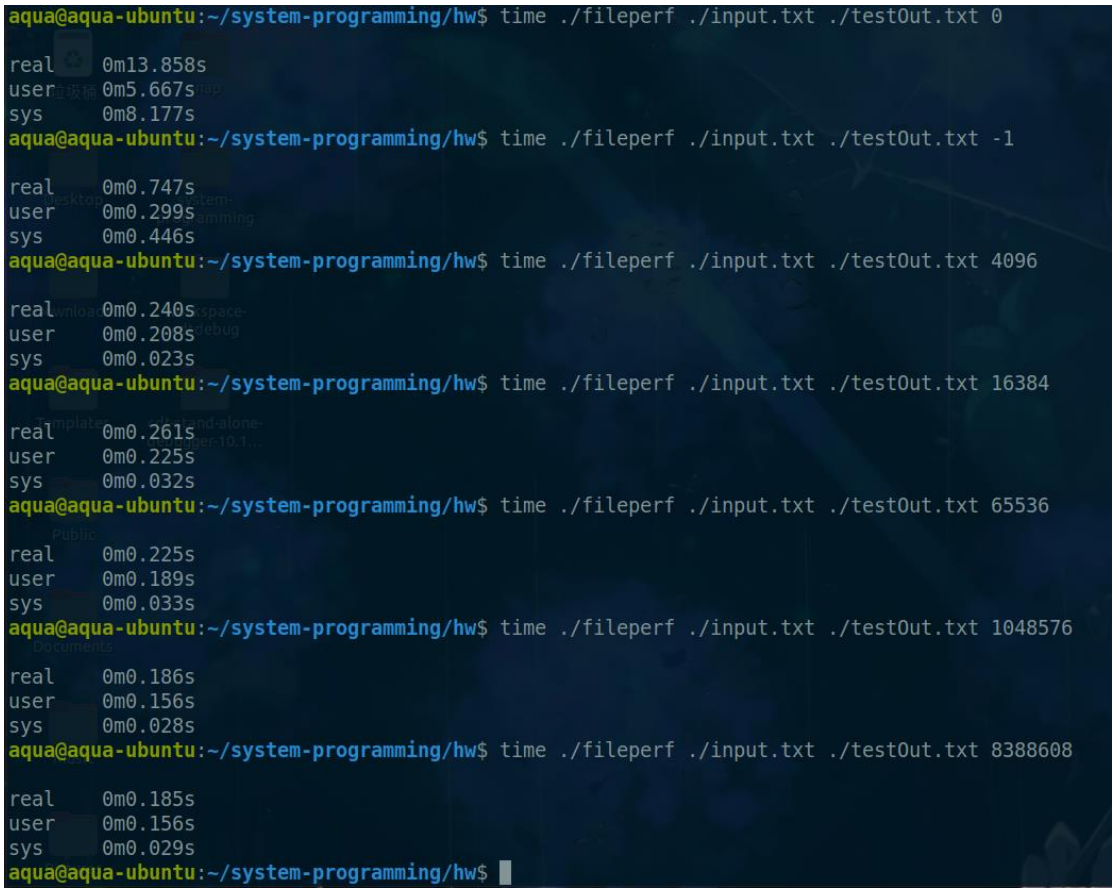
作業 4

408210005 謝宗哲

1. 分別使用不同的 buffersize 執行:



