Use java

a) Topic: Processes, streams (threads).

Task: Compose a program (process) that runs two background threads (threads) in parallel. Use thread synchronization on a shared resource.

Description of the program:

When starting the program with the "Start" button, two threads Tthread1 and TThread2 are simultaneously started in parallel, which try to set the "slider" to its position (1st position 10, 2nd position 90)

Threads can be prioritized, and depending on the priority, preference is given to one or the other thread

Threads are destroyed when the program terminates

b) Topic: Management of processes, streams (threads) in the critical section using a blocking variable (the simplest semaphore)

In the task, use the program of task 1

Make the following changes in the program:

Enter a global variable for a semaphore of integer type

Place the START 1 and START 2 buttons to start the first and second streams (threads), before that the semaphore is set to the "busy" position

Place the STOP 1 and STOP 2 buttons to stop the first and second streams (threads), the semaphore is set to the "free" position

The START 1 button sets the first thread to the lowest priority

The START 2 button sets the second thread to the highest priority

Description of the program

Threads are started sequentially. If one of the threads is running, the other cannot be started because the critical section is busy and displays the message "Thread Busy"

The Stop button releases the critical section and destroys the current thread.

The START button starts a thread and blocks pressing the STOP button of another thread

The correct operation of the program is as follows: The START button 1 sets the slider to position 10, where it remains until we press the START button 2, which sets it to position 90

Pay attention! A semaphore (blocking variable) is a global variable available to both threads, so they work in the same address space (of a given process).

If the semaphore regulates the interaction of not flows, but processes, then it should be global in relation to them and thus be in the address space of the operating system that manages the processes