OS 學期計畫第一組

103703003 資科三 張芷銓

103703019 資科三 蔡漢龍

103703020 資科三 李其祐

計畫問題: 2. Tobacco Smokers' (TS) Problem

一.問題內容

Tobacco Smokers' (TS) Problem

Three smokers sit around a table. Each has a permanent supply of precisely one of three resources, namely tobacco, cigarette papers, and matches, but is not permitted to give any of the three resources. The smoker who has the permanent supply of the remaining resource is then in a position to make and smoke a cigarette. On finishing the cigarette this smoker signals the agent, and the agent may then make again available a supply of some two resources.

The smokers are three threads, and the agent can be regarded as a set of three threads. As regards the latter, either none or exactly two of them run at 'anyone time. The problem is to have the six threads cooperate in such a way that deadlock is prevented, e.g., that when the agent supplies paper and matches, it is indeed the smoker with the supply of tobacco who gets both, instead of one or both of these resources being acquired by the other two smokers.

二.解決方式

三條 thread 代表三個抽菸者；三條 thread 代表三種 agent 發放材料的情況
(tobacco、match；tobacco、paper；match、paper)。

0. 一開始三條 smoker threads 會被 lock 住。
1. 利用亂數控制 agnet thread，三條 agent threads 只會有一條發放材料，另外兩條不做任何事情。
2. 發放材料的 agent thread 執行完後會 unlock 三條 smoker threads，並被 lock 住。
3. smoker thread 確認場上兩種材料都是自己所需的時候，才能拿取；反之則什麼都不做，並被 lock 住。
4. 拿到材料的 smoker thread 抽菸，結束後 unlock agent thread，並被 lock 住。
5. 回到第一步驟繼續執行。

三.各項 Thread 功能與程式碼介紹

```
1  #include "mainwindow.h"
2  #include <QString>
3  #include <iostream>
4  #include <QApplication>
5  #include <thread>
6  #include <cstdlib>
7  #include <mutex>
8  #include <chrono>
9  #include <cstdlib>
10 #include <QPushButton>
11 #include <QObject>
12 #include <random>
13 using namespace std;
14 MainWindow *w;
15 int agent=-1;
16
17 //light_there
18 int match=0;//light_match
19 int tobacco=0;//light_tobacco
20 int paper=0;//light_paper
21 int light_s1=0;//S_match
22 int light_s2=0;//S_tobacco
23 int light_s3=0;//S_paper
24 int s1=0,s2=0,s3=0;
25 int Text=0,stop=0;
26
27 mutex agent_Mutex;
28 mutex agent_Mutex1;
29 mutex agent_Mutex2;
30 mutex agent_Mutex3;
31 mutex smoker_Mutex1;
32 mutex smoker_Mutex2;
33 mutex smoker_Mutex3;
34
```

UI 控制變數

Thread 共同變數

UI 控制變數

Mutex lock

Random 隨機從三個原料中選兩個材料，共有三種可能，將隨機選中組合對應的 agent_thread 解鎖。

```
35 void rAgent(){
36     while(1){
37         agent_Mutex.lock();
38         agent = rand()%3;
39         if(agent==0){
40             agent_Mutex1.unlock();//match,tobacco
41         }
42         else if(agent==1){
43             agent_Mutex2.unlock();//match,paper
44         }
45         else if(agent==2){
46             agent_Mutex3.unlock();//tobacoo,paper
47         }
48     }
49 }
```

Agent_thread:

提供 match & tobacco 兩種原料的 Agent，提供之後，通知三個 smokers 來領原料。

```
51 void agent_1(){
52     while(1){
53         agent_Mutex1.lock();
54         if(agent==0 && match+tobacco+paper==0){
55             match=1;
56             tobacco=1;
57             cout << endl;
58             cout << "|||||||||||||||||\n";
59             cout << "| match tobacco |\n";
60             cout << "|||||||||||||||||\n";
61            // Sleep(2000);
62             this_thread::sleep_for(chrono::seconds(2));
63             smoker_Mutex1.unlock();
64             smoker_Mutex2.unlock();
65             smoker_Mutex3.unlock();
66         }
67     }
68 }
```

