Time(sec) \Sort	Bubble	Insertion	Merge	Quick	Heap	Mmap
5	0.0000300000	0.0000150000	0.0000100000	0.000008000	0.000010000	0.000004000
10	0.0000130000	0.0000130000	0.0000210000	0.000011000	0.000019000	0.000007000
100	0.0006920000	0.0006850000	0.0006840000	0.000079000	0.000233000	0.000075000
1000	0.0668340000	0.0720210000	0.1124630000	0.001172000	0.002881000	0.000811000
10000	6.6470820000	6.9549920000	10.658056000	0.014625000	0.034252000	0.011048000
100000	NaN	NaN	NaN	0.174407000	0.447128000	0.121484000
1000000	NaN	NaN	NaN	2.107751000	5.725050000	1.732629000
10000000	NaN	NaN	NaN	24.37657800	84.75162300	27.92946400

В результате эксперимента, проводившегося на различных наборах строк длины 42, сгенерированных вспомогательным скриптом, были получены средние значения фактических времен работы алгоритмов.

Для определения времени работы алгоритма использовалась встроенная функция clock языка C.

Для вылавливания возможных утечек памяти применялась утилита: Valgrind. С флагом –leak-check=full.

Лог валгринда:

```
==30239==
==30239== HEAP SUMMARY:
==30239==     in use at exit: 0 bytes in 0 blocks
==30239==   total heap usage: 11 allocs, 11 frees, 11,640 bytes allocated
==30239==
==30239== All heap blocks were freed -- no leaks are possible
==30239==
==30239== For counts of detected and suppressed errors, rerun with: -v
==30239== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

Выводы

- **При малых входных данных различия между скоростями работы алгоритмов не существенны.**
- **При больших входных данных различия между скоростями работы алгоритмов весьма существенны.**
- **Алгоритмы, не использующие дополнительную память, эффективнее по времени.**
- **Алгоритм Quick Sort показал себя наиболее эффективным.**
- **Реализация с прямым отображением на физическую память(Mmap) позволяет получить значительный прирост в скорости. При условии, наличия достаточного объема памяти для отображения.**