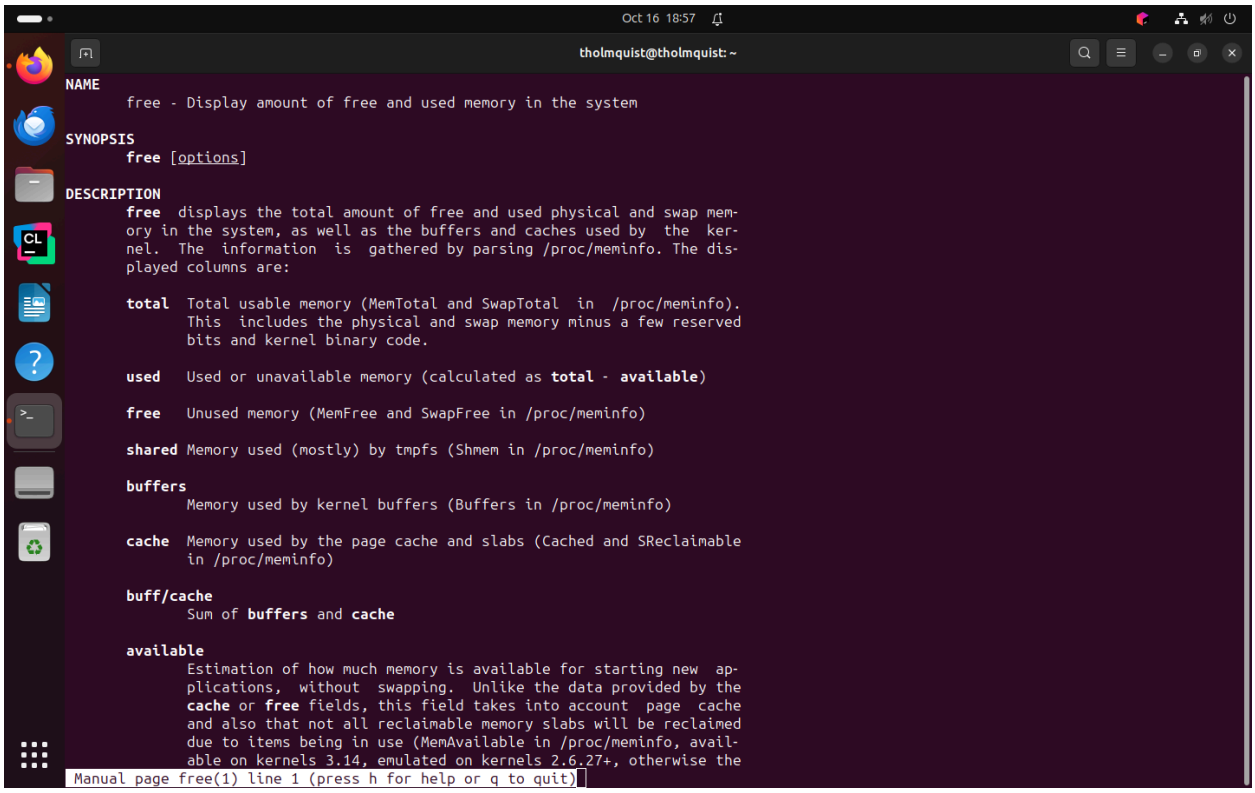
Ch.13 Coding Homework

Tyler Holmquist

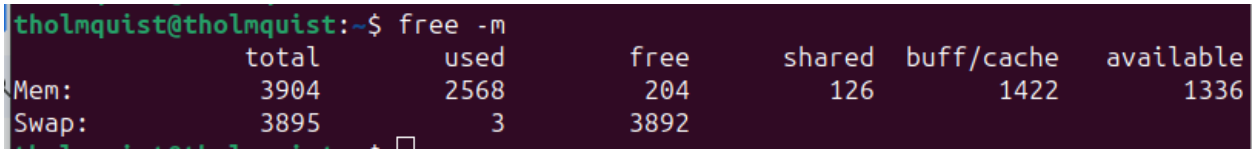
gcc (Ubuntu 13.2.0-4ubuntu3) 13.2.0

gcc -o mem_user memory-user.c

1.



