#include<deque>

#include<thread>

#include<iostream>

#include<mutex>

#include<chrono>

using namespace std;

deque<int> q;

mutex mu;

condition\_variable cond;

void function\_1()

{

int count = 10;

while (count > 0)

{

unique\_lock<mutex> locker(mu);

q.push\_front(count);

cout << "thread1 is: " << this\_thread::get\_id() << endl;

cout << "t1 pushed:" << count << endl;

locker.unlock();

cond.notify\_one();

this\_thread::sleep\_for(chrono::seconds(1));

count--;

}

}

void function\_2()

{

int data = 0;

while (data != 1) {

unique\_lock<mutex> locker(mu);

//if (!q.empty()) {

cond.wait(locker, []() {return !q.empty(); });

data = q.back();

q.pop\_back();

locker.unlock();

cout << "thread2 is: " << this\_thread::get\_id() << endl;

cout << "t2 got an value from t1:" << data << endl;

//}

//else {

// locker.unlock();

// this\_thread::sleep\_for(chrono::milliseconds(50));

//}

}

}

int main()

{

thread t1(function\_1);

thread t2(function\_2);

t1.join();

t2.join();

return 0;

}