可以看出當 buffersize = 0 (unbuffered)的時候，花費時間最多

再來是 buffersize = -1 (linebuffered)，花費時間有明顯的減少

最後是 fully buffered，花費時間比上面兩者都少了許多，但是隨著 buffersize 成長，時間並沒有明顯降低的趨勢

2. 使用 ltrace 觀察呼叫函數庫的情況，分別測試 unbuffered, linebuffered, buffersize = 4K 的情況:

```
aqua@aqua-ubuntu:~/system-programming/hw$ ltrace -c ./fileperf ./sourceFile.txt
./testOut.txt 0
% time     seconds  usecs/call     calls      function
------ ----------- ----------- --------- --------------------
 36.87    0.004718        4718         1 fopen
 17.82    0.002281        2281         1 fputc
 11.38    0.001457        1457         1 fputs
  8.79    0.001125        1125         1 getc
  8.14    0.001042        1042         1 setvbuf
  7.20    0.000922         922         1 memset
  6.52    0.000834         834         1 __isoc99_sscanf
  3.27    0.000419         419         1 fprintf
------ ----------- ----------- --------- --------------------
100.00    0.012798                     8 total
aqua@aqua-ubuntu:~/system-programming/hw$ ltrace -c ./fileperf ./sourceFile.txt
./testOut.txt -1
% time     seconds  usecs/call     calls      function
------ ----------- ----------- --------- --------------------
 26.48    0.005239        5239         1 getc
 20.36    0.004029        4029         1 fopen
 16.10    0.003185        3185         1 fputs
 15.93    0.003151        3151         1 fputc
  7.64    0.001512        1512         1 __isoc99_sscanf
  5.12    0.001014        1014         1 fprintf
  4.63    0.000916         916         1 memset
  3.74    0.000740         740         1 setvbuf
------ ----------- ----------- --------- --------------------
100.00    0.019786                     8 total
```

```
aqua@aqua-ubuntu:~/system-programming/hw$ ltrace -c ./fileperf ./sourceFile.txt
./testOut.txt 4096
% time     seconds  usecs/call     calls      function
------ ----------- ----------- --------- --------------------
 27.68    0.000581         581         1 fopen
 11.82    0.000248         248         1 __isoc99_sscanf
 11.67    0.000245         245         1 setvbuf
 10.96    0.000230         230         1 getc
 10.34    0.000217         217         1 memset
  9.29    0.000195         195         1 fprintf
  9.15    0.000192         192         1 fputs
  9.10    0.000191         191         1 fputc
------ ----------- ----------- --------- --------------------
100.00    0.002099                     8 total
```

除了原本程式裡面呼叫的函數，還有 fputs, fputc 這兩個沒寫在程式裡的函數，可能是編譯器把 fprintf 轉成這兩個函數的。

但是我不太理解為什麼每個函數的呼叫次數只有 1 次，像是 getc 在迴圈裡面每次都會讀一個字元，呼叫次數應該會跟檔案長度一樣。

3. 使用 strace 觀察呼叫作業系統的情況，分別測試 unbuffered, linebuffered, buffersize = 4K 的情況:

```
aqua@aqua-ubuntu:~/system-programming/hw$ strace -c ./fileperf ./sourceFile.txt
./testOut.txt 0
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 64.00    0.006720           8       816           read
 14.49    0.001521          12       123           write
  5.14    0.000540          77         7           mmap
  4.91    0.000516          86         6           pread64
  3.65    0.000383          95         4           openat
  3.50    0.000368          92         4           mprotect
  1.35    0.000142          47         3           brk
  1.14    0.000120         120         1           munmap
  0.63    0.000066          33         2         1 arch_prctl
  0.60    0.000063          31         2           fstat
  0.58    0.000061          30         2           close
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           execve
------ ----------- ----------- --------- --------- ----------------
100.00    0.010500                   972         2 total
```

```
aqua@aqua-ubuntu:~/system-programming/hw$ strace -c ./fileperf ./sourceFile.txt
./testOut.txt -1
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 20.10    0.000635          90         7           mmap
 18.11    0.000572          47        12           write
 15.48    0.000489          81         6           pread64
 13.55    0.000428         107         4           mprotect
  8.29    0.000262          65         4           openat
  6.62    0.000209          52         4           fstat
  4.87    0.000154          51         3           brk
  4.15    0.000131          43         3           read
  3.96    0.000125         125         1           munmap
  2.53    0.000080          40         2         1 arch_prctl
  2.34    0.000074          37         2           close
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           execve
------ ----------- ----------- --------- --------- ----------------
100.00    0.003159                    50         2 total
```

```
aqua@aqua-ubuntu:~/system-programming/hw$ strace -c ./fileperf ./sourceFile.txt
./testOut.txt 4096
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 37.03    0.000267          66         4           openat
 20.67    0.000149          49         3           read
 18.72    0.000135          33         4           fstat
 14.98    0.000108         108         1           write
  8.60    0.000062          20         3           brk
  0.00    0.000000           0         2           close
  0.00    0.000000           0         7           mmap
  0.00    0.000000           0         4           mprotect
  0.00    0.000000           0         1           munmap
  0.00    0.000000           0         6           pread64
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         2         1 arch_prctl
------ ----------- ----------- --------- --------- ----------------
100.00    0.000721                    39         2 total
```

可以從上面三張圖觀察到 unbuffered 的時候 read 被呼叫 816 次, write 被呼叫 123 次。 linebuffered 的時候 read 被呼叫 3 次, write 被呼叫 12 次。 而 4K buffer 的時候 read 被呼叫 3 次, write 被呼叫 1 次。

所以有無 buffer 對呼叫作業系統的次數是有很大的影響的。

4. 從 2 跟 3 的測試中可以看出每個 function call 所花的時間比 system call 多,
但是 system call 被呼叫的次數也比 function call 還多。
總共花的時間是 system call 的部分比較少。