Agent_thread:

提供 match & paper 兩種原料的 Agent，提供之後，

通知三個 smokers 來領原料。

```
void agent_2(){
    while(1){
        agent_Mutex2.lock();
        if(agent==1 && match+tobacco+paper==0){
            match=1;
            paper=1;
            cout << endl;
            cout << "||||||||||||||||||\n";
            cout << "| match   paper |\n";
            cout << "||||||||||||||||||\n";
          //  Sleep(2000);
            this_thread::sleep_for(chrono::seconds(2));
            smoker_Mutex1.unlock();
            smoker_Mutex2.unlock();
            smoker_Mutex3.unlock();
        }
    }
}
```

Agent_thread:

提供 tobacco & paper 兩種原料的 Agent，提供之後，

通知三個 smokers 來領原料。

```
void agent_3(){
    while(1){
        agent_Mutex3.lock();
        if(agent==2 && match+tobacco+paper==0){
            tobacco=1;
            paper=1;
            cout << endl;
            cout << "||||||||||||||||||\n";
            cout << "| tobacco paper |\n";
            cout << "||||||||||||||||||\n";
           // Sleep(2000);
            this_thread::sleep_for(chrono::seconds(2));
            smoker_Mutex1.unlock();
            smoker_Mutex2.unlock();
            smoker_Mutex3.unlock();
        }
    }
}
```

Smoker_thread:

手上握有 tobacco 的 smoker，只有當 Agent 提供另外兩種原料

match、paper 時，才可以抽煙，抽完菸後，通知 Agent 發原料。

```
void smoke_tobacco(){
    while(1){
        smoker_Mutex2.lock();
        unsigned seed = chrono::system_clock::now().time_since_epoch().count();
        default_random_engine generator(seed);
        poisson_distribution<int> distribution(2.5);

        int number = distribution(generator);
        while(number>5||number<0){
            number = distribution(generator);
        }
        this_thread::sleep_for(chrono::seconds(number));
        if(match==1 && paper==1){
            s2=-1;
            light_s2=1;

            cout << "Hey, I'm tobacco.\n";
            Text=2;
          //   Sleep(2000);
           this_thread::sleep_for(chrono::seconds(2));
            match=0;
            paper=0;
            cout << "goto smoke!\n";
            Text=4;
           // Sleep(2000);
           this_thread::sleep_for(chrono::seconds(5));
            cout << "Agent wake up!\n";
            Text=5;
            light_s2=0;
            s2=0;
            agent_Mutex.unlock();
        }
        else{
             light_s2=-1;
            s2=1;
            this_thread::sleep_for(chrono::seconds(1));
            s2=0;
             light_s2=0;
        }
    }
}
```

Smoker_thread:

手上握有 match 的 smoker，只有當 Agent 提供另外兩種原料

tobacco、paper 時，才可以抽煙，抽完菸後，通知 Agent 發原料。

```
106 void smoke_match(){
107     while(1){
108         smoker_Mutex1.lock();
109         unsigned seed = chrono::system_clock::now().time_since_epoch().count();
110         default_random_engine generator(seed);
111         poisson_distribution<int> distribution(2.5);
112         int number = distribution(generator);
113         while(number>5||number<0){
114             number = distribution(generator);
115         }
116         this_thread::sleep_for(chrono::seconds(number));
117         if(tobacco==1 && paper==1){
118            s1=-1;
119             light_s1=1;
120 
121             cout << "Hey, I'm match.\n";
122             Text=1;
123           //   Sleep(2000);
124            this_thread::sleep_for(chrono::seconds(2));
125             tobacco=0;
126             paper=0;
127             cout << "goto smoke!\n";
128             Text=4;
129           //  Sleep(2000);
130            this_thread::sleep_for(chrono::seconds(5));
131             cout << "Agent wake up!\n";
132             Text=5;
133             light_s1=0;
134             s1=0;
135             agent_Mutex.unlock();
136         }
137         else{
138             light_s1=-1;
139             s1=1;
140             this_thread::sleep_for(chrono::seconds(1));
141             s1=0;
142              light_s1=0;
143         }
144     }
145 }
```

