AUBURN UNIVERSITY

CPSC 3333 Programming Assignment 4

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 write a simulation of the least recently used (LRU) algorithm

Deliverables

You must submit the following file:

LruC.java

LruS.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.

- LruC.java
  This will simulate the page table in a computer system. The program should be flexible enough to have from 1 to *n* entries in this table. Each page table entry will have two properties, the valid/invalid value, and the physical memory (RAM) address. Each physical address must be unique. You will use the **counting** method mentioned in the textbook / videos to track which page to replace.

  All pages will start as invalid.

  The constructor will take the 1 parameter, the number of pages in the table (integer).

  This class must have three public methods:

  boolean addPage(int)

  This will add a page to the page table. The parameter passed in will be the physical address of this block of memory. The method will return true if that block was already in the page table (i.e. the block is valid) and return false if the block was not in the page table (i.e. the block is invalid).

  boolean checkPage(int)

  This will check the valid bit of a page in the page table. The parameter passed in will be the physical address of this block of memory. The method will return true if that block was already in the page table (i.e. the block is valid) and return false if the block was not in the page table (i.e. the block is invalid).

  Dictionary<int,boolean> getPages()

  This will return a Dictionary containing all the pages, and their current valid/invalid values. The *int* is the unique memory address, the *boolean* is the valid bit. *True* is **valid** and *False* is **invalid**.

- LruS.java
  This will simulate the page table in a computer system. The program should be flexible enough to have from 1 to *n* entries in this table. Each page table entry will have two properties, the valid/invalid value, and the physical memory (RAM) address. Each physical address must be unique. You will use the **stack** method mentioned in the textbook / videos to track which page to replace.

  All pages will start as invalid.

  The constructor will take the 1 parameter, the number of pages in the table (integer).

  This class must have three public methods:

  boolean addPage(int)

  This will add a page to the page table. The parameter passed in will be the physical address of this block of memory. The method will return true if that block was already in the page table (i.e. the block is valid) and return false if the block was not in the page table (i.e. the block is invalid).

  boolean checkPage(int)

  This will check the valid bit of a page in the page table. The parameter passed in will be the physical address of this block of memory. The method will return true if that block was already in the page table (i.e. the block is valid) and return false if the block was not in the page table (i.e. the block is invalid).

  Dictionary<int,boolean>

  This will return a Dictionary containing all the pages, and their current valid/invalid values. The *int* is the unique memory address, the *boolean* is the valid bit. *True* is **valid** and *False* is **invalid**.