INFOMOV 2017 THEORY CHECKUP

1. A modern CPU uses a pipeline to process a sequence of instructions, inspired by the ‘fetch-decode-execute-write back’ sequence.

   a) Why does a typical modern CPU have far more stages than just the original four?
   b) Name a disadvantage of having many stages in the pipeline of a CPU.
   c) What is, in the context of the CPU instruction pipeline, a 'bubble'?
   d) How is a 'superscalar' pipeline related to instruction level parallelism?
   e) How does a compiler help a superscalar processor to run at maximum efficiency?
   f) How can the programmer improve the efficiency of a superscalar processor?
   g) How does a CPU handle conditional branching instructions, where program flow could continue at two different program lines?

2. Your code contains the following snippet:

   ```
   float a = table[20];
   a += b; a += c;
   a += d; a += e;
   a += f; a += g;
   ```

   How would you modify this code to make it execute faster on a modern CPU?

3. "Going from 4-wide (SSE) to 8-wide (AVX) SIMD and beyond shows diminishing returns." Is this true or false? Explain your answer.

4. Explain the following concepts in 30 words or less.
   a) False sharing
   b) Prefetching
   c) Bélády's algorithm
   d) A prefetch thread
   e) Out-of-order execution
   f) Loop hoisting

5. Most reads and writes from C++ code are 4 or 8 bytes in size. Nevertheless, CPU caches typically use 64-byte cache lines. Why?

6. Which is better: AMD's 48-way set associative L3 cache, or Intel's 16-way L3 cache?

7. It is hard to measure the cost of a single instruction, even if we manage to prevent the compiler from optimizing it away.
   a) Define this 'cost'.
   b) What factors affect this cost?

8. NVidia's Pascal GPU architecture uses a cache line width of 128 bytes. Why is it, you think, that it uses a wider cache line than a CPU?