CMPS 111 Operating Systems
Winter 2018, Homework #1

Aaron Steele, atsteele@ucsc.edu

Question 1

- 1940s-1950s
  - The era prior to operating systems. Computers were programmed with assembly language directly, sometimes with plugboards.
- 1950s-1960s
  - The first operating system used for real work was the GM-NAA I/O, created by the General Motors' Research Division in 1956
  - Most other operating systems were also written by consumers at this point
- 1960s-1965
  - IBM wanted to make a single operating system for their OS/360 hardware, using the same instruction architecture.
  - This was the era of mainframe operating systems, when many different companies creates a whole slew of new operating systems that all worked totally differently.
- Late 1960s
  - The Unix operating system was created in AT&T's Bell Laboratories. Unix was programmed in C, which meant that when that language was ported to a new CPU architecture, the entire OS could also be ported. Because of this, it started gaining a large amount of popularity.
- 1970s - Microcomputers: small computers with 8-bit processors, like the Intel 8080, these were some of the first affordable computers on the market for hobbyists
  - Digital Research's CP/M-80 for the 8080/8085/Z-80 CPUs. Was mainly based on the PDP-11 architecture.
  - Microsoft's first operating system MSDOS/MIDAS was also developed on the PDP-11 architecture.
  - Apple DOS was released in 1978.

- 1980s
  - MS-DOS was released in 1981
  - Mac OS was released in 1984
  - Windows 1.0 was released in 1985
  - Version 8 of Unix was released in 1985
- These four operating systems started to dominate the market, especially in the late 1980s and into the 1990s, when they became the go-to operating systems for a majority of manufactured computers.

Question 2

There are a total of four processes created by running this code. First, P0 is the original process. P0 runs the first fork() and creates P1. P1 now runs the second fork() and creates P2. P2 runs exit(1) and quits.
After P0 has ran the first fork(), it runs the second and creates P3, which then runs exit(1) and quits.

So, the program creates four processes, but all of these processes might not be running at the same time, since it's easily possible that P0 might have exited by the time P2 is created, depending on the machine.

Question 3

A 32-bit unsigned integer has a maximum value of 4,294,967,295 ($2^{32} - 1$). Converting the max value of a 32-bit unsigned integer to a timestamp gives us: Sunday, February 7, 2106 6:28:15 AM. This would give us 88 years before the value overflows.

I would argue that is not enough time and that a 64-bit value should be used, which would give a much larger timeframe of 2,874 years.

Question 4

A web server that has a large amount of traffic has a number of problems, but the primary one, to my mind, is being able to serve the amount of data requested to each user from either the database or it's local fileserver. These are all read operations, which means that no locking is required and multiple threads can access the data without worrying about corruption or things of that nature.

Since the data can be accessed by multiple threads at once, it becomes fairly simple to load the data into shared memory and allow the threads to all read from it. This allows us to scale up the number of threads to meet the demand of traffic very easily, whereas using a single thread, or just a few threads, would be a much more challenging problem to serve the data to that many users.

This type of structure can be created using a thread pool with a circular bounded buffer. This also allows us to change the data that is being served to users, which we need to be able to do if we ever want to update the web server without fully taking the service down.

Question 5

Part A

Given $U = 1 - p^n$, we know U, and p, n is our unknown.

$$U = 0.99$$

$$p = 0.60$$

$$0.99 = 1 - 0.60^n$$

$$0.99 - 1 = 0.60^n$$

$$\log 0.01 = (\log 0.60)^n$$

$$\log 0.01 = n \log 0.60$$

$$\lfloor \frac{\log 0.01}{\log 0.60} \rfloor = n$$

$$9 = n$$

Therefore, there are 9 processes running at 99% CPU utilization. So, $16 - 9\text{MB} = 7\text{MB}$ of memory left free.

Part B

3MB per process. So, $\lfloor 16/3 \rfloor = 5$ maximum processes.

$$U = 1 - 0.60^5$$

$$U = 0.92$$

Therefore, the CPU has 92% utilization in this scenario.