Правительство Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования «Национальный исследовательский университет «Высшая школа экономики»

Факультет компьютерных наук

Департамент программной инженерии

Отчет к Микропроекту №2

**«Многопоточное приложение с использованием стандартной библиотеки С++»**

По дисциплине

«Архитектура вычислительных систем»

Работу выполнил:

Студент группы БПИ-194 Романюк А. С.

Вариант 21

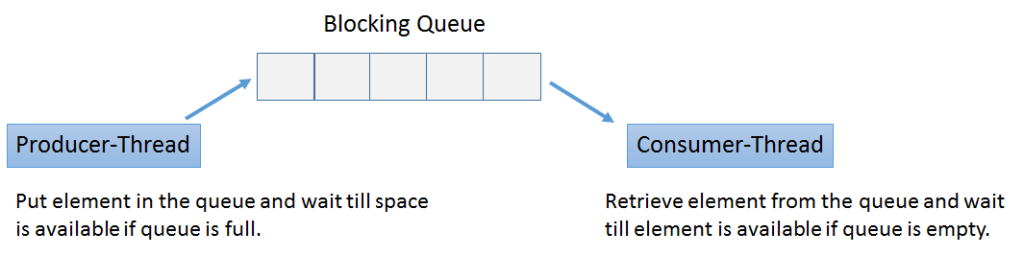
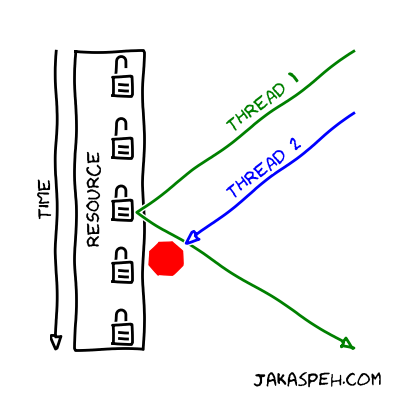
**Москва 2020**

# Условие

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. 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.

1. **Решение**

Для реализации программы использовалась стандартная библиотека С++ для работы с потоками. Ниже приведен разбор условия и реализация конкретного пункта в приложении.

1. *«В магазине работают два отдела, каждый отдел обладает уникальным ассортиментом».* В исходном коде программы был реализовал класс **Department (Producer)**, который выполняет роль отдела в магазине. В главном потоке у пользователя запрашивается число “**countProducts**” – количество продуктов, распределенных между двумя отделами. Ограничение на пользовательский ввод: ***[minProduct, maxProduct].*** Чтобы ассортимент был уникальным: создаётся коллекциях из элементов ***0..countProducts***, а дальше случайным образом выбирается ровно **(countProducts / 2)** элементов и загружаются в первый отдел магазина, а все остальные – во второй, таким образом, все продукты – уникальны.
2. *«В каждом отделе работает один продавец».* Для каждого отдела выделяем по одному потоку.
3. *«В магазин ходят исключительно забывчивые покупатели, поэтому каждый покупатель носит с собой список товаров, которые желает купить».* Для каждого покупателя, которых ровно n штук (задаётся через консольный ввод) случайным образом генерируется его список покупок из K товаров, где K ∈ **[customerMin; customerMax].**
4. *«Покупатель приобретает товары точно в том порядке, в каком они записаны в его списке».* Покупатель в отдельном потоке проходится по всем элементам коллекции списка товаров и делает запросы в нужный отдел и ожидает ответа от продавца нужного отдела, пока тот не ответит, покупатель не пойдёт за другим товаром.
5. *«Продавец может обслужить только одного покупателя за раз»*. Тут использовался стандартный шаблон «Producer – Consumers», реализация которой приходится на очередь. А для того, чтобы обеспечить безопастное получение данных из очереди – использовались мьютексы.
6. *«Покупатель, вставший в очередь, засыпает пока не дойдет до продавца».* Для реализации данного пункта использовались мьютексы для текущего покупателя отдела – std::unique\_lock<std::mutex>.
7. *«Продавец засыпает, если в его отделе нет покупателей, и просыпается, если появится хотя бы один».* Сам поток продавца устроен как бесконечный цикл, но благодаря переменной **isSleep** и condition\_variable::wait() мы можем контролировать его засыпание на неопределенное время, пока другой поток его не разбудит. А разбудить его может только покупатель c помощью метода condition\_variable::notify\_one().
8. **Тестирование**

