

# CPSC 410

## Project 4

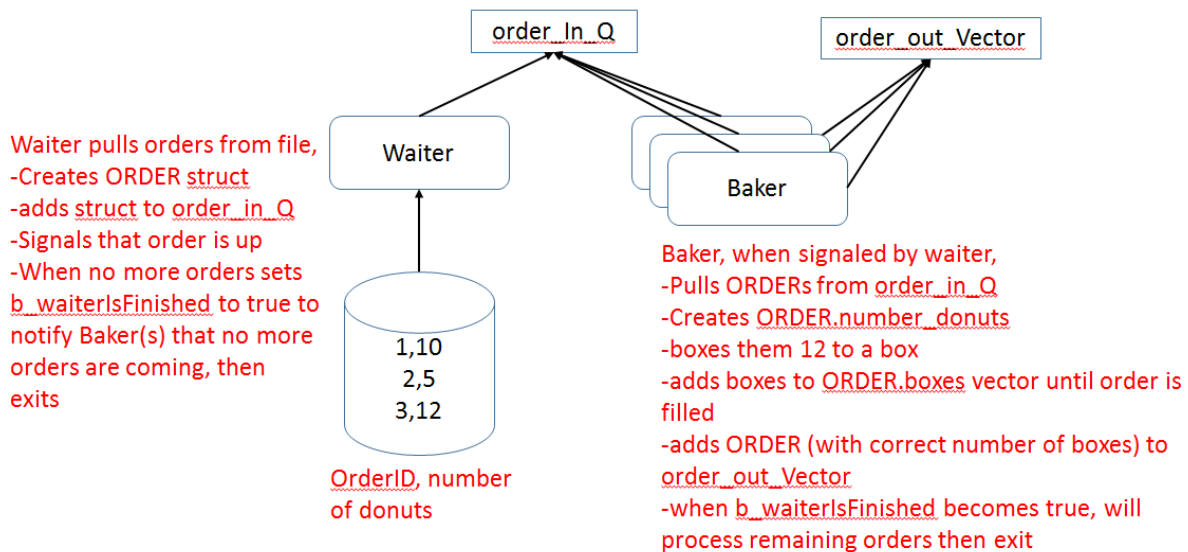
### Motivation: Topics covered by this project;

