Madolyne Dupraz
https://github.com/madolynedupraz/RES_Labo1

RES :

1) Conditions de l'expérience

Le code donné permet de comparer les temps de lecture et d'écriture en fonction de si on utilise un buffer ou non, ainsi que la taille des blocs utilisés.

2) Les mesures

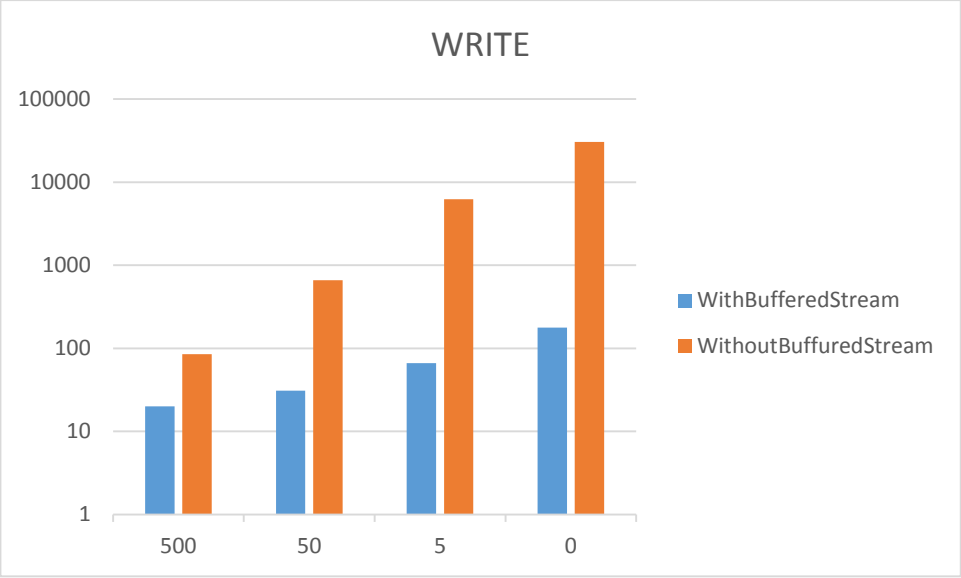
On obtient donc les temps en millisecondes de ses lectures/écritures en fonction des différents paramètres ; opération (write, read), avec/sans buffer, taille du bloc (500, 50, 5, 0).

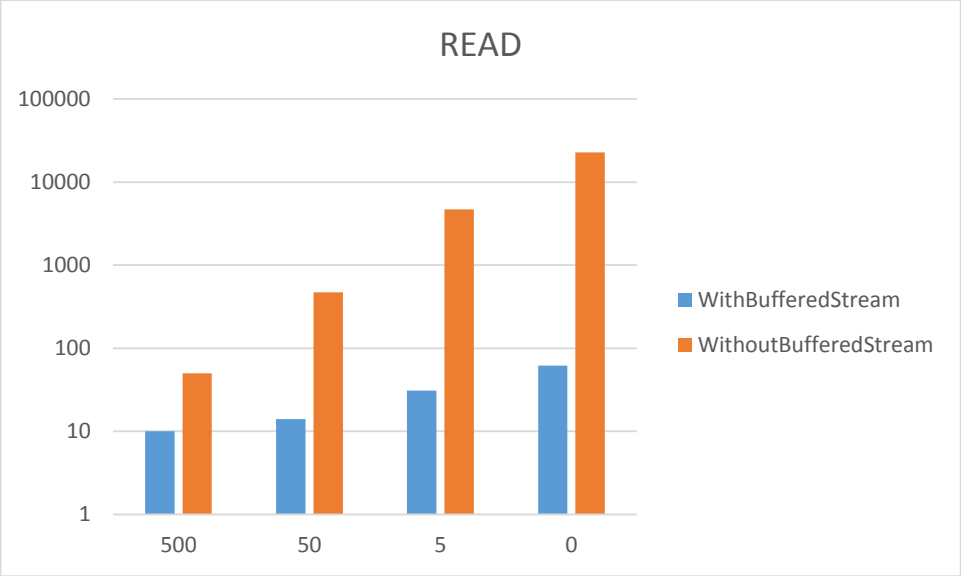
Voici les résultats obtenus :

operation	strategy	blockSize	fileSizeInBytes	durationInMs
WRITE	BlockByBlockWithBufferedStream	500	10485760	20
WRITE	BlockByBlockWithBufferedStream	50	10485760	31
WRITE	BlockByBlockWithBufferedStream	5	10485760	66
WRITE	ByteByByteWithBufferedStream	0	10485760	177
WRITE	BlockByBlockWithoutBufferedStream	500	10485760	85
WRITE	BlockByBlockWithoutBufferedStream	50	10485760	660
WRITE	BlockByBlockWithoutBufferedStream	5	10485760	6234
WRITE	ByteByByteWithoutBufferedStream	0	10485760	30583
READ	BlockByBlockWithBufferedStream	500	10485760	10
READ	BlockByBlockWithBufferedStream	50	10485760	14
READ	BlockByBlockWithBufferedStream	5	10485760	31
READ	ByteByByteWithBufferedStream	0	10485760	62
READ	BlockByBlockWithoutBufferedStream	500	10485760	50
READ	BlockByBlockWithoutBufferedStream	50	10485760	472
READ	BlockByBlockWithoutBufferedStream	5	10485760	4669
READ	ByteByByteWithoutBufferedStream	0	10485760	22680

3) Analyse des mesures

Voici les deux graphes effectués pour interpréter les mesures :





J'ai décidé de les séparer selon l'opération effectuée (écrire ou lire) et de mettre une échelle logarithmique afin d'y voir quelque chose.

On remarque sur ces deux graphes que c'est nettement plus rapide d'écrire ou de lire avec un flux bufferisé plutôt que sans.

Si l'on compare maintenant par taille de bloc, on constate que par bloc de 500 il nous faut 10ms en bufferisé et 50ms en non bufferisé ; ce qui fait 5 fois plus. Hors par bloc de 0 on obtient 62ms en bufferisé et 22680 en non bufferisé ; soit environ 365 fois plus. On remarque donc qu'il vaut mieux utiliser des blocs de 500 que des blocs de 0. Le fait qu'il n'y ai pas beaucoup de différence en bufferisé et non bufferisé avec des blocs de 500 est dû au faite que la taille d'un bloc du disque est de 512 octets.

4) Le code du programme

J'ai fait le choix de ne pas modifier le code donné, mais d'uniquement ajouter ce qui était nécessaire à la génération du « .csv ». J'ai donc créer un fichier à l'aide de ces lignes de codes :

```
static File fichier = new File ("test.csv");
static FileWriter ecrireFichier;
```

Et j'écris dans le fichier avec la méthode write(). Après avoir écrit dans mon fichier, j'utilise la méthode close() pour fermer le flux.