SAINT MARY'S UNIVERSITY SINCE 1802 | FACULTY OF SCIENCE

CSCI 3431.1 Fall, 2017 – Assignment 4

Operating Systems

Due Date: Wednesday, November 29, 2017
Closing Date: Friday, December 1, 2017

Deadline Policy

There will be a due date and a closing date for each assignment

- If you submit your work after the due date (but before the closing date), a *late submission* is applied to your overall assignments submission.
- You are allowed to use a *late submission* for up to 2 assignments throughout the semester without being penalized. After that, your assignment grade will be penalized by 5%.
- You will not be able to submit or receive credit for your work after the closing date.

Submission

Once you are done with your assignment, make sure you do the following before the assignment due date:

- Prepare a report containing the answers to the assignment questions.
- Fill out your Name and A# as indicated in the last page and attach this page to your report.
- Create one PDF file for your report.
- Combine your source code and PDF report into one zip file.
- Name your file following this convention: CSCI3431-Assignment4-*StudentID*, where *StudentID* is your Banner ID number starting with A.
- Submit your PDF file via Brightspace at https://smu.brightspace.com before the deadline.
- Your programs will be compiled and tested on the CS Linux machine.
- Do not submit a program that either won't compile or won't run. Instead of debugging your code, the marker/I will assign **Zero** credit for your submission.

Assignment Description

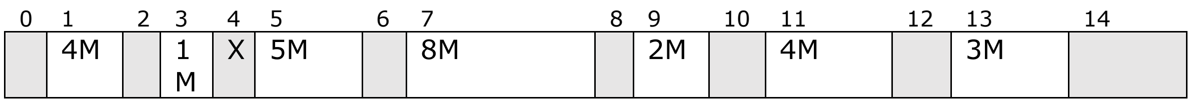
The assignment consists of two parts: Part I - practice exercises and Part II - programming/short paper part. Part I is mandatory. You can choose between Option A (short paper) or Option B (programming exercise) for Part II.

Part I: Practice Exercises

1. **[6 points]** Consider the Intel address-translation scheme shown in Figure 8.22 in your textbook.
    a. Describe all the steps taken by the Intel Pentium in translating a logical address into a physical address.
    b. What are the advantages to the operating system of hardware that provides such complicated memory translation?
    c. Are there any disadvantages to this address-translation system? If so, what are they? If not, why is this scheme not used by every manufacturer?

2. **[4 points]** can you think of any situations where supporting virtual memory would be a bad idea, and what would be gained by not having to support virtual memory? Explain.

3. **[4 points]** This diagram shows an example of memory configuration under dynamic partitioning, after a number of placement and swapping-out operations have been carried out. Addresses go from left to right; gray areas indicate blocks occupied by processes; white areas indicate free memory blocks. The last process placed is 2 Mbytes and is marked with an X. Only one process was swapped out after that. A new 3-Mbyte allocation request must be satisfied next. Indicate the intervals of memory where a partition will be created for the new process under the following four placement algorithms:
    a. Best-fit
    b. First-fit
    c. Next-fit
    d. Worst-fit



4. **[5 points]** Consider the following segment table:

Segment	Base	Length
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

What are the physical addresses for the following logical addresses? Indicate if either addresses is illegal.

a. 0,430
b. 1,10
c. 2,500
d. 3,400
e. 4,112

5. **[6 points]** We've discussed swapping as a technique used by systems to multi-task. Although mobile systems accommodate multi-tasking, they do not typically support swapping in any form.
   a. Find out why mobile systems do not support swapping.
   b. Choose one of the available mobile OS (either Apple's iOS or Android) and discuss the strategies they implement to allow multi-tasking.
   c. Knowing this restriction, how does this affect your next mobile application development?

Part II: OPTION A - Short Paper

One of the most important and complex tasks of an operating system is memory management. Memory management involves treating main memory as a resource to be allocated to and shared among a number of active processes. To use the processor and the input/output facilities efficiently, it is desirable to maintain as many processes in main memory as possible. In addition, it is desirable to free programmers from size restrictions in program development.

The basic tools of memory management are paging and segmentation. With paging each process is divided into relatively small, fixed-size pages. Segmentation provides for the use of pieces of varying size. It is also possible to combine segmentation and paging in a single memory management scheme.

Write a short paper (max 4 pages including references and diagrams) reviewing how the Windows operating systems handles memory management. You paper should illustrate the virtual memory manager, the virtual address map, paging, swapping, etc…

Part II: OPTION B - Programming Exercise

Write a Java program that implements the banker's algorithm discussed in Section 7.5.3. Several customers' request and release resources from the bank. The banker will grant a request only if it leaves the system in a safe state. A request is denied if it leaves the system in an unsafe state.

