Лабораторная работа №1

Ответы на вопросы

1. Привести определения процесса и потока.

   Процесс (process) — объект операционной системы, который хранит все ресурсы процесса, их дескрипторы, потоки и т.д. Каждый процесс имеет как минимум один поток. Каждый процесс имеет свое собственное виртуальное адресное пространство и контекст выполнения, а потоки созданные одним процессом разделяют его адресное пространство и не имеют доступ к пространству других процессов.

   Поток (thread) — это, сущность операционной системы. Общее назначение потоков — параллельное выполнение на процессоре двух или более различных задач.

2. Сколько потоков и в каком порядке создается в ходе работы приложения?

   В моей программе создается 4 потока. Порядок создания потоков:

   ```
    1. main - главный поток при запуске приложения. Данный поток обрабатывает очередь сообщений для обработки событий, например, вывода чисел, полученные потоком thread_reader, обновления интерфейса. Также данный поток создает потоки thread_generate, thread_generate2 и thread_reader.

    2. thread_generate - первый поток генерации чисел

    3. thread_generate2 - второй поток генерации чисел

    4. thread_reader - поток чтения чисел из буфера и отправка потоку main для отображения в интерфейсе
   ```

3. Чем определяется порядок выполнения потоков? Какая дисциплина используется?

   Поряд выполнения потоков определяется **операционной системой**. Операционная система использует дисциплину планирования **многоуровневой очереди с обратной связью**. Это означает, что потоки с *высоким приоритетом* будут выполняться раньше, чем потоки с *низким приоритетом*. С обратной связью означает что приоритет потока может быть изменен в зависимости от его состояния. Например, если поток с высоким приоритетом пребывает в состоянии ожидания, то его приоритет будет уменьшен. Поэтому данный алгоритм планирования является **вытесняющим**.

4. Описать все смены состояний потоков в ходе работы приложения.

5. Какова дисциплина обслуживания буфера и почему выбрана именно она?

   Дисциплина обслуживания буфера - **FIFO**. Это означает, что элементы буфера будут обрабатываться в том порядке, в котором они были добавлены в буфер. Данная дисциплина

была выбрана потому, что она является наиболее простой и понятной. Также данная дисциплина обслуживания буфера является **блокирующей**. Это означает, что если буфер заполнен, то поток, который пытается добавить элемент в буфер, будет заблокирован до тех пор, пока не освободится место в буфере.

6. Какие операции доступа к буферу должны синхронизироваться и почему?

   Доступ к буферу должен синхронизироваться потоками, которые добавляют элементы в буфер. Это необходимо для того, чтобы избежать **гонки данных**. Гонка данных возникает, когда два или более потоков одновременно пытаются изменить одну и ту же переменную. В результате гонки данных, значение переменной может быть изменено неправильно. Для того, чтобы избежать гонки данных, необходимо синхронизировать доступ к буферу.

7. Какой механизм выбран для реализации синхронизации и почему? Чем он отличается от классического семафора Дейкстры?

   Для реализации синхронизации была выбрана **критическая секция**. В отличие от семафора, доступ к разделяемым ресурсам будет предоставлен только одному потоку, как и при использовании мьютекса, но критические секции обеспечивают более быстрый и более эффективный механизм синхронизации. В операционных системах Windows разница между семафором, мьютексом и критической секцией в том, что семафор и мьютекс является объектами ядра и могут быть использованы несколькими процессами одновременно, критическая секция же выполняется в контексте пользователя, принадлежит одному процессу и служит для синхронизации только его потоков.

   0 - закрытое состояние, любое другое - открытое Три операции: I - установка начального состояния семафора P - операция закрытия семафора V - операция открытия семафора