CPSC 3333 Programming Assignment 1

Overview

It is important that you name the required classes and methods exactly the same (case included) as specified. Testing will fail if you do not.

This week we will simulate simple queues of Process Control Blocks (PCBs). The image below illustrates what your program will do.

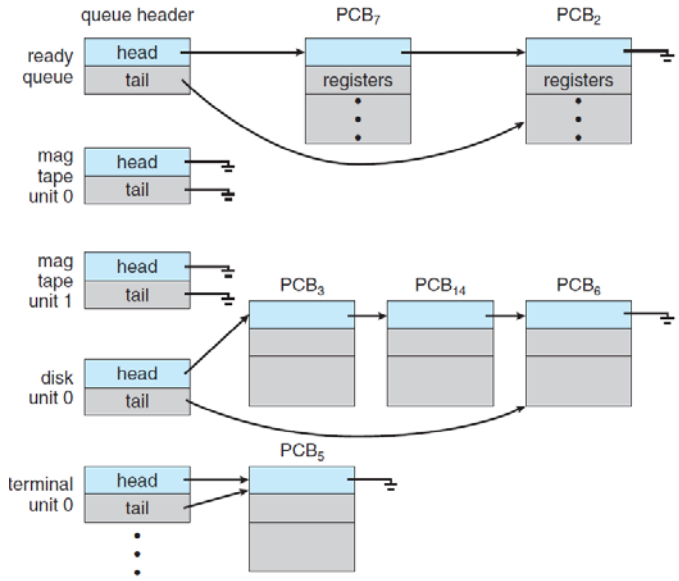

Figure 3.5 The ready queue and various I/O device queues.

Deliverables

You must submit the following files:

MasterQueue.java

Pcb.java

Specifications

For this assignment you can use any library provided by java (e.g. ArrayList, LinkedList, etc.) I will give you the freedom to select what you think works best. You may also have as many private variables and private methods as you see fit. You are only allowed to have the public methods specified below. You are not allowed to have any public variables.

- Pcb.java
  This will implement a simple PCB. It will hold the following attributes, Low memory bound (integer), High memory bound (integer), state (integer), process id (pid) (integer).

  The constructor will take the four parameters specified above.

  This class must have eight public methods:

  void setLowMem(int)

  this method will set the value of the low memory bound variable.

  Int getLowMem()

  This method will return the integer value of the low memory bound variable.

  void setHighMem(int)

  this method will set the value of the high memory bound variable.

  Int getHighMem()

  This method will return the integer value of the high memory bound variable.

  void setState(int)

  this method will set the value of the state variable.

  Int getState()

  This method will return the integer value of the state variable.

  void setPid(int)

  this method will set the value of the pid variable.

  Int getPid()

  This method will return the integer value of the pid variable.

- MasterQueue.java

  This will hold all the queues for the system. The above system has the “ready”, “mag tape unit 0”, “mag tape unit 1”, “disk”, and “terminal unit 0” queues listed. Your class must be flexible to hold any number of queues (e.g. I might request a single queue be created or I might request *n* number of queues.) These will all be a simple FIFO (first in first out) queues. Each queue must be flexible enough to hold between 0 and *n* Pcbs. When a Pcb is added, it is always added to the end of the queue and when a Pcb is removed, it is always removed from the front of the queue.

  Each queue must have a unique name (e.g. you cannot have two queues named “ready”.)

  The constructor will take no parameters.

  This class must have three public methods:

  boolean addQueue(String)

  This method will have a String parameter. It will return a boolean.

  It will add a new queue specified by the parameter. E.g. addQueue(“ready”) would create a new queue called “ready” to the list of queues. If there is already a queue by the specified name, then this method will return false. E.g. If you already have a queue named “ready” and you call addQueue(“ready”), it will return false. Otherwise, it will create the queue and return true.

  boolean addPcb(Pcb, String)

  This method will have two parameters, a Pcb (class created by you) and a String. It will return a boolean.

  It will add the specified Pcb to the specified queue. E.g. Assume we have already created a Pcb with pid = 100, the variable name we chose for this Pcb is ‘p’. Assume we also have created a queue named “ready”. Calling addPcb(p, “ready”) will add ‘p’ to the ready queue and return true.

  This method should also check for a valid queue. Assume we have only created one queue, “ready”. Calling addPcb(p, “notReady”) would return false. Nothing would be added to any queue, as the queue specified did not exist.

  Pcb removePcb(String)

  This method will have a String parameter. It will return a Pcb.

It will remove the first Pcb from the queue specified by the String parameter and return that Pcb.

If the specified queue does not exist, then this method will return a Pcb with pid = -1 (the values of the other fields do not matter as long as the pid = -1).

If the specified queue does not have any Pcbs (i.e. the queue is empty) then it will also return a Pcb with pid = -1. E.g. Assume we created our MasterQueue, and then we added the queue “ready”, but we have yet to add a Pcb to “ready”. Calling removePcb(“ready”) would return a Pcb will pid = -1.