2018 Fall Embedded Operating System

Homework 03

請使用 C/C++完成以下程式，並且能在 Linux 的環境下執行

這次作業將以上一次的 overcook 作業為基礎，加上 multithread 或 multi-process 使程式能擁有多工處理的能力。

1. 我們要更改**讀取 order 的方式**。不再是讀取一個寫死的檔案，而是由鍵盤直接輸入，並且在程式執行的**任何時候**都要能夠輸入追加餐點。使用 multithread 或 multi-process 來負責鍵盤的輸入，並且讓其他的 thread/process 能繼續煮飯。請將 overcook.cpp 下面圖中的兩行註解的範例程式刪掉再開始

```
int main(){

//      char order_file[] = "orders.txt";
        char steps_file[] = "steps.txt";
        char tools_file[] = "tools.txt";

//      me.setOrder(order_file);
        me.setSteps(steps_file);
        me.setTool(tools_file);
```

2. 接下來要實現**多工煮飯**的部分。在**煮飯工具(tool)**與**食材原料充足**的情況下，程式要能有多位廚師同時進行料理，**最多四位最少兩位**。使用 multithread 或 multi-process 來完成。
3. 烹煮料理時螢幕需**輸出料理狀態**，基本上跟上次作業是一樣的格式，且”take”、”use”、”get”、”return”時不需要 delay
4. 當鍵盤輸入**”timeout”**的時候，螢幕輸出”end”並結束。不用很準確地立刻結束，但是要盡早結束。請把 cook.h 下面註解掉的那行刪掉再開始

```
/**
* Get one order from the order list and remove it from the list
**/
string getOrder(){
        if(orders.empty()){
                //cout << "end" << endl;
                return string();
        }
        string order = orders.back();
        orders.pop_back();
        return order;
}
```