Smoker_thread:

手上握有 paper 的 smoker，只有當 Agent 提供另外兩種原料

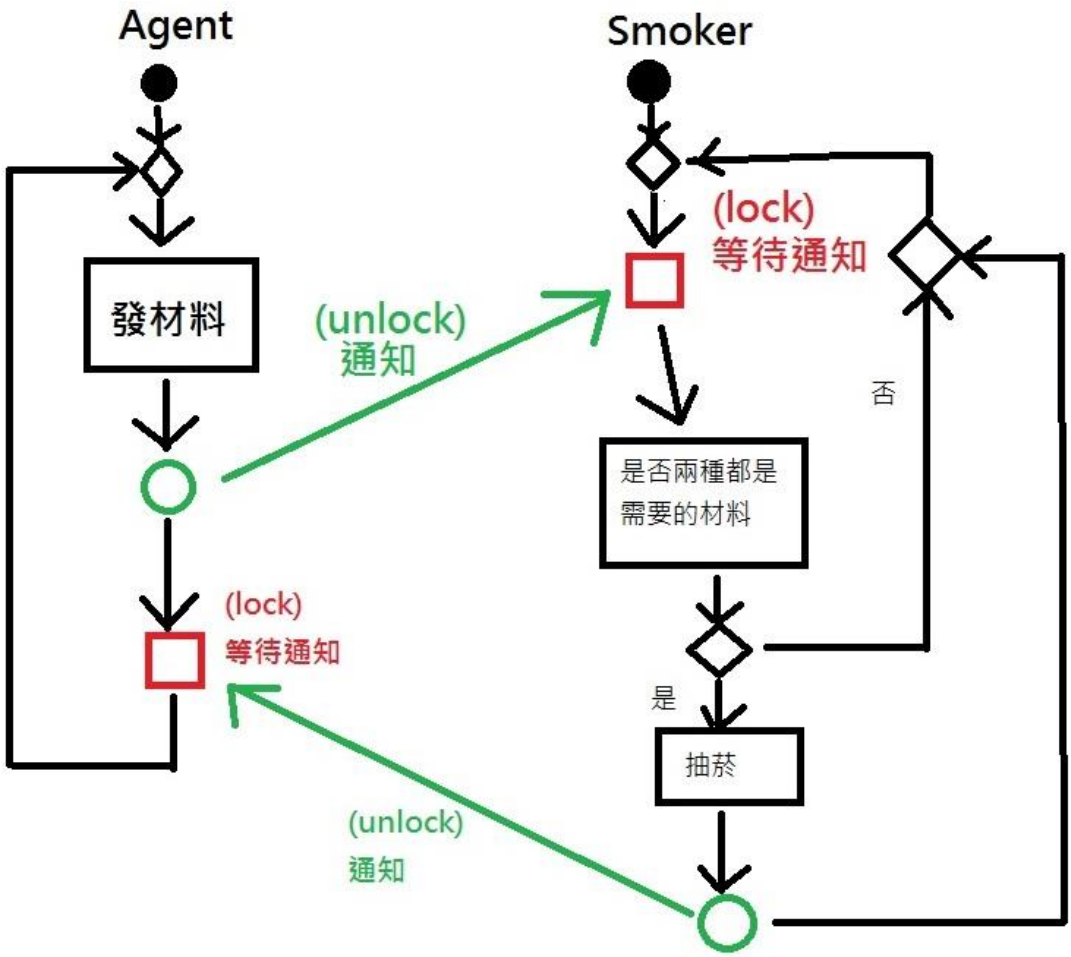
tobacco、match 時，才可以抽煙，抽完菸後，通知 Agent 發原料。

```
void smoke_paper(){
    while(1){
        smoker_Mutex3.lock();
        unsigned seed = chrono::system_clock::now().time_since_epoch().count();
        default_random_engine generator(seed);
        poisson_distribution<int> distribution(2.5);

        int number = distribution(generator);
        while(number>5||number<0){
            number = distribution(generator);
        }
        this_thread::sleep_for(chrono::seconds(number));
        if(match==1 && tobacco==1){
            s3=-1;
            light_s3=1;

            cout << "Hey, I'm paper.\n";
            Text=3;
            // Sleep(2000);

            this_thread::sleep_for(chrono::seconds(2));
            match=0;
            tobacco=0;
            cout << "goto smoke!\n";
            Text=4;
            //Sleep(2000);
            this_thread::sleep_for(chrono::seconds(5));
            cout << "Agent wake up!\n";
            Text=5;
            light_s3=0;
            s3=0;
            agent_Mutex.unlock();
        }
        else{
             light_s3=-1;
            s3=1;
            this_thread::sleep_for(chrono::seconds(1));
            s3=0;
             light_s3=0;
        }
    }
}
```

