//C++ For C Programmers Part B week 3

//Ira Pohl

//illustrate some newer ideas from C++11

//such as move semantics

//implement our own form of <array>

#include <iostream>

using namespace std;

template <class T, int n>

class my\_container {

public:

my\_container(){ a = new T[n];}

explicit my\_container(T \*b):my\_container()

{

for(int i = 0; i <n ; ++i)

a[i] = b[i];

}

my\_container(const my\_container &b):my\_container()

{

for(int i = 0; i <n ; ++i)

a[i] = b.a[i];

}

my\_container(my\_container &&b)noexcept

{

a = b.a;

b.a = nullptr;

}

my\_container& operator=(const my\_container& b){

for(int i = 0; i <n ; ++i)

a[i] = b.a[i];

return \*this;

}

my\_container& operator=(my\_container&& b)noexcept

{

a = b.a;

b.a = nullptr;

return \*this;

}

void fill(T x)

{

for(int i = 0; i <n ; ++i)

a[i] = x;

}

void print()

{

if (a == nullptr){ cout << "empty" << endl; return;}

for(int i = 0; i <n ; ++i)

cout << a[i] << ",";

cout << endl;

}

void swap(my\_container &b)

{

my\_container temp = move(b);

b = move(\*this);

\*this = move(temp);

}

T& operator[](int index) // [],=,(), -> must be members

{ return(a[index]);}

private:

T\* a;

};

int main ()

{

double data [10]{1,2,3,4,5,6,7,8,9,10};

my\_container<double,5> x;

my\_container<double,5> y(data);

x.fill(2.0);

x.print();

y.print();

y.swap(x); //move based swap

x.print();

y.print();

x = move(y);

x.print();

y.print();

cout<< " one element in x is " << x[3] << endl;

}