The Bank

The bank will consider requests from **n** customers for **m** resources. The bank will keep track of the resources using the following data structures:

```
private int n;              // the number of threads in the system
private int m;              // the number of resources
private int[] available;    // the amount available of each resource
private int[][] maximum;    // the maximum demand of each thread
private int[][] allocation;// the amount currently allocated to each thread
private int[][] need;       // the remaining needs of each thread
```

The functionality of the bank appears in the interface provided in `Bank.java`. The implementation of this interface will require adding a constructor that is passed the number of resources initially available. For example, suppose you have three resource types with 10,5,7 resources initially available. In this case, one can create an implementation of the interface using the following technique:

```
Bank theBank = new BankImpl(10,5,7); // create the bank
```

The bank will grant a request if the request satisfies the safely algorithm outline in Section 7.5.3.1. If granting the request does not leave the system in a safe state, the request is denied.

Testing

You can use the file `TestHarness.java`, which is available on the course website, to test your implementation of the Bank interface. The program expects the implementation of the Bank interface to be named `BankImpl` and requires an input file containing the maximum demand of each resource type for each customer. For example, if there are five customers and three resource types, the input file might include the following entry: 9,0,2 indicating that the maximum demand for this customer$_I$ (represented by the index of the entry in the file) is 9,0,2. Each line represents a separate customer, this means the `addCustomer()` method should be invoked as each line is read in, initializing the value of maximum for each customer. `Infile.txt` is provided as a sample input file.

`TestHarness.java` also requires the initial number of resources available in the bank. For example, if there are initially 10, 5, and 7 resources available, you need to invoke `TestHarness.java` as follows:

```
Java TestHarness infile.txt 10 5 7
```

`Infile.txt` refers to a file containing the maximum demand for each customer followed by the number of resources initially available. The available array will be initialized to the values passed on the command line. Below is a sample run:

```
Java TestHarness infile.txt 10 5 7

RQ 1 4 5 6
*4*
*5*
*6*

 Customer # 1 requesting 4 5 6 Available = 10  5  7  Approved
RL 2 5 7 8
*5*
*7*
*8*

 Customer # 2 releasing 5 7 8 Available = 11  7  9  Allocated =
[-5  -7  -8  ]
RQ 3 10 20 30
INSUFFICIENT RESOURCES
*10*
*20*
*30*

 Customer # 3 requesting 10 20 30 Available = 11  7  9  Denied
```

Self-evaluation

Please answer the following questions:

1. **[1 points]** Were you able to complete this assignment? What grade are you expecting? Please justify.

2. **[2 points]** Describe 2-3 challenges you faced while completing this assignment. How did you tackle those challenges?

3. **[2 points]** Provide a break down for the activities/milestones for this assignment. Give an estimate of hours spent on each activity. Try to be honest!

Date	Activities	Hours	Outcome

Hints and Suggestions

- ✓ Start early
- ✓ Document your work properly.
- ✓ Backup your work frequently. It's possible (and most likely) you go try a new feature and your program crashes!

Academic Integrity

You are required to demonstrate academic integrity in all of the work that you do. The University provides policies and procedures that every member of the university community is required to follow to ensure academic integrity. Unless stated otherwise, it is expected that all the work you submit for this course, is your OWN work.

Lack of knowledge of the academic integrity policy is not a reasonable explanation for a violation. You are encouraged to consult the Academic Integrity and Student Code of Conduct sections of the Academic Regulations in the Academic Calendar, in order to be well informed on the consequences of dishonest behavior. Please visit the links below for more information.

Evaluation Scheme

Student Name:	
Student ID:	
Total Points	**/125**
Part I Practice Exercises	**/25**
Part II	**/95**
Self-evaluation questions	**/5**

Question	1	2	3	4	5
Points	6	4	4	5	6
Score					

PartII – OPTION A Short Paper Evaluation Scheme	**/95**
Technical Correctness The topics discussed are technically correct	/10
Virtual Memory Manager Paper discusses the windows virtual memory manager	/20
Paging and Segmentation Paper addresses how windows implements paging and/or segmentation	/30
Swapping Paper addresses how windows handles swapping	/15
Organization and clarity Paper is well organized and clear about the various aspects of memory management. Proper use of diagrams to illustrate some idea.	/5
Language Paper is well written. Proper use of grammar.	/5
Length of paper Paper is within the length required	/5
Bibliography Appropriate use of references.	/5

PartII – OPTION B Programming Exercise	**/95**
Program runs properly • Program runs successfully • Program keeps displaying information regarding the state of the requests • No exceptions or blocked threads • Programs reads input from file	/15

Safety Algorithm • The safety algorithm is properly implemented	/20
Banker's Algorithm • Banker's algorithm uses the safety algorithm to grant/deny customers requests	/20
BankImpl • The Bank interface is properly implemented according to the provided interface	/20
Handling Errors and Exceptions Code deals with exceptions in an appropriate manner.	/5
Documentation Code shows good indentation, meaningful variable names, modularity, and helpful comments.	/5
Testing Readme file showing test cases performed along with screenshots.	/10