Main 函示

```
236 int main( int argc, char** argv){
237     QApplication a(argc, argv);
238     w=new MainWindow();
239 
240     w->show();
241 
242     cout << "Check\n";
243     smoker_Mutex1.lock();
244     smoker_Mutex2.lock();
245     smoker_Mutex3.lock();
246     agent_Mutex1.lock();
247     agent_Mutex2.lock();
248     agent_Mutex3.lock();
249     thread mThreadA(rAgent);
250     thread mThreadA_1(agent_1);
251     thread mThreadA_2(agent_2);
252     thread mThreadA_3(agent_3);
253     thread mThreadM(smoke_match);
254     thread mThreadT(smoke_tobacco);
255     thread mThreadP(smoke_paper);
```

流程圖



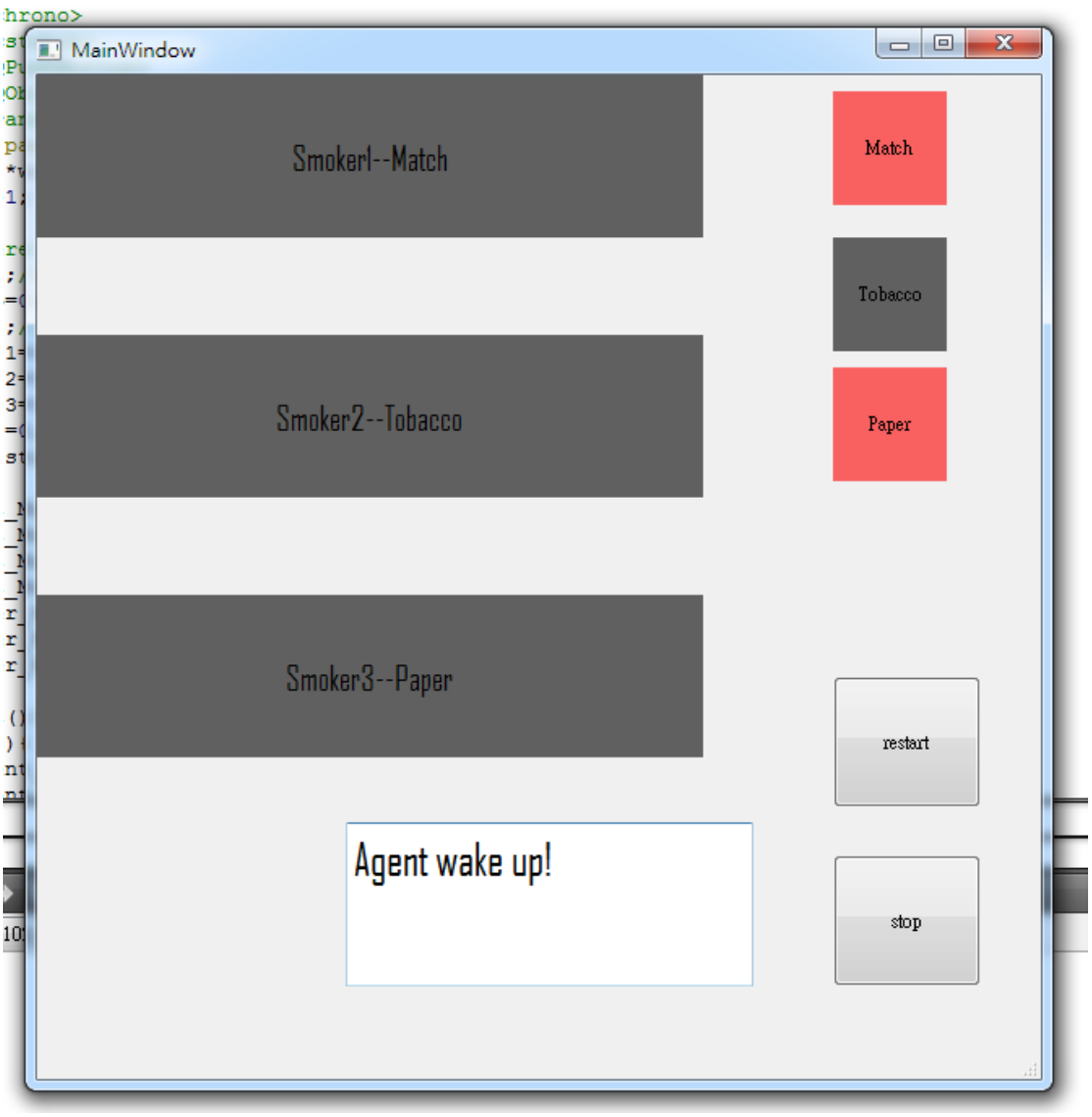
poisson time 的使用

放在 smoker thread 前面，以 poisson distribution 亂數取得一個等待時間，模擬 agent 通知 smoker 後，三個 smoker 隨機到來的情況。

