Rubric
for
Homework #3

- Problems 1-6 worth 5 points
- Problems 7-8 worth 10 points

- Check along left side means earned all points for the problem
- Total minus points for a problem is along left side of page

Answers not labeled clearly: -1 overall (e.g. Problem 1 : R-1.12)
Low Readability: up to -3 per question
Lack of Clarity: up to -3 per question
Missed/wrong ideas or facts: -1 per idea/fact
Answer too long: up to -2 per question

If minus points not labeled, they are for missing ideas/facts.

Problem 1 : R-3.1

Multitasking: the illusion of programs running at the same time is done by time slicing. Each running process is given a tiny slice of time to do some work, and then it moves on to the next process. Because each time slice is so small and the context switching between running processes happens so fast, all the active processes appear to be running at the same time to us humans.
Problem 2: R-3.3

7 processes
Problem 3 : R-3.9

- solves problem of memory scarcity if enough processes are running
- allows processes to act as if their memory is contiguous when in reality it may be fragmented and spread across RAM
  - allows for indexing large arrays as contiguous chunks of memory
- allows total size of address spaces of executing processes to be larger than the actual main memory of the computer

Problem 4 : R-3.10

- highly customizable
- easy to use toolbar
- shows tree of processes
- show mini graphs showing the usage histories of CPU time, main memory and I/O
- colors make it easy to identify newly started processes, processes being terminated, users processes, and system processes
- in addition to monitoring performance, provides info about the process image
  - the executable program associated with the process
  - name of entity that developed the program
  - location on disk (path)
- can be used to verify a signature and display the name of the entity who signed the image.

Problem 5 : R-3.12

\[ 200,000 \times 2^{24} = 3,355,443,200,000 = 3.36 \times 10^{12} \]

Problem 6 : R-3.15

- reduces overall amount of computer hardware and materials
- reduces space requirements for situating the hardware
- uses less overall electrical power
- makes use of unused computer cycles.

Problem 7 : C-3.2

Note: there is some range of interpretation possible on this problem.

-1 for not having enough of the computation for both systems that leads to the conclusion(s).
-1 for not listing assumptions about how the picture system actually works.

The problem statement isn't absolutely clear that a user's favorite pictures will be shown only once during a login attempt (no repeats) although this is probably what the authors intended.
We are not told how many non-favorite photos there are to sample from for the login. The only thing we know is that there is probably at least 20 of these.

One way of looking at this problem: assuming no repeats during a single login attempt and the order of presenting the picture pairs is important. For example, \((p_1, p_2)\) then \((p_3, p_4)\) is different than \((p_3, p_4)\) then \((p_1, p_2)\).

Min search space size: \(2 \times (20!) = 4.86580402 \times 10^{18}\)

- If you were using a standard password system with, say, 20-digit long password using the 52 lower and uppercase alphabetic digits, the search space is

\[52^{20} = 2.08961787 \times 10^{34}\] (note: this doesn't even include shorter passwords)

- so the picture search space is pretty large (with my assumptions) but not as much as with the standard password system. With alternative picture assumptions, the space is even smaller.

Problem 8 : C-3.4

- 3 points per missing scenario (of up the three)

- if the scenarios seem to be duplicates, I take a one or two points off

- It seems to be a major point that if a system only has access permissions for owners of file, then many more usernames would need to be created. These extra usernames are actually group names. This would be ungainly system that requires a lot of extra effort.

Some possibilities:

- what about executable files, such as a word processor, that will potentially be executed by everyone on the system.
- group project necessitating group access to and creation of common data/executable files. For example: a group programming project
- want general public to be view a file but want to give them the minimum privilege possible. For example: public request for comments documents.