# Отчет по домашнему заданию №4

## Общая инфо

ФИО: Афонин Александр Сергеевич
Группа: БПИ 217
Вариант 19

## 4 балла

- Условие:
  ***Задача о магазине - 2 (забывчивые покупатели).*** *В магазине работают два отдела, каждый отдел обладает уникальным ассортиментом. В каждом отделе работает один продавец. В магазин ходят исключительно забывчивые покупатели, поэтому каждый покупатель носит с собой список товаров, которые желает купить. Покупатель приобретает товары точно в том порядке, в каком они записаны в его списке. При этом товары в списке расположены в случайном порядке, что заставляет покупателя переходить от отдела к отделу, если это требуется для совершения покупок. Продавец может обслужить только одного покупателя за раз. Покупатель, вставший в очередь, засыпает пока не дойдет до продавца. Продавец засыпает, если в его отделе нет покупателей, и просыпается, если появится хотя бы один.* ***Создать многопоточное приложение, моделирующее работу магазина в течение рабочего дня.***
- Используется модель **Клиенты и серверы** – способ взаимодействия неравноправных потоков. Клиентский поток *(покупатели)* запрашивает сервер *(продавца)* и ждет ответа. Серверный поток ожидает запроса от клиента, затем действует в соответствии с поступившим запросом.
- Входные данные можно описать в следующем формате:
  Сначала вводится целое положительное число $N$ – количество покупателей. Затем вводятся $N$ строк. Каждая $i$-я строка начинается с целого положительного числа $K_i$ – количества покупок $i$-го покупателя и далее через пробел вводятся числа $p_{i,j}$ – номер кассы (1 или 2), где нужно совершить $i$-ю покупку.
- Данные можно вводить через консоль. **(./main.exe CONSOLE_IO)**

## 5 баллов

- Добавлены комментарии как для переменных, так и для блоков программы.
- Можно выделить следующие сущности: покупатель и продавец. Покупатель имеет список покупок, охарактеризованных как список продавцов, у которых он должен взять товар. Но так как я решил ограничиться двумя сущностями, то можно интерпретировать список товаров, как список продавцов. Покупатель смотрит на очередной элемент из списка и всегда становится в конец очереди к соответствующему продавцу. Покупатели работают параллельно друг другу, если это возможно. Продавцы же работают тогда, когда в соответствующей им очереди больше одного покупателя, и также работают параллельно друг другу.

## 6 баллов

- Упорядочим сущности, с помощью идентификаторов для удобства. Также каждая сущность имеет свой поток, для выполнения параллельности. Т. е. $i$-й покупатель имеет контроль над потоком $ct_i$, и $j$-й покупатель имеет контроль над потоком $st_j$. Реализовывая модель задачи, имеем следующий алгоритм для потока покупателей:
  1) Обходим всех продавцов по соответствующему покупателю списку.

2) Покупатель “встает в очередь” к продавцу. Можно реализовать это с помощью mutex’ов. Т.е. пока все потоки-покупатели, которые еще не обработались, подошедший покупатель обработан не будет.
3) Когда настает очередь текущего потока-покупателя, он делает обращение в соответствующей текущему продавцу потоку и ждет ответ.

А алгоритм для потока продавца:

1) Поток-продавец ждет запрос от потока-покупателя на обработку продажи.
2) Когда приходит запрос он сразу обрабатывается, и потом поток-продавец ждет следующего запроса и возвращается к п. 1.

- Из командной строки вводится формат ввода данных

## 7 баллов

- Добавлен файловый ввод/вывод. **(./main.exe FILE_IO <input> <output>)**
- Результат файлового вывода также выводится в консоль
- Тесты:

