Exercise 24.1 bonnie++

bonnie++ is a widely available benchmarking program that tests and measures the performance of drives and filesystems. It is descended from **bonnie**, an earlier implementation.

Results can be read from the terminal window or directed to a file, and also to a **csv** format (**c**omma **s**eparated **v**alue). Companion programs, **bon_csv2html** and **bon_csv2txt**, can be used convert to html and plain text output formats.

We recommend you read the **man** page for **bonnie++** before using as it has quite a few options regarding which tests to perform and how exhaustive and stressful they should be. A quick synopsis is obtained with:

```
$ bonnie++ -help
 bonnie++: invalid option -- 'h'
usage:
bonnie++ [-d scratch-dir] [-c concurrency] [-s size(MiB)[:chunk-size(b)]]
      [-n number-to-stat[:max-size[:min-size][:num-directories[:chunk-size]]]]
      [-m machine-name] [-r ram-size-in-MiB]
      [-x number-of-tests] [-u uid-to-use:gid-to-use] [-g gid-to-use]
      [-q] [-f] [-b] [-p processes | -y] [-z seed | -Z random-file]
      [-D]

Version: 1.96
```

A quick test can be obtained with a command like:

```
$  time sudo bonnie++ -n 0 -u 0 -r 100 -f -b -d /mnt
```

where:

- `-n 0` means don't perform the file creation tests.
- `-u 0` means run as root.
- `-r 100` means pretend you have 100 MB of RAM.
- `-f` means skip per character I/O tests.
- `-b` means do a **fsync** after every write, which forces flushing to disk rather than just writing to cache.
- `-d /mnt` just specifies the directory to place the temporary file created; make sure it has enough space, in this case 300 MB, available.

If you don't supply a figure for your memory size, the program will figure out how much the system has and will create a testing file 2-3 times as large. We are not doing that here because it takes much longer to get a feel for things.

On an **RHEL 7** system:

```
$ time sudo bonnie++ -n 0 -u 0 -r 100 -f -b -d /mnt
Using uid:0, gid:0.
Writing intelligently...done
Rewriting...done
Reading intelligently...done
start 'em...done...done...done...done...done...
Version  1.96       ------Sequential Output------ --Sequential Input- --Random-
Concurrency   1     -Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine        Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP K/sec %CP  /sec %CP
q7             300M           99769  14 106000  12           +++++ +++ 257.3   1
Latency                         226us     237us               418us     624ms

1.96,1.96,q7,1,1415992158,300M,,,,99769,14,106000,12,,,+++++,+++,257.3,1,,,,,,,,,,,,,,,,,,226us,237us,,418us,624ms,,,
```

On an **Ubuntu 14.04** system, running as a virtual machine under hypervisor on the same physical machine:

```
$ time sudo bonnie++ -n 0 -u 0 -r 100 -f -b -d /mnt
Using uid:0, gid:0.
Writing intelligently...done
Rewriting...done
Reading intelligently...done
start 'em...done...done...done...done...done...
Version  1.97       ------Sequential Output------ --Sequential Input- --Random-
Concurrency   1     -Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine        Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP K/sec %CP  /sec %CP
ubuntu         300M           70000  61 43274  31           470061  96  2554  91
Latency                         306ms     201ms              9276us     770ms

1.97,1.97,ubuntu,1,1415983257,300M,,,,70000,61,43274,31,,,470061,96,2554,91,,,,,,,,,,,,,,,,,,306ms,201ms,,9276us,770ms,,,
```

You can clearly see the drop in performance.

Assuming you have saved the previous outputs as a file called `bonnie++.out`, you can convert the output to html:

```
$ bon_csv2html < bonnie++.out > bonnie++.html
```

or to plain text with:

```
$ bon_csv2txt < bonnie++.out > bonnie++.txt
```

After reading the documentation, try longer and larger, more ambitious tests. Try some of the tests we turned off. If your system is behaving well, save the results for future benchmarking comparisons when the system is sick.