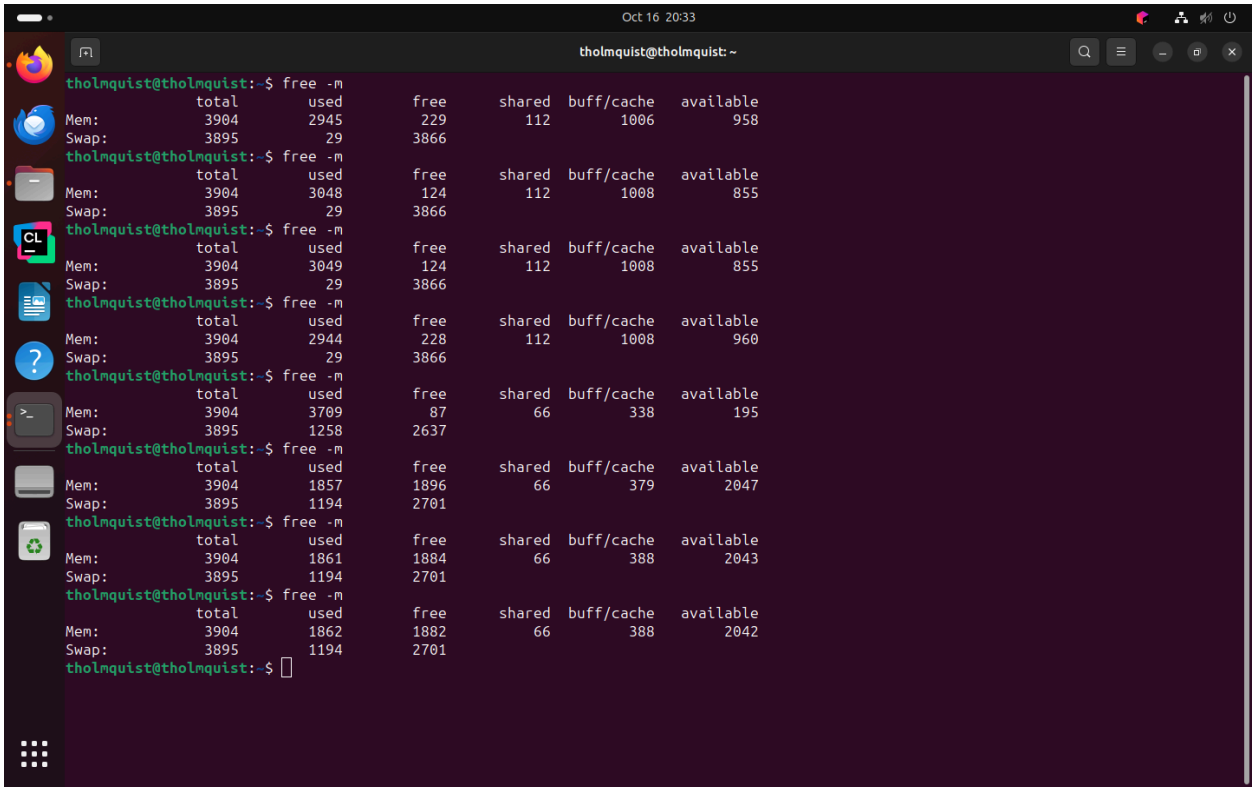
2.



I have 3904 megabytes in my system, 204 mb are free and I would have figured there would be more free according to my intuition.

3.

4.

