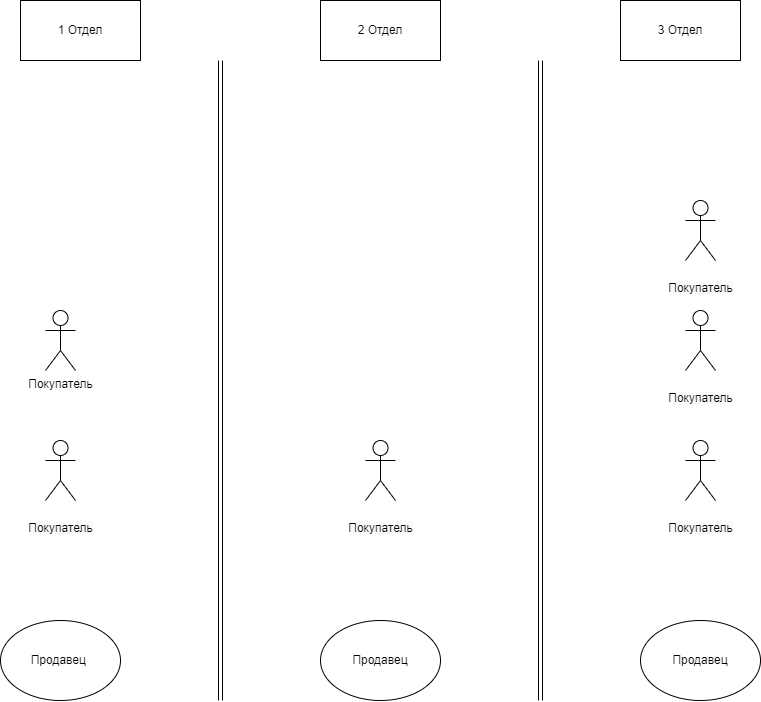
**Жалилов Актан БПИ223**

**Вариант 13**

**Условие:**

В магазине работают три отдела, каждый отдел обслуживает один продавец. Покупатель, зайдя в магазин, делает покупки в одном или нескольких произвольных отделах, обходя их в произвольном (случайном) порядке. Если в выбранном отделе продавец не свободен, покупатель становится в очередь и ожидает, пока продавец не освободится. Создать многопоточное приложение, моделирующее рабочий день магазина. Каждого покупателя и продавцов моделировать отдельными потоками. Размер очереди не оговаривается. Считается, что для данной задачи она не ограничена (но моделирование должно быть в разумных пределах).

**Сценарий:**



Имеется 3 продавца (по 1 в каждом отделе), каждый из них свободен до тех пор, пока не подойдет хотя бы один покупатель, так же существует несколько покупателей (их количество пользователь ввод сам). Покупатель может посетить от 1 до 3 отделов (при этом покупатель может посетить один и тот же одел несколько раз). Если продавец свободен покупатель вызывает его и передает ему свой номер, после того как продавец его обслуживает покупатель покидает отдел, если же продавец занят, покупатель становиться в очередь (очередь очень плохая так как все ее нарушают и идут не по порядку прибытия) и стоит до тех пор, пока его не обслужит продавец. После того как все покупатели покинули магазин, рабочий день заканчивается.

**Модель параллельных вычислений:**

**Общая Память:** Потоки делят общие глобальные переменные для взаимодействия и обмена данными.

**Синхронизация:** Мьютексы используются для предотвращения одновременного доступа к общим данным, а условные переменные контролируют порядок работы потоков.

**Потоки для Покупателей и Продавцов:** Отдельные потоки создаются для покупателей и продавцов, каждый из которых выполняет свои задачи параллельно.

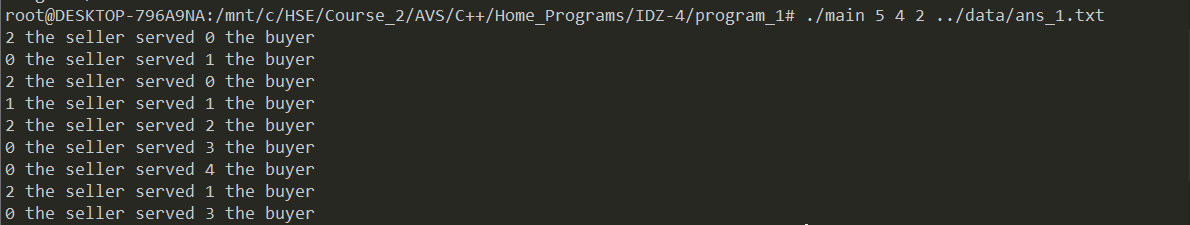
**Взаимодействие и Циклическое Поведение:** Потоки взаимодействуют через общие структуры данных и синхронизацию, с бесконечным циклом для продавцов и управляемым завершением для покупателей.

