SFU SIMON FRASER UNIVERSITY
ENGAGING THE WORLD

CMPT 300: Operating Systems I

Assignment 3

Due July 26, 2018

POLICIES:

1. **Coverage**
   Chapters 7-9
2. **Grade**
   10 points, 100% counted into the final grade
3. **Individual or Group**
   Individual based, but group discussion is allowed and encouraged
4. **Academic Honesty**
   Violation of academic honesty may result in a penalty more severe than zero credit for an assignment, a test, and/or an exam.
5. **Submission**
   Electronic copy via CourSys
6. **Late Submission**
   2-point deduction for late submission within one week;
   5-point deduction for late submission over one week;
   Deduction ceases upon zero;
   Late submissions after the sample solution is available will NOT be graded.

QUESTIONS:

1. **2 points**
   Consider the following snapshot of a system:

	Allocation	Max
	ABCD	ABCD
P_0	3014	5117
P_1	2210	3211
P_2	3121	3321
P_3	0510	4512
P_4	4212	6325

Using the banker's algorithm, determine whether or not each of the following

states is unsafe. If the state is safe, illustrate the order in which the processes may complete.

a. Available = (0, 3, 0, 1)

b. Available = (1, 0, 0, 2)

[Grading Rubric: 1 point per state. If a safe sequence exists, BOTH the correct safe sequence AND the derivation steps are required.]

2. 2 points

Consider a system with a number *r* of resources of the same type. These resources are shared by a number *p* of processes. A process can request or release only one resource at a time. Prove that the system is deadlock free given the following two conditions:

a. The number of the maximum need of each process is in [1, *r]*;

b. The sum of all maximum needs is less than $r + p$.

[Grading Rubric: 2 points if a correct proof is provided. 0 point otherwise.]

3. 1 point

Why are page sizes always powers of 2?

Consider a logical address space of 64 pages of 1,024 words each, mapped onto a physical memory of 32 frames.

a. How many bits are there in the logical address?

b. How many bits are there in the physical address?

[Grading Rubric: 1 point if ALL three subquestions are correctly answered with necessary derivations. 0 point otherwise.]

4. 1 point

Given six memory partitions of 300 KB, 600 KB, 350 KB, 200 KB, 750 KB, and 125 KB (in order), how would the first-fit, best-fit, and worst-fit algorithms place processes of size 115 KB, 500 KB, 358 KB, 200 KB, and 375 KB (in order)?

[Grading Rubric: 1 point if ALL three placement decisions are correctly determined. 0 point otherwise.]

5. 1 point

Consider the two-dimensional array A:

```
int A[][] = new int[100][100];
```

where A[0][0] is at location 200 in a paged memory system with pages of size 200. A small process that manipulates the matrix resides in page 0 (locations 0 to 199). Thus, every instruction fetch will be from page 0.

For three page frames, how many page faults are generated by the following array-initialization loops? Use LRU replacement, and assume that page frame 1 contains the process and the other two are initially empty.

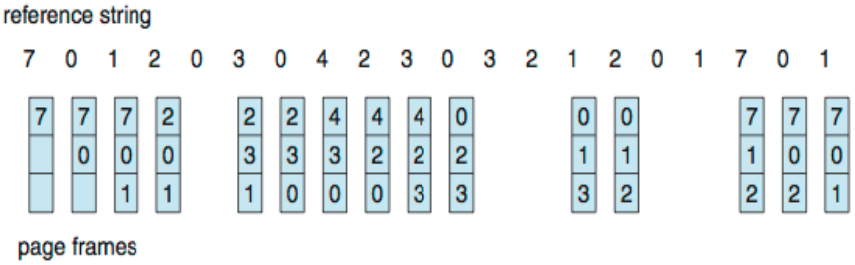
a.
```
for (int j = 0; j < 100; j++)
    for (int i = 0; i < 100; i++)
        A[i][j] = 0;
```

b.
```
for (int i = 0; i < 100; i++)
    for (int j = 0; j < 100; j++)
        A[i][j] = 0;
```

[Grading Rubric: 1 point if ALL two subquestions are correctly answered with necessary derivations. 0 point otherwise.]

6. **1 point**

Consider the following page reference string:
7, 2, 3, 1, 2, 5, 3, 4, 6, 7, 7, 1, 0, 5, 4, 6, 2, 3, 0, 1.
Assuming demand paging with three frames, how many page faults would occur for the following replacement algorithms?
a. LRU replacement
b. FIFO replacement
c. Optimal replacement
[Grading Rubric: 1 point if ALL three subquestions are correctly answered with illustrations similar as the following figure. 0 point otherwise.]



7. **2 points**

In a multilevel cache system, the CPU first sends the memory request to level 1 cache. If it is a cache hit, the data is transferred to the CPU. If it is a cache miss, the CPU will send the same memory request to level 2 cache. If it is still a cache miss there, the CPU will further send the memory request to lower level caches until a cache hit happens or memory access takes place.

A limitation of the preceding multilevel caching is that, even though a data block is cached in some lower level cache, the system still needs to endure all the time cost by the memory request going through all higher level caches with cache misses. Similarly, even if a data block is not cached, the CPU still sends memory request to one level of cache after another, taking likely a long time before accessing the memory.

Design a possible solution against the preceding limitation and show how it speeds up the average memory access time.
[Grading Rubric: Open question! Time to convince the TA.]