Calculation of cache misses

Variant 0:

n: The number of colums and number of rows in the matrix. (n x n matrix)
ecl: Elements per cache line
entry: An element of the matrix
cl: A single cache line

```
ecl = sizeof(entry) / sizeof(cl) = 32 / (64 * 8) = 16

Misses = (3/ecl) * n ^ 2 = (3/16) * n ^ 2
Accesses = 3 * n ^ 2
Misses in % = Misses / Accesses = ((3/ecl) * n ^ 2) / (3 * n ^ 2) = 1/ecl
= 1/16 = 6.6%
```

Variant 1:

Under the Assumption that a complete Row of the Matrix cannot be stored in Cache. Otherwise on the next outer Loop Iteration the Cache would be able to return the first Column.

```
ecl = sizeof(element) / sizeof(CacheLine) = 32 / (64 * 8) = 16

Misses   = 3 * n ^ 2
Accesses = 3 * n ^ 2
Misses in % = Misses / Accesses = (3 * n ^ 2) / (3 * n ^ 2) = 100%
```

Findings:

When we go through the matrix in row major order we get a missrate of 3.3% on write operations and 0.7% on read operations. If we iterate with column major through the matrixes we notice a cache miss rate of 99.9% on writes but only 14% in reads in the level 1 cache on cachegrind. Cachegrind provides us with a simulated enviroment and perf reports the actual hardware performance. Therefore it could result in the difference to cachegrind. Here we found a cache miss rate of 0.231% for row major and 1.909% for column major. We assume there is a special prediction in the caches and therefore we get a lot less misses as we expcted.