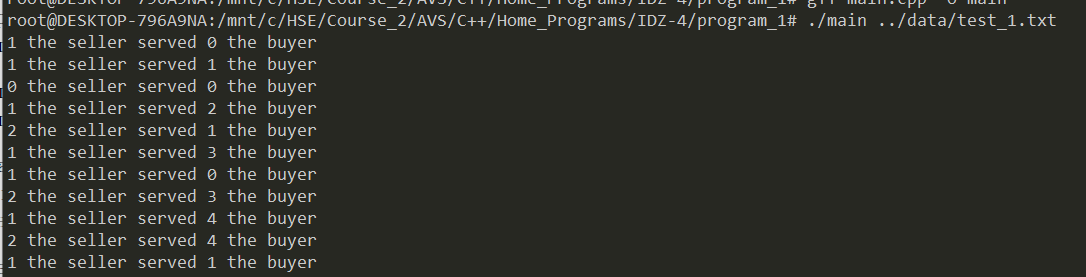
**Входные данные:**

Существует два варианта входных данных через командную строку:

1. Пользователь вводит путь к конфигурационному файлу, в котором содержаться количество покупателей, задержка потока покупателя (от 1 до 7 секунд), задержка потока продавца (от 1 до 7 секунд) и путь к файлу, в который необходимо записать результат.
2. Количество покупателей, задержка потока покупателя (от 1 до 7 секунд), задержка потока продавца (от 1 до 7 секунд) и путь к файлу, в который необходимо записать результат.

**Результаты работы программы:**





**Обобщённый алгоритм:**

Задаются глобальные переменные, такие как задержки для потоков покупателей (buyerSleep) и продавцов (salesmanSleep), буфер (buffer) для хранения информации о взаимодействии между покупателями и продавцами, и массив сообщений (answer) для логирования действий.

Инициализируются семафоры (needed, available) для управления доступом к продавцам и мьютекс (mutex) для синхронизации доступа к общему массиву сообщений.

Чтение конфигурационных данных из файла или командной строки, установка количества покупателей, задержек и пути к файлу вывода.

Для каждого продавца создается поток (Salesman), который ожидает вызова от покупателя.

Для каждого покупателя создается поток (Buyer), который взаимодействует с продавцами.

**Работа продавца (Salesman):**

Ожидание вызова от покупателя (семафор needed).

При вызове обрабатывается запрос покупателя, логируется действие, и продавец становится доступным (семафор available).

**Работа покупателя (Buyer):**

Покупатель выбирает продавца случайным образом и ожидает, пока он станет доступен. После того как продавец доступен, покупатель взаимодействует с ним и вызывает его (семафор needed).

После обработки всех покупателей, потоки покупателей завершаются.

Потоки продавцов должны быть явно завершены, так как они работают в бесконечном цикле.

Все события, происходящие в процессе работы программы, логируются в общий массив answer.

По завершении работы сообщения записываются в файл.

**Генерация случайных чисел:**

Используется функция randomInteger куда передается максимально и минимальное из возможных чисел. Она нужна чтобы определить сколько отделов покупатель посетит и какой именно отдел покупатель посетит в данный момент.

**Вторая программа:**

Условные Переменные и Мьютексы:

Для каждого продавца определены два набора: условная переменная (pthread\_cond\_t) и мьютекс (pthread\_mutex\_t). Один набор (needed и needed\_mutex) используется для управления вызовами продавца покупателями, а другой (available и available\_mutex) - для индикации доступности продавца.

Функция Salesman (Продавец):

Каждый поток продавца ожидает, пока покупатель не обратится к нему. Он использует pthread\_cond\_wait на условной переменной needed, что позволяет потоку спать, пока не будет получен сигнал от покупателя.

После обслуживания покупателя продавец отправляет сигнал через условную переменную available, сообщая, что он теперь свободен.

Функция Buyer (Покупатель):

Покупатель, выбирая продавца, сначала проверяет, свободен ли он, используя pthread\_cond\_wait на условной переменной available.

