CSC 369 Worksheet 6 Solution

August 21, 2020

1. Done. See link here

2. First, I need to find out how much memory is in my system, and how much is free. Running the command `free -m`, we have

```
ubuntu@primary:~$ free -m
              total        used        free      shared  buff/cache   available
Mem:            981         157         131           0         691         668
Swap:             0           0           0
```

   Using this information, I can write that the computer has

   - 981 MB of total memory
   - 131 MB of free memory

   Second, I need to answer if these numbers match my intuition.

   It does match my inution that free memory should be less than total memory.

   Notes

   - I should ask professor the type of intution the author was expecting
   - Installing Ubuntu Virtual Machine link: here
   - Start virtual machine by typing command: `multipass start ubuntu-lts`

3. I need to write a little program `question_3.c` (I will create this instead of `memory-user.c` for recording keeping purposes) that uses a certain amount of memory.

   It's critera are:

- The program should take one command-line argument: the number of megabytes of memory it will use.
- The program should allocate an array.
- The program should constantly stream through the array, touching each entry.
- The program should do this indefinitely, or for a certain amount of time also specified in command line.

For it's solution, please referr to `question_3.c`

Notes

- Learned that a large array (> 5000) can be generated using heap memory [1]
- **Command Line Arguements in C**

```
#include <stdio.h>

int main (int argc, char *argv[])
{

  return 0;
}
```

  - `argc`
    - Is the number of arguments passed to the program
  - `argv`
    - Is an array of strings
    - Each string represents one of the arguements passed to the program

  Example

For example, the command line

```
gcc -o myprog myprog.c
```

would result in the following values internal to GCC:

```
argc
     4
argv[0]
     gcc
argv[1]
     -o
argv[2]
     myprog
argv[3]
     myprog.c
```

- **String to integer**
  - **Syntax:** `int atoi(const char *string)`

    Example

    ```
    int output = atoi("20") /* Stores 20 in output*/
    ```

References

1) Stackoverflow, Allocating A Large (5000+) Array, link
2) Techdelight.com. Hpw tp fomd execution time of a C program, link
3) Stackoverflow, How do you clear the console screen in C, link

4. First, I need to revise program made in question 3 to also run the `free` tool.

   For it's solution, please referr to `question_4.c`.

   Second, I need to answer how the memory is changing when the programming is running.

   As program runs, the amount of memory available for the program decreases.

   Once it runs out, it returns "Resource temporarily unavailable" error.

   Third, I need to answer what happens to memory when program is killed.

   In the current solution, I am using heap memory, and I haven't set it up for the array to be freed on kill signal. So, the lost memory is not reclaimed once killed.

Fourth, I need to answer what happens when a large amount of memory is used.

The answer is the same as second part of the question. It tries to fill up, but once available memory runs out, it force terminates with "Resource unavailable" error.

```
Current index: 14
Time elapsed: 14 seconds
Amount of memory allocated: 10 MB
Size of array: 2500000
Press ctrl + z to terminate program
vm.swapusage: total = 1024.00M  used = 274.00M  free = 750.00M  (encrypted)
ERROR: Fork failed: Resource temporarily unavailable
moegu@MacBook-Pro-5 worksheet_6 % sysctl vm.swapusage
vm.swapusage: total = 1024.00M  used = 421.00M  free = 603.00M  (encrypted)
```

Notes

- Mac's equivalent of Ubuntu's `free -m` is `sysctl vm.swapusage` [1]

References

1) Stack Overflow, Is there a Mac OS X Terminal version of the "free" command in Linux systems? ,link

5. Please see link here

6.

```
moegu          5416   0.0  0.3  4365492  21036   ??  S     1:16am   0:00.43 /System/Library/Frameworks/Core
moegu          5414   0.0  0.2  8857488  20064   ??  S     1:16am   0:00.11 /Applications/Google Chrome.app
moegu          5413   0.0  0.8  9226640  64844   ??  S     1:16am   0:00.84 /Applications/Google Chrome.app
```

PID I will go for :)

Notes

- Alternative of pmap on Mac is `vmmap <PID>` [1]

References

1) Yong Sun's Blog, Tips: the equivalents of ldd(1) and pmap(1) on Mac OS X, link