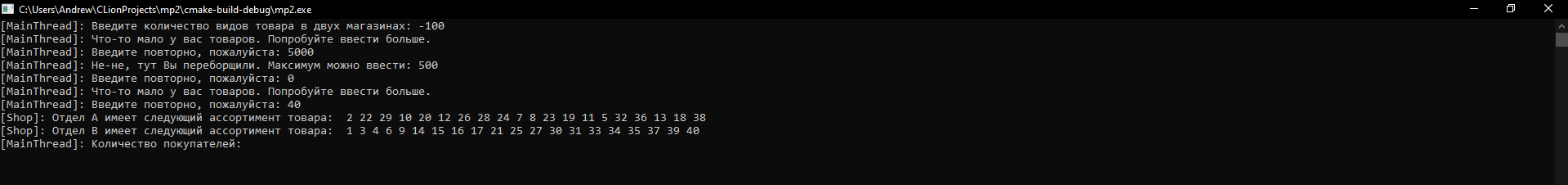


Рисунок 1 ­– Результат обработки некорректного ввода и работы с пользовательским вводом.

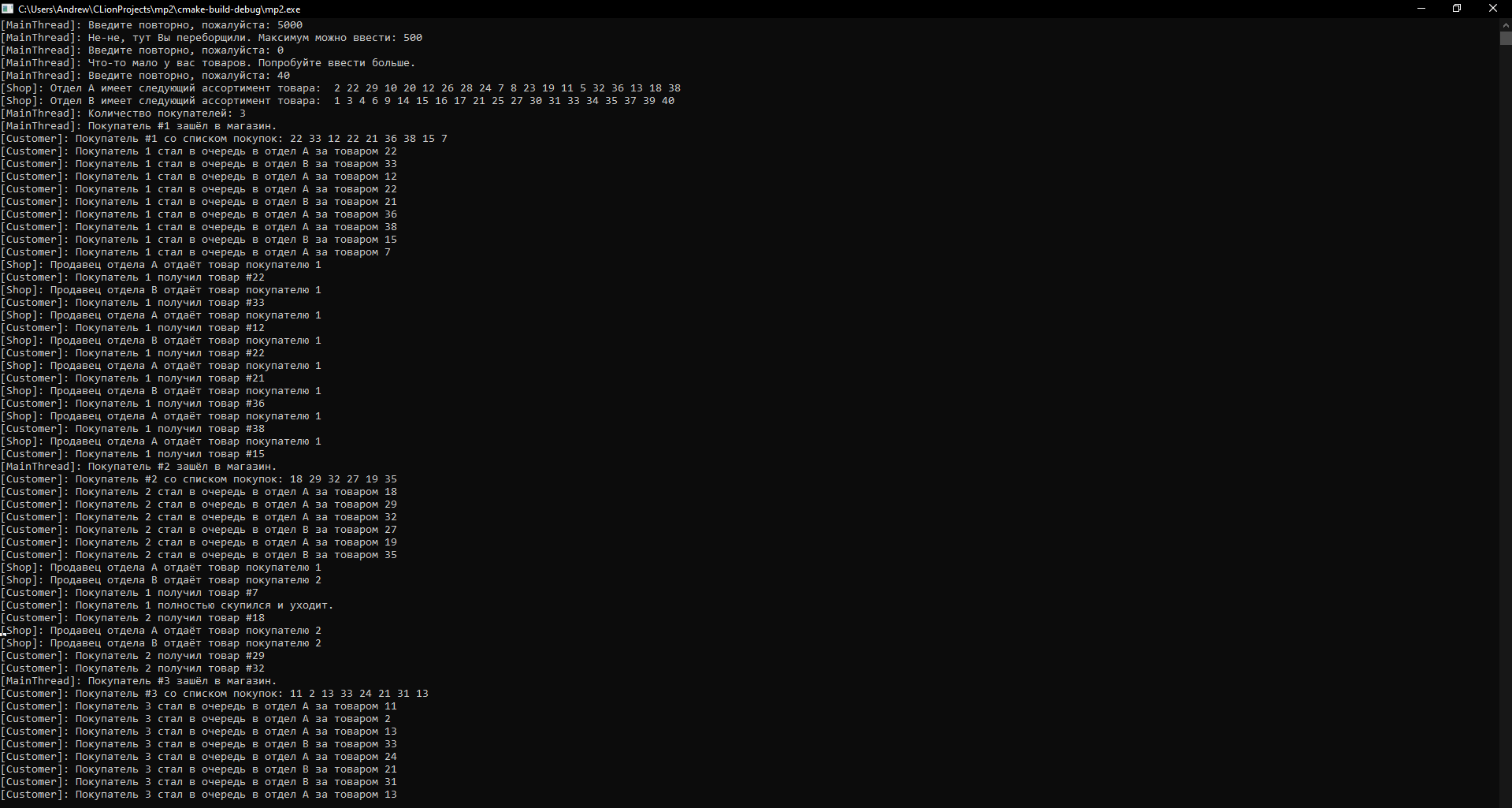
