**CS 3343 Operating Systems Assignment 3 12 points**

**Due February 14 at 5pm**

**One homework assignment submission per student. Microsoft Word format only. No AI or GPT use. Cite all references.**

**Email your answers to me at** [**harringp@nsuok.edu**](mailto:harringp@nsuok.edu)

**Send emails exclusively via** [**https://mail.google.com/**](https://mail.google.com/)

**Sign in with your NSU email and password**

**Chapter 3**

**Part 1 : Use the slides or textbook definitions to answer the following (8 points)**

1. Describe the following parts of a process: **(Book: 106)**
   1. Stack

The stack contains temporary data such as functions, parameters, return addresses, and local variables

* 1. Text section

Usually, the program code of a process

* 1. Heap

Memory that is dynamically allocated during process run time

* 1. Data section

The data section contains global variables

1. List the different states of a process. **(Ch3 Slides: 6)**

According to the slides we have the states (new, running, waiting, ready, and terminated). These states are pretty self-explanatory. New means the process is being created, running means the instructions are being executed, waiting means the process is waiting for an event to occur, ready means the process is waiting to be assigned to a processor, and termination means the process has finished execution.

1. What is the purpose of the process control block? **(Book: 107 - 108)**

The process control block contains many pieces of info associated with a specific process.

including: Process state, program counter, CPU registers, CPU-scheduling information, memory-management information, accounting information, and I/O status information.

Also, the PCB serves as the repository for any info that many vary from process to process

1. What is the difference between a long-term and a short-term scheduler? **(Ch3 Slides: 13)**

A long-term scheduler selects which processes should be brought into the ready queue.

A short-term scheduler selects which process should be executed next and allocates CPU.

1. What is the difference between an I/O-bound process and a CPU-bound process? **(Ch3 Slides: 15)**

I/O-bound process spends more time dealing with I/O than computations. This leads to many short CPU bursts.

A CPU-bound process spends more time doing computations. This leads to few but very long CPU bursts.

1. What is context switching? **(Ch3 Slides: 16)**

Context switching occurs when a CPU switches to another process. The system must make sure to save the state of the old process and load the saved state of the new process. Also, according to the slides, context-switch time is overhead. This means the system does no useful work while switching.

1. What are cooperating processes? **(Ch3 Slides: 26)**

According to the slides, processes can be either independent or cooperating. Cooperating processes can affect or be affected by other processes. This can be a good thing and leads to things like sharing data, file sharing, computation speedup, modularity, and convenience. To have cooperating processes, they need to have interprocess communication.

1. What is the difference between synchronous and asynchronous communication in terms of message passing? **(Ch3 Slides: 40)**

According to the slides message passing can be either blocking or non-blocking. Blocking is considered to be synchronous, and non-blocking is considered to be asynchronous.

Synchronous –

Blocking send has the sender block until the message is received, and blocking receive has the receiver block until a message is available.

Asynchronous –

Non-blocking send has the sender send the message and continue, non-blocking receive has the receiver receive a valid message or null.

**Part 2: Java programming (4 points):**

In the three java files contained in the zip file “cs 3343 assignment 3 java code spring 2024.zip”, modify the code to send a string of “hello world” instead of a date. **(I used the slides for Interface from the Big Java book, but I didn’t really end up needing them).**

Channel.java

public interface Channel<E>

{

    public void send(E item);

    public E receive();

}

MessageQueue.java

import java.util.Vector;

public class MessageQueue<E> implements Channel<E>

{

    private Vector<E> queue;

    public MessageQueue() {

        queue = new Vector<E>();

    }

    public void send(E item) {

        queue.addElement(item);

    }

    public E receive() {

        if (queue.size() == 0)

            return null;

        else

            return (E) queue.remove(0);

    }

}

Test.java

public class Test

{

    public static void main(String[] args) {

        Channel<String> mailBox = new MessageQueue<String>();

        mailBox.send(new String("Hello World!"));

        String rightNow = mailBox.receive();

        System.out.println(rightNow);

    }

}

