Homework #3

(5 points) Implement the `flat_hash_set` hash table discussed in the lecture (see also this video: `https://www.youtube.com/watch?v=ncHmEUmJZf4`). For this assignment, it is sufficient to implement a SIMD-accelerated `find` function and any functional implementation of `insert`.

The elements will be random 64-bit integers, you may use the identity function as the hash. You can assume the table size is fixed and known in advance – dynamic resizing is not required.

Experiment:
Insert N random values into a table of size M. Then, measure the *average search time* for two cases:

1. Successful lookups – elements present in the table,
2. Unsuccessful lookups – elements not present in the table.

anc compare the performance of your implementation to `std::unordered_set`. Measure how the efficiency of your implementation depends on the *load factor* N/M (50—95%)?

Hints and notes:

- For SSE2 instructions (e.g., `_mm_cmpeq_epi8`), include the header `<emmintrin.h>` (see `https://intel.ly/2nTEFye`).
- To find the position of the least significant set bit, use GCC built-ins: `__builtin_ctz` or `__builtin_ffs` (see `https://bit.ly/32oX19x`); the last set bit can be cleared using Kernighan's trick: `x &= x - 1;` (see `https://stanford.io/33JeVnl`, `https://stanford.io/2VUASNU`).
- Test your program properly (e.g. against `std::unordered_set`) – correctness is more important than raw performance!
- For easy generation of unsuccessful lookups, you can, for example, insert only odd numbers and then search for even ones.
- Perform many iterations of `find`, but report *average time per operation.* Always specify the units of measurement.
- To obtain a machine-independent metric, you may also report the average probe sequence length for various load factors.
- For measurements, I recommend using the google benchmarks library `https://github.com/google/benchmark`. I also recommend reading the sections about reducing variance (`https://tinyurl.com/3jp2acrz`) and preventing compiler from optimizing all your code away (a.k.a. how to not shoot yourself in your foot `https://tinyurl.com/ypskdms4`).