- Threads, mutexes, condition variables

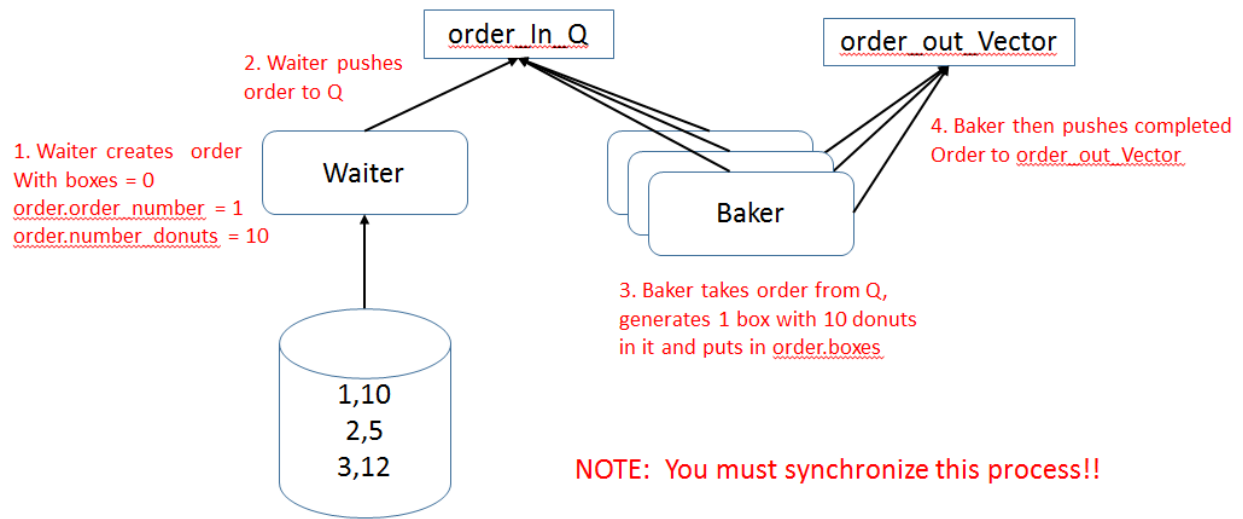
### Overview

The best ideas often occur while your daydreaming, where your wandering mind connects disparate bits of information in a way that produces something new and innovative. This is not one of those.

You are to model a donut order fulfillment system. A waiter takes orders for donuts, while bakers fill and box these orders (see below).



As an example, suppose the Waiter is pulling in order 1, for 10 donuts (see above). The process will follow this outline



## Teams

Please work 2 to a team. Please include a description of what each team member did as part of your submission. Percentage responsibility for the final product is fine.

## File\_io (all is given to you)

This set of functions duplicates what you did in project 2. Note that it contains orders now. You are given a file that has an unknown number of rows, and 2 columns of integers. Assume there are no malformed rows (ie != 2 columns). It looks like the following (mine will be much more elaborate).

1,10
2,5
3,3
5,15
10,11

This file can have any name, but I will call it testdata.txt for the purposes of this document. The first column is order\_number, second is number\_donuts . File\_IO provides functions to load this data into a vector and sort it by order\_number.

## Logger

You are given both a header and cpp file. Please make this class threadsafe. Please remember that File IO can also throw exceptions, plan your locking accordingly.

## Baker

You are given a header file and a stripped .cpp file. Please fill in the cpp file. Please see header file for function explanation.

Additional notes: Baker must not start processing orders until they are available. Do this by waiting for a signal from condition variable cv\_order\_inQ (defined in 410\_proj4.cpp and externed in externs.h). Baker will also wait on b\_WaiterIsFinished, when this happens no more orders are coming.

## Waiter

You are given a header file and a stripped .cpp file. Please fill in the cpp file. Please see header file for function explanation

Additional notes: You must use condition variable cv\_order\_inQ (defined in 410\_proj4.cpp and externed in externs.h) to signal the baker that there are orders available in the order\_in\_Q or to signal that there are no more orders available (b\_WaiterIsFinished)

## 410\_proj4.cpp

I have given you a stripped down implementation. Use this file to launch a thread for the waiter and multiple bakers. There is a useful function audit\_results() that should help you determine whether orders were filled properly.

## Other Information

Box.cpp and .h, used by baker when boxing donuts

Datastructs.h, holds ORDER info

PRINT.h contains helpful threadsafe print functions that you can use.

Externs.h contains external references to global variables, include wherever you need those global variables.

(extern is a promise to the compiler. It states that an externed variable is not defined in this file, but is in another. When all object files linked (remember compilation is preprocess->compile->link) the linker will find it. It's the only way to work with globals that must be shared amongst different compilation units without passing them as function references.)

## Documentation and Testing

Make sure you comment each function and the program as a whole. I recommend you test each module separately (which implies you write test code to verify its behavior).

## Directory Structure

Please do not modify the project directory structure

## To Turn In

waiter.cpp, baker.cpp and logger.cpp ONLY!

## Grading

I will drop your waiter.cpp, baker.cpp and logger.cpp into the appropriate places in my project. Therefore these are the only files that I need from you.

10% waiter.cpp, baker.cpp and logger.cpp files submitted as described in 'ToTurn In'

10% logger.cpp works correctly

25% condition variables correctly used for signaling between baker and waiter

30% 1 waiter and 1 baker work correctly in separate threads

25% 1 waiter and multiple bakers work correctly

**This assignment is complex and is weighted 2.5 times the weight of project 1**

### Other Hints

I will test with many bakers so use condition variable `notify_all()`.