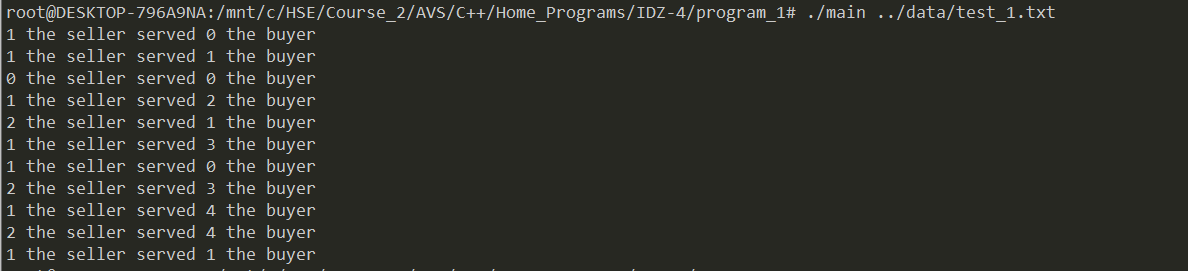
После того как продавец становится доступен, покупатель обновляет данные в buffer, затем отправляет сигнал продавцу через условную переменную needed, чтобы начать взаимодействие.

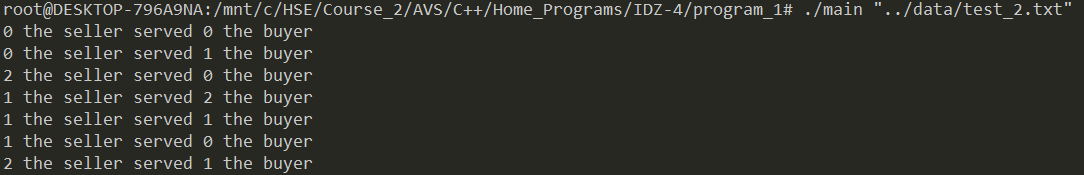
Синхронизация Доступа к Общим Данным:

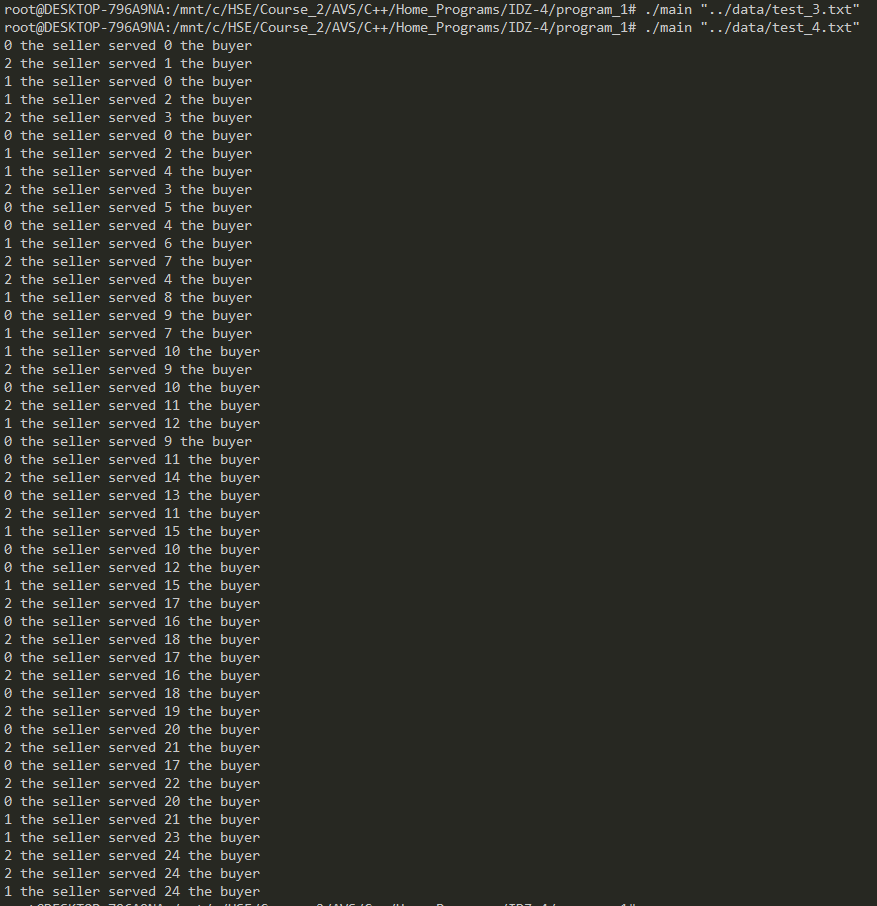
Перед чтением или записью в общие данные (buffer, sellerBusy), потоки блокируют соответствующий мьютекс, чтобы предотвратить гонки данных и обеспечить целостность данных.

**Сравнительный анализ:**

При просмотре результатов программ можно понять, что выводы программ не совпадают, следовательно, их работа не идентична, но при этом структура вывода одинакова, что говорит о том, что они решают одну и ту же задачу.

**1-я программа:**

**2-я программа:**



**2-я программа:**