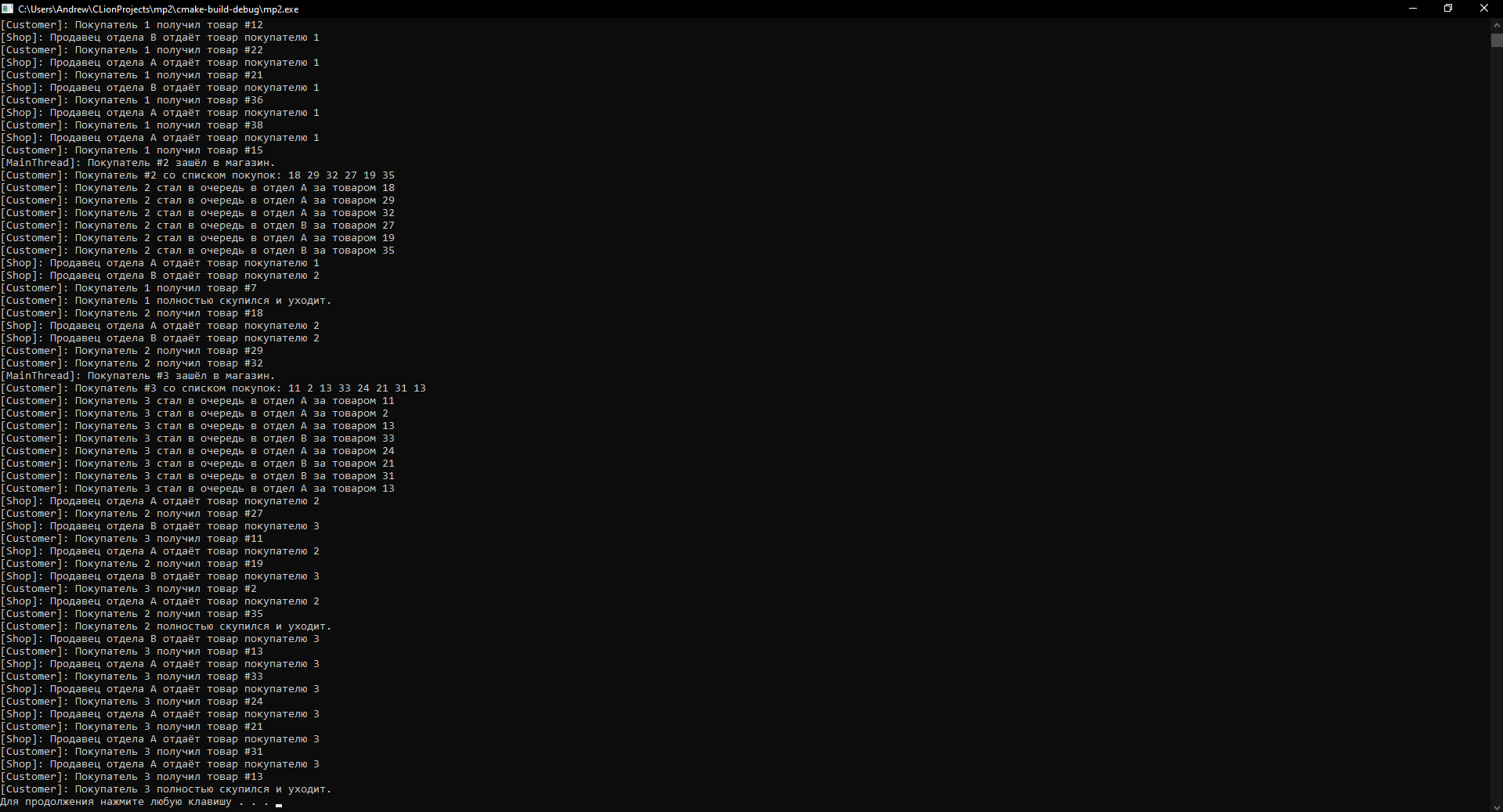
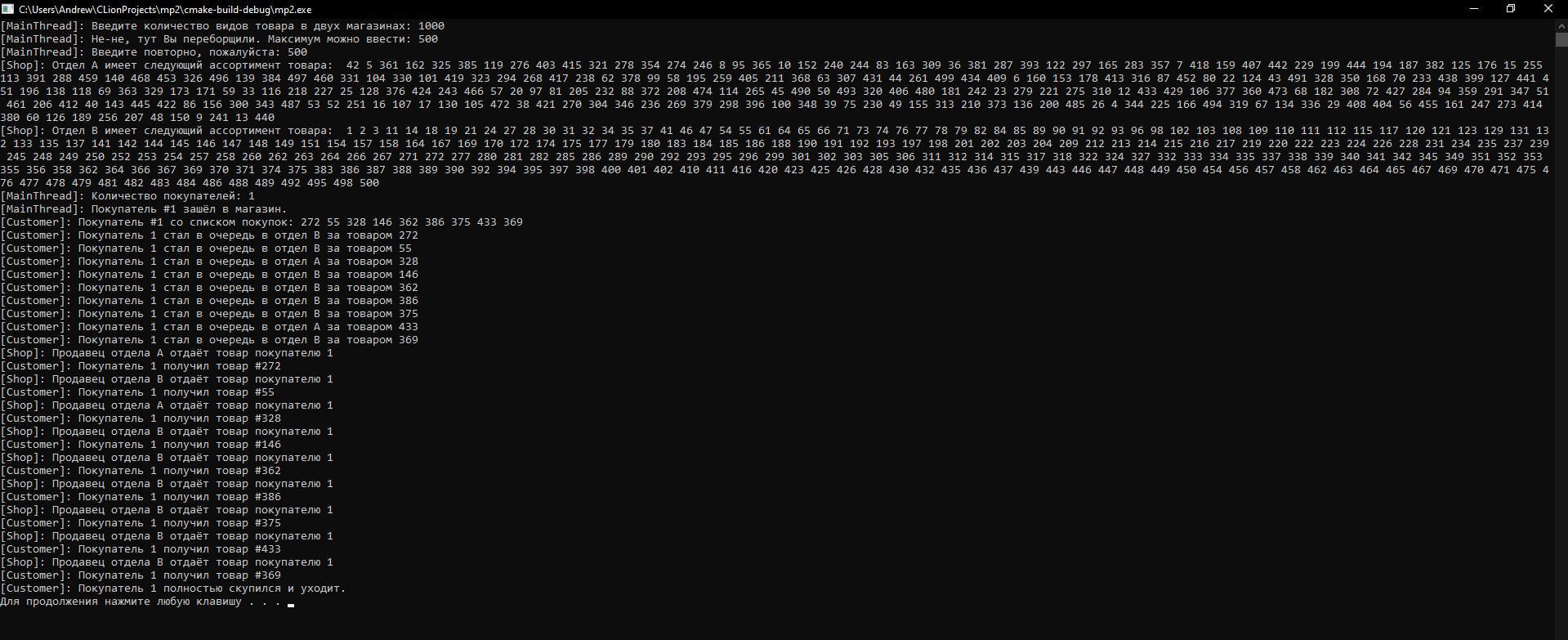


