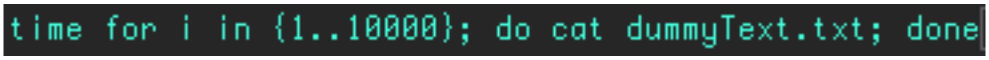
Humberto Atondo Martín del Campo
A00959979
February 10, 2019

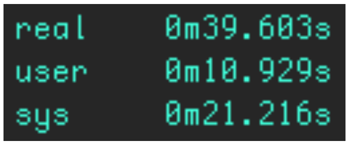
Lab1.1 Report

For this lab we were asked to build our own cat function and measure the performance against the already existing cat.c file. In contrast of the original cat.c file, we were asked to write our cat file using basic C functions like read, write, open and close.
To get the performance in our file we use the following command:



```
time for i in {1..10000}; do cat dummyText.txt; done
```

Where the for loop is where we establish come many times we are running our cat file and the time file will give us the time it took for all the processes to finish, we receive it in the following format:



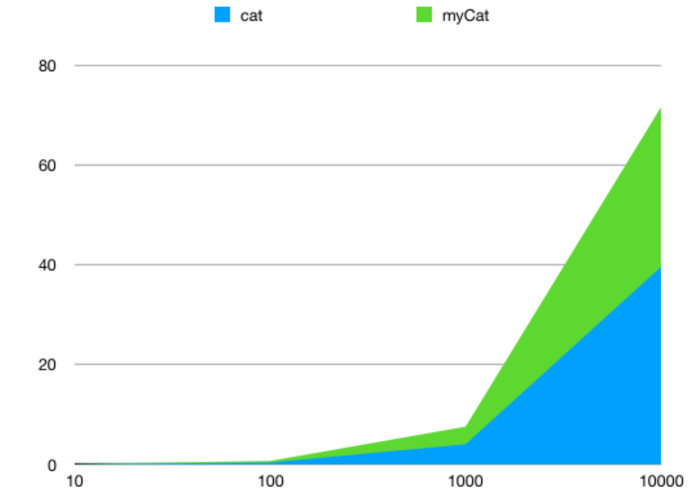
```
real    0m39.603s
user    0m10.929s
sys     0m21.216s
```

- **Real:** Is wall clock time - time from start to finish of the call. This is all elapsed time including time slices used by other processes and time the process spends blocked (for example if it is waiting for I/O to complete).
- **User:** Is the amount of CPU time spent in user-mode code (outside the kernel) *within* the process. This is only actual CPU time used in executing the process. Other processes and time the process spends blocked do not count towards this figure.
- **Sys:** Is the amount of CPU time spent in the kernel within the process. This means executing CPU time spent in system calls *within the kernel,* as opposed to library code, which is still running in user-space. Like 'user', this is only CPU time used by the process. See below for a brief description of kernel mode (also known as 'supervisor' mode) and the system call mechanism.

We will now proceed to the results where the time is given in seconds and our x value are the times we run the file:

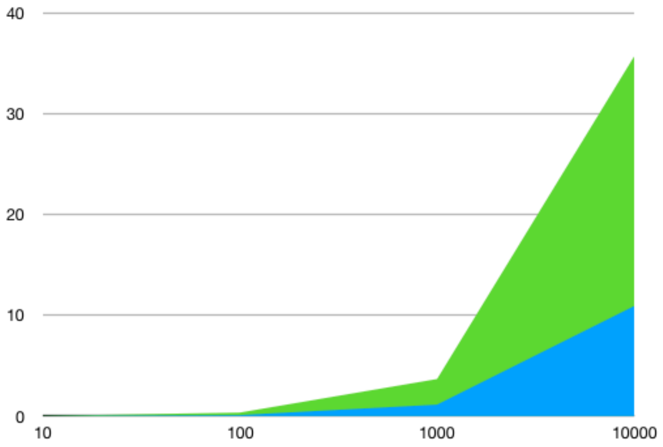
Real

iterations	cat	myCat
10	0.041	0.071
100	0.393	0.712
1000	4.063	7.585
10000	39.603	71.642



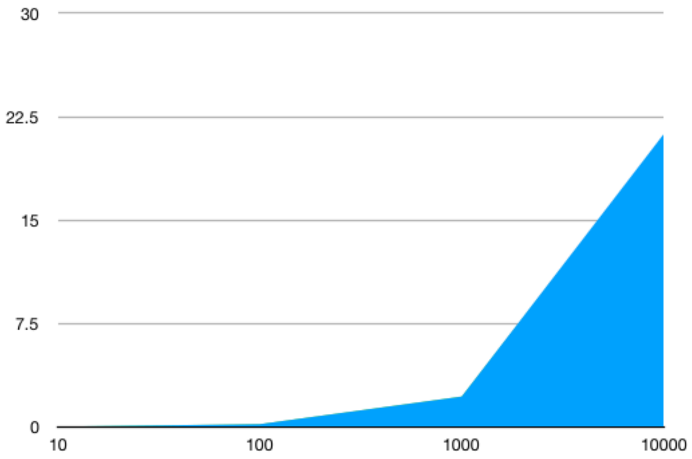
User

iterations	cat	myCat
10	0.011	0.035
100	0.110	0.355
1000	1.145	3.683
10000	10.929	35.690



Sys

iterations	cat	myCat
10	0.021	0.020
100	0.206	0.207
1000	2.208	2.211
10000	21.216	21.021



As we can observe from the results, the already implemented cat.c file is way faster in real and user time but almost the same in sys time. This is because the functions we used are way slower than the ones used for the original cat.c file.