| Ввод | Вывод |
|---|---|
| 6<br>1 2<br>2 1 1<br>2 1 2<br>4 1 2 2 1<br>1 1<br>1 1 | [0.000523]: Customer #1 moved to Seller #2<br>[0.001297]: Seller #2 started selling for Customer #1<br>[0.001447]: Customer #2 moved to Seller #1<br>[0.002439]: Seller #1 started selling for Customer #2<br>[0.002649]: Customer #3 moved to Seller #1<br>[0.002841]: Customer #4 moved to Seller #1<br>[0.003077]: Customer #5 moved to Seller #1<br>[0.003446]: Customer #6 moved to Seller #1<br>[3.996620]: Seller #2 ended selling for Customer #1<br>[3.996703]: Seller #1 ended selling for Customer #2<br>[4.000535]: Customer #2 moved to Seller #1<br>[4.000602]: Seller #1 started selling for Customer #2<br>[7.993028]: Seller #1 ended selling for Customer #2<br>[7.993131]: Seller #1 started selling for Customer #4<br>[11.989152]: Seller #1 ended selling for Customer #4<br>[11.993577]: Customer #4 moved to Seller #2<br>[11.993673]: Seller #1 started selling for Customer #5<br>[11.998162]: Seller #2 started selling for Customer #4<br>[15.989398]: Seller #1 ended selling for Customer #5<br>[15.989444]: Seller #2 ended selling for Customer #4<br>[15.994167]: Customer #4 moved to Seller #2<br>[15.994357]: Seller #2 started selling for Customer #4<br>[15.994490]: Seller #1 started selling for Customer #6<br>[19.985101]: Seller #2 ended selling for Customer #4<br>[19.990198]: Customer #4 moved to Seller #1<br>[19.990249]: Seller #1 ended selling for Customer #6<br>[19.990332]: Seller #1 started selling for Customer #3<br>[23.980250]: Seller #1 ended selling for Customer #3<br>[23.985700]: Customer #3 moved to Seller #2<br>[23.991163]: Seller #2 started selling for Customer #3<br>[23.991284]: Seller #1 started selling for Customer #4<br>[27.980385]: Seller #2 ended selling for Customer #3<br>[27.980438]: Seller #1 ended selling for Customer #4 |
| 3<br>3 2 1 2<br>2 2 2<br>4 1 1 2 2 | [0.000825]: Customer #1 moved to Seller #2<br>[0.001524]: Seller #2 started selling for Customer #1<br>[0.001652]: Customer #2 moved to Seller #2<br>[0.002128]: Customer #3 moved to Seller #1<br>[0.003287]: Seller #1 started selling for Customer #3<br>[3.995669]: Seller #2 ended selling for Customer #1<br>[3.999665]: Customer #1 moved to Seller #1<br>[3.999801]: Seller #2 started selling for Customer #2 |

| | |
|---|---|
| | [3.999838]: Seller #1 ended selling for Customer #3<br>[4.004191]: Customer #3 moved to Seller #1<br>[4.004344]: Seller #1 started selling for Customer #1<br>[7.996051]: Seller #2 ended selling for Customer #2<br>[8.000325]: Customer #2 moved to Seller #2<br>[8.000390]: Seller #2 started selling for Customer #2<br>[8.000406]: Seller #1 ended selling for Customer #1<br>[8.004983]: Customer #1 moved to Seller #2<br>[8.005213]: Seller #1 started selling for Customer #3<br>[11.996419]: Seller #2 ended selling for Customer #2<br>[11.996551]: Seller #2 started selling for Customer #1<br>[11.996586]: Seller #1 ended selling for Customer #3<br>[12.001415]: Customer #3 moved to Seller #2<br>[15.991999]: Seller #2 ended selling for Customer #1<br>[15.992102]: Seller #2 started selling for Customer #3<br>[19.987360]: Seller #2 ended selling for Customer #3<br>[19.992453]: Customer #3 moved to Seller #2<br>[19.992558]: Seller #2 started selling for Customer #3<br>[23.987526]: Seller #2 ended selling for Customer #3 |
| 2<br>1 1<br>1 1 | [0.000509]: Customer #1 moved to Seller #1<br>[0.000991]: Seller #1 started selling for Customer #1<br>[0.001172]: Customer #2 moved to Seller #1<br>[3.992166]: Seller #1 ended selling for Customer #1<br>[3.992293]: Seller #1 started selling for Customer #2<br>[7.987273]: Seller #1 ended selling for Customer #2 |

- Командная строка также поддерживает файловый ввод через параметр **FILE_IO**.

## 8 баллов

- Добавлена генерация случайных данных **(./main.exe RANDOM [<optional-seed>])** в след. диапазонах:
    1) $0 < N \leq 10$,
    2) $0 < K_i \leq 4$,
    3) $p_{i,j} \in \{1, 2\}$.
- См. пункт на 7 баллов
- Ввод дополнен с помощью аргумента **RANDOM**