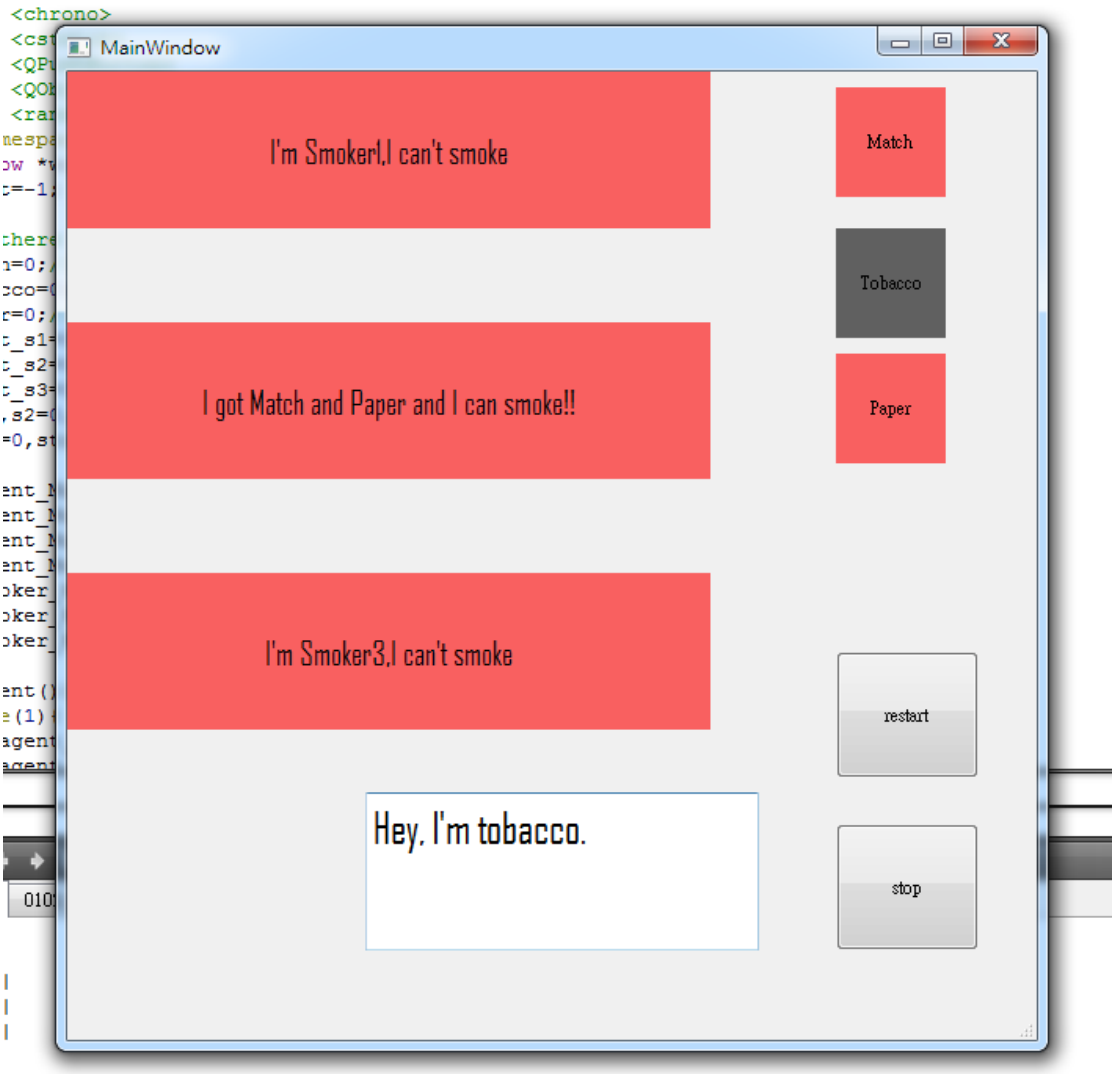
```
unsigned seed = chrono::system_clock::now().time_since_epoch().count();
default_random_engine generator(seed);
poisson_distribution<int> distribution(2.5);
int number = distribution(generator);
while(number>5||number<0){
    number = distribution(generator);
}
this_thread::sleep_for(chrono::seconds(number));
```

四.UI 介紹

(一)