Рисунок 2 ­– Процесс работы приложения.



Рисунок 3 ­– Продолжение работы приложения и её завершение.

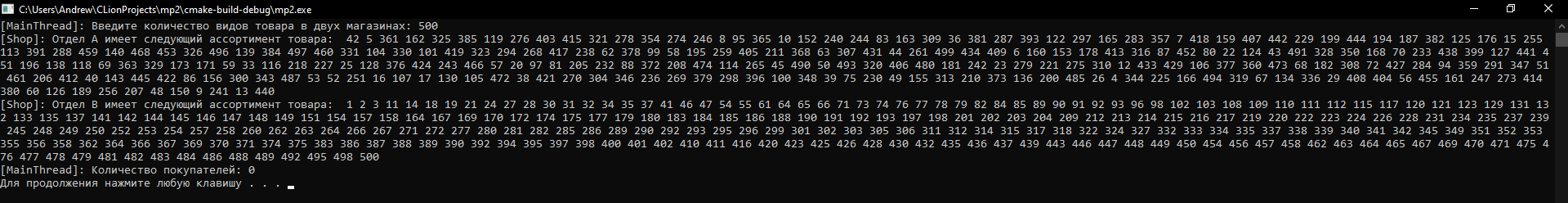
Рисунок 4 ­– Результат выполнения приложения.

Рисунок 5 ­– Результат ввода 0.

**Список используемых источников**

1. Cppreference (2020) «Документация по С++: std::condition\_variable::wait» (<https://en.cppreference.com/w/cpp/thread/condition_variable/wait>). Просмотрено 03.12.2020
2. Cppreference (2020) «Документация по С++: std::mutex» (<https://ru.cppreference.com/w/cpp/thread/mutex>). Просмотрено 03.12.2020
3. Docs Microsoft (2020) «Creating Threads» (<https://docs.microsoft.com/en-us/windows/win32/procthread/creating-threads>). Просмотрено 03.12.2020
4. Habr (2020) «Потоки, блокировки и условные переменные в C++11» (<https://habr.com/ru/post/182626/>). Просмотрено 03.12.2020
5. Intel (2011) «Шаблон проектирования Producer-Consumer» (<https://software.intel.com/content/www/ru/ru/develop/articles/producer-consumer.html>). Просмотрено 03.12.2020
6. Wikipedia (2014) «Producer–consumer problem» (<https://en.wikipedia.org/wiki/Producer%E2%80%93consumer_problem>). Просмотрено 03.12.2020
7. Легалов А.И.(2020) «Архитектура параллельных вычислительных систем. Многопоточность » (<http://softcraft.ru/edu/comparch/lect/07-parthread/>) Просмотрено 03.12.2020
8. Легалов А.И.(2020) «Многопоточность. Простая многопоточная программа. Основные функции» (<http://softcraft.ru/edu/comparch/practice/thread/01-simple/>). Просмотрено 03.12.2020
9. Легалов А.И.(2020) «Многопоточность. Синхронизация потоков. Методы синхронизации» (<http://softcraft.ru/edu/comparch/practice/thread/02-sync/>). Просмотрено 03.12.2020