**Programming Question**

**Ques 1:** **(30 Points )**

Three processes are involved in printing a file. Process A reads the file data from the disk to Buffer 1, Process B copies the data from Buffer 1 to Buffer 2, finally Process C takes the data from Buffer 2 and print it. Assume all three processes operate on one (file) record at a time, both buffers' capacity are one record. Write a program to coordinate the three processes using semaphores.

Solution:

Design:

Input a text file, every time Process A read a line from the text file and store it into buffer1, then Process B copy the line from buffer1 to buffer2, last Process C print the line in buffer2.

Here one record means one line in the text file and the capacity of buffer1 and buffer2 is one line of text.

1. There are 5 java documents in this program folder and one text for testing, namely:

Main.java: Creates and starts three processes;

Reader.java: Read file data from disk to Buffer1;-------------Process A

Converter.java: Copies data from Buffer1 to Buffer2;------------Process B

Printer.java: Print data in Buffer2;--------------------------------Process C

Semaphore.java: Define four semaphores used for the assignment to control the process odder.

2. To ensure the execution order of Process A -> Process B-> Process C, two Semaphore are defined, namely:

**public** **static** Semaphore *readBufferEmpty* = **new** Semaphore(1);

**public** **static** Semaphore *readBufferFull* = **new** Semaphore(0);

**public** **static** Semaphore *writeBufferEmpty* = **new** Semaphore(1);

**public** **static** Semaphore *writeBufferFull* = **new** Semaphore(0);

In Process A:

**Semaphores.readBufferEmpty.acquire();**

**read();**

**Semaphores.readBufferFull.release();**

In Process B:

**Semaphores.readBufferFull.acquire();**

**Semaphores.writeBufferEmpty.acquire();**

**convert();**

**Semaphores.readBufferEmpty.release();**

**Semaphores.writeBufferFull.release();**

In Process C:

**Semaphores.writeBufferFull.acquire();**

**print();**

**Semaphores.writeBufferEmpty.release();**

By using Semaphores: *readBufferEmpty, readBufferFull, writeBufferEmpty, writeBufferFull*, the execution order of Process A, Process B and then Process C can be guaranted.

3. Result Analysis:

Input:

I

am

testing

1

2

3

4

5

Without using Semaphores, the Process C only prints the last record of the text file like following:

Main Thread

Reader starts

Converter starts

Printer starts

Every Step of Print result: 5

Every Step of Print result: 5

Every Step of Print result: 5

Every Step of Print result: 5

Every Step of Print result: 5

Every Step of Print result: 5

Every Step of Print result: 5

Every Step of Print result: 5

After using Semaphores, the Process C only prints every record of the text file in right order like following:

Main Thread

Reader starts

Converter starts

Printer starts

Every Step of Print result: I

Every Step of Print result: am

Every Step of Print result: testing

Every Step of Print result: 1

Every Step of Print result: 2

Every Step of Print result: 3

Every Step of Print result: 4

Every Step of Print result: 5

(Here I use a flag, which equals to the length of records in the input text file to make sure the termination of all the threads. In the above case flag = 8)