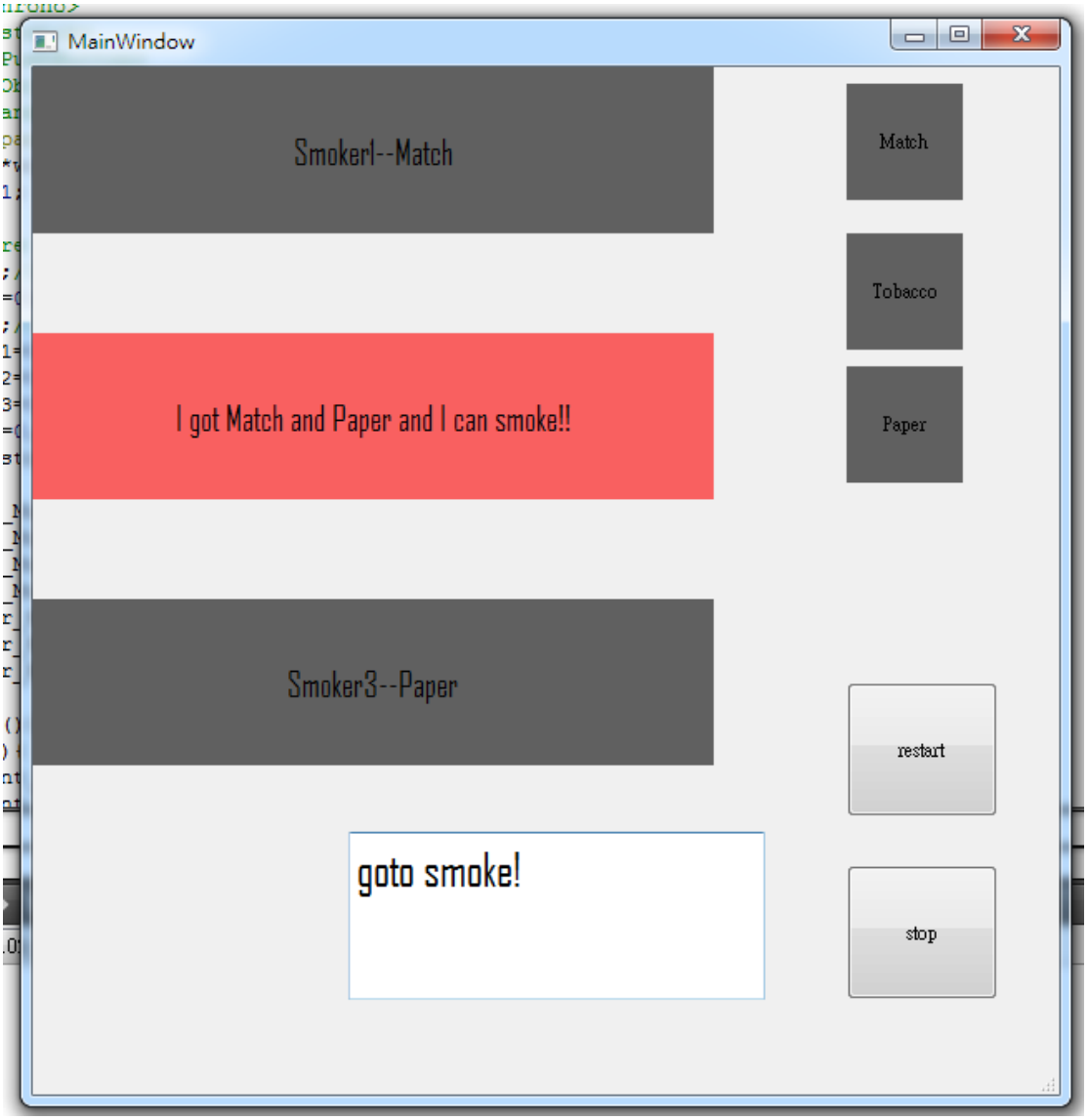


左上方為三個 Smoker，右上方為三個材料，下方為文字說明，亮起來的表示 Agent 提供放置桌上的兩個原料，程式一執行 Agent 便會發出兩個原料(圖為 Match、Paper)



(二)

三個 Smoker 會等待隨機 possion 分布時間後啟動測試是否可以取得完整三個原料，若無法取得會顯示 I can't smoke 一秒後暗調，可以取得的 Smoker(圖為 Smoker2 Tobacco)會繼續亮著執行抽菸。



(三) 可以取得的 Smoker(圖為 Smoker2 Tobacco)執行抽菸完成後，呼叫 Agent 啟動，重新抽取兩個原料

可隨時按下 stop 鍵可暫停程式看清界面顯示的變化，按下 restart 可恢復執行

五.心得

六.分工&進度表

11/02 建立 github	芷銓、漢龍、其祐
11/16 討論題目解法	芷銓、漢龍、其祐
11/30 分別使用 thread 完成簡易題目要求	芷銓、漢龍、其祐
12/07 增加 mutex	漢龍、其祐
12/14 新增 possion 分布	其祐

12/21 使用 QT 設計 ui 介面　　芷銓
12/28 改進程式配合 ui　　漢龍
1/04　更改細部文字顯示、書面報告　芷銓、漢龍、其祐