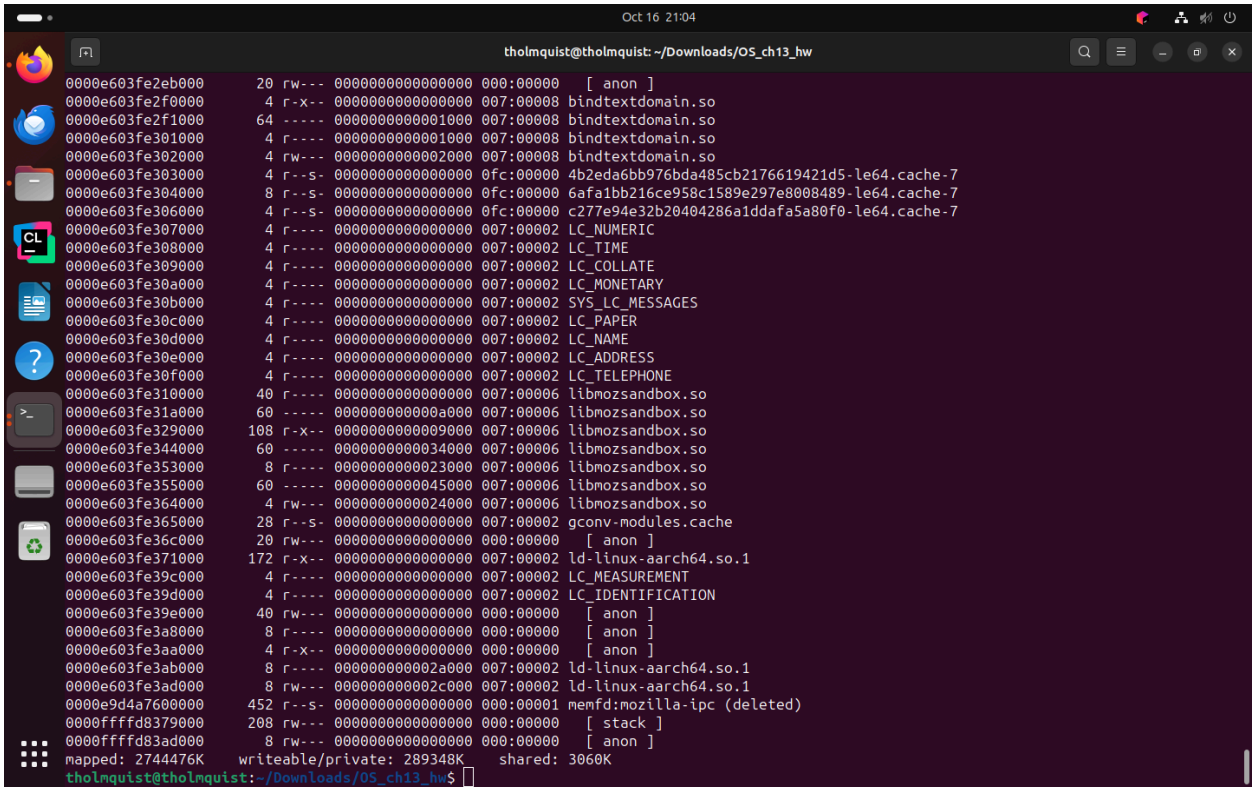


When running the program memory-user the amount of memory being used increases dramatically as the number input into the program increases and the amount of memory that is free decreases dramatically as the number gets larger. As you can see from the runs above the first run of free is when the program is not being run. The second and third are when the program is being run with an input of 100. Then the fifth is with an input of 10000 and as you can see the amount of free memory is all the way down to 87 mebibytes and the amount of memory being used is 3709 mebibytes which is much more than the 1857 being used after canceling the program.

5.

6.

