CSE 301: Operating Systems
Homework 2
(due Noon Sept 14)

Instructions

1. The deadline is a hard one. The form upload will close at sharp noon Sept 14. There will be no extensions.
2. You have to type the assignment using a word processing engine, create a pdf and upload on the form. Please note that only pdf files will be accepted.
3. Name the submission as {branch}_{roll_number}_{name}.pdf
4. All code/Jupyter notebooks must be put up as **secret gists** and linked in the created pdf. Again, only secret gists. Not the public ones.
5. Any instances of cheating/plagiarism will not be tolerated at all.
6. Cite all the pertinent references in IEEE format.
7. The least count of grading would be 0.5 marks.
8. Some suggestions for plotting - WolframAlpha, Academo.Org, Geogebra, Matploltib, GNUplot, Matlab, Octave
9. You can find the course VM on the course web page. The root password is: 1234

1. Create a Jupyter notebook to compare the efficacy of FIFO, Random, LRU and Optimal (Oracle) page replacement algorithms on the 90-10 workload: 90% of the references are to 10% of the pages and the remaining 10% of the references are to the remaining 90% of the pages. Make a plot showing hit rate v/s cache size for the different algorithms. **[3 marks]**

   Some hints/suggestions/assumptions:

   (a) Keep the workload size as 1000 (number of page accesses)

(b) You can use the instructor provided sample code for Random to get started

(c) For random, unlike other algorithms, your hit rate would depend on the random seed. Run the random page replacement for different random seeds and plot not only the mean but also the standard deviation of the hit rate

(d) Show a proof of the workload being 90-10

(e) You can vary the cache size as : [1, 2, 3, 5, 10, 15, 20, 25, 30, 35, 40...1000]

2. Consider a system with a 6 bit virtual address space, and 16 byte pages/frames. The mapping from virtual page numbers to physical frame numbers of a process is (0,9), (1,1), (2,6), and (3,8). Translate the following virtual addresses to physical addresses. Note that all addresses are in decimal. You may write your answer in decimal or binary. (a) 22 (b) 62 (c) 109 **[1.5 marks]**

3. Question 3 and 4 from OSTEP Chapter 13: Create a little program that uses a certain amount of memory, called memory-user.c. This program should take one command-line argument: the number of megabytes of memory it will use. When run, it should allocate an array, and constantly stream through the array, touching each entry. The program should do this indefinitely, or, perhaps, for a certain amount of time also specified at the command line. Now, while running your memory-user program, also (in a different terminal window, but on the same machine) run the free tool. How do the memory usage totals change when your program is running? How about when you kill the memory-user program? Do the numbers match your expectations? Try this for different amounts of memory usage. What happens when you use really large amounts of memory? **[1.5 marks]**

Not for grading

1. Write a program that takes following inputs: a) workload size (n), b) num unique pages (np), c) cache size 1 (c1) and returns the workload sequence which shows Belady's anomaly for c1 and c1 + 1. For example, given n = 12, np = 5, c1 = 3, we have: the sequence 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5 for which we have more hits for cache size, c1 = 3, compared to c1 + 1 =4.