Submit response?

Your username (**hermann@pdx.edu**) and responses will be recorded when you submit this form.

SWITCH ACCOUNT SUBMIT

Assignment: vmstat

Use this form to complete your assignment. Students may work alone or with a partner.
Do the homework set at the end of Chapter 22. Do steps 1 through 5. No need to do 6 or 7.
The steps are reworded and listed below.

This assignment introduces you to a new tool, vmstat, and how it can be used to understand memory, CPU, and I/O usage.

To do this activity you will need to use a Linux system such as linux.cs.pdx.edu so that you can run 'vmstat'. Mac and Windows systems, for example, do not support these tools, so you need to work on a linux system.

The name, username and photo associated with your Google account will be recorded when you upload files and submit this form. Not **hermann@pdx.edu**? Switch account

* Required

Your Name *

Hermann Yepdjio

Partner's Name (if you worked with a partner)

N.A

Find the mem.c program in the vm-beyondphys sub-directory within the OSTEP homework directories. Carefully read the README.md file in that directory. This is the same mem.c program that you used previously *

- [x] yes, I read the README.md, built the mem program and briefly tested it

Login to your linux machine (e.g., linux.cs.pdx.edu) and run 'man vmstat' to get the manual page for vmstat. Read the manual page carefully to understand what vmstat does and how to use it. *

- [x] yes, I read the man page for vmstat

Open two separate terminal connections to the same linux machine so that you can easily run programs in either window. Now, in one window, run vmstat -w 1, so that it shows statistics about machine usage every second. Widen your terminal window so that each row of stats is shown on a single row of the terminal window. Leave the first window running vmstat for the rest of the exercises below. Next, run the mem program with very little memory usage. For example, run './mem 1' (which uses only 1 MB of memory). How do the CPU usage statistics (shown by vmstat) change when running mem? Do the numbers in the user time column make sense? How does this change when running more than one instance of mem at once? *

When running mem.c with 1MB, the user time increases from 2 to about 15 and the idle time decreases from about 95 to about 81.
The more instances of mem.c I run the more the user time increases and the idle time decreases.

Next, look at some of vmstat's memory statistics while running mem. Focus on two columns: swpd (the amount of virtual memory used) and free (the amount of idle memory). Run './mem 1024' (which allocates 1 GB) and watch how these values change. Then stop the running program and watch again how the vmstat values change. What do you notice about the values? In particular, how does the 'free' column change when the program exits? Does the amount of free memory increase by the expected amount when mem exits? *

The swpd remains at 0 and never changes.
The free memory decreases by about 1GB when running mem.c with 1GB. It increases by about 1GB when mem exits

We'll next look at the swap columns (si and so), which indicate how much swapping is taking place to and from the disk. Of course, to activate these you'll need to run mem with large amounts of memory. First, examine how much free memory is on your Linux system (for example, by using the 'free' program). . Let's as-sume it's something like 30 GB of memory; if so, start by runningmem 12000 (about 12 GB) and watching the swap in/out columns. Do they ever give non-zero values? Then, try with larger values, up to 60% or 70% of free memory. What happens to these values as the program enters the second loop (and beyond), as compared to the first loop? How much data (total) are swapped in and out during the second, third, and subsequent loops? (do the numbers make sense?) *

I tried several number up to about 90% of free memory but si and so never changed and remained to 0; When getting closer to 90% of free memory, I get a Segmentation fault.

Do the same experiments as above, but now watch the other statistics (such as CPU utilization, and block I/O statistics). How do they change while mem is running? If yes, then how do they change? *

bi remains to 0
bo changes between 0 and about 5000 but the changes seems to not be related to whether I am running mem or not
The cpi usage increase from 1 or 2 to about 6 when running mem with a lot of space (60% of free memory). When using small amount of space it does not increase

Next, examine performance. Pick an input for mem that comfortably fits in memory (e.g., 40% of free memory).How long does loop 0 take (and subsequent loops 1, 2, etc.)? *

When using an input of 40% of free memory, loop 0 takes about 10,000 ms while the subsequent ones take about 3,000 ms.

Now run the mem program with a size that is 110% of available memory (i.e., a bit larger than available memory). How long do the loops take here? How do the bandwidth numbers compare? How different is performance when constantly swapping versus fitting everything comfortably in memory? *

I get a segmentation fault whenever I try anything bigger than 90% of free memory space

Create a graph with the size of memory used by mem on the x-axis, and the bandwidth of accessing said memory on the y-axis. For the X axis try values that are 10%, 20%, 30%, 40% ... 110% of available memory. Your choice: use Google Sheets for graphing, use your favorite graphing software, or make a hand-drawn graph. Upload the graph here. *

IMG_20201016_...

How does the performance of the first loop compare to that of subsequent loops, for both the case where everything fits in memory and where it doesn't. *

When everything fits in memory, the subsequent loops take less time than the 1st loop.
When everything does not fit in memory, I get a segmentation fault

Submit