };

m.print(); // 1 2 3 4

return 0;

}