6.18 给　n 台电脑分配三台打印机，每台电脑拥有不同的优先级

class Monitor {

private:

std::mutex mtx;

std::condition\_variable cv;

constexpr static int initSize = 3;

std::list<int> printer = { 1, 2, 3 };

public:

int pop() {

std::unique\_lock<std::mutex> lck(mtx);

this->cv.wait(lck, [this]{

return this->printer.size();

});

auto one = printer.front();

printer.pop\_front();

return one;

}

void push(const int NO) {

std::unique\_lock<std::mutex> lck(mtx);

this->printer.emplace\_back(NO);

if(!this->printer.empty())

this->cv.notify\_all();

}

} monitor;

int main() {

srand(unsigned(time(nullptr)));

struct pair {

mutable int NO = -1;

int priority;

mutable std::thread process;

pair(const int p) : priority(p)

{}

bool operator<(const pair& rhs) const {

return this->priority < rhs.priority;

}

};

std::priority\_queue< pair, std::vector<pair> > Q;

for(int i = 0;i < 30; ++i) {

std::cout << "第　" << i + 1 << "　台电脑发出请求\n";

Q.emplace(rand() % 30);

}

while(!Q.empty()) {

Q.top().process = std::thread([&]{

if(Q.top().NO == -1)

Q.top().NO = monitor.pop();

std::cout << "优先权为　" << Q.top().priority << " 的计算机获得第　" << Q.top().NO << " 号打印机\n";

std::this\_thread::sleep\_for(std::chrono::seconds(rand() % 3 + 2));

});

if(Q.top().process.joinable())

Q.top().process.join();

monitor.push(Q.top().NO);

Q.pop();

}

return 0;

}