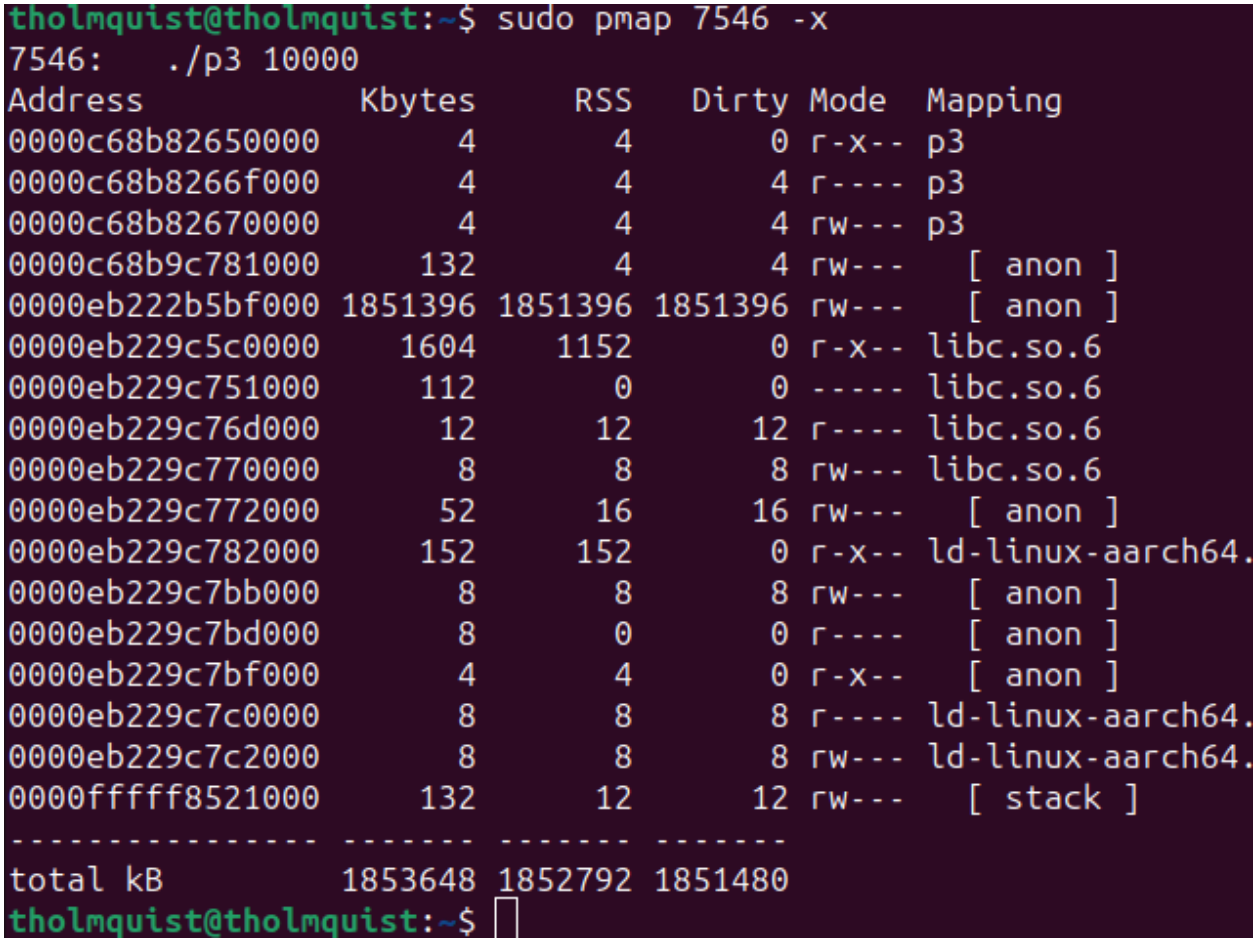
7.



Many different entities make up an address space on top of the code, heap and stack. We can also see here many anonymous pieces of memory with some library calls and other assortments of memory allocation outside of the three basic ones described.

8.

```
tholmquist@tholmquist:~$ sudo pmap 7523 -x
[sudo] password for tholmquist:
7523:   ./p3 100
Address           Kbytes     RSS   Dirty Mode  Mapping
0000aee51a5a0000       4       4       0 r-x-- p3
0000aee51a5bf000       4       4       4 r---- p3
0000aee51a5c0000       4       4       4 rw--- p3
0000aee53349a000     132       4       4 rw---   [ anon ]
0000ea80e9aaf000  102404  102404  102404 rw---   [ anon ]
0000ea80efeb0000    1604    1152       0 r-x-- libc.so.6
0000ea80f0041000     112       0       0 ----- libc.so.6
0000ea80f005d000      12      12      12 r---- libc.so.6
0000ea80f0060000       8       8       8 rw--- libc.so.6
0000ea80f0062000      52      16      16 rw---   [ anon ]
0000ea80f007f000     152     152       0 r-x-- ld-linux-aarch64.so.1
0000ea80f00b8000       8       8       8 rw---   [ anon ]
0000ea80f00ba000       8       0       0 r----   [ anon ]
0000ea80f00bc000       4       4       0 r-x--   [ anon ]
0000ea80f00bd000       8       8       8 r---- ld-linux-aarch64.so.1
0000ea80f00bf000       8       8       8 rw--- ld-linux-aarch64.so.1
0000ffffc5528000     132      12      12 rw---   [ stack ]
---------------- ------- ------- -------
total kB          104656  103800  102488
```



In the first screenshot I ran the program with an input of 100 mb and in the second screenshot an input of 10000 mb. As can be seen by the screenshots there is a block of memory being stored that is much larger in the second screenshot than the first while everything else remains showing the size of the heap increasing as more memory is requested for, which would match my expectations based off of what I have learned.