Aditya Dhulipala
EE/CSCI 451, Spring 2015
Homework 1 Submission

1.

1. **Superscalar processor** – A processor which has the ability to execute multiple instructions in the same clock cycle.
2. **Row major layout** – Data layout ordering wherein contiguous blocks of memory are loaded with contiguous rows of the data (matrix) (as opposed to column major layout where contiguous columns are stored)
3. **Cache pollution** – The situation wherein data fetched into the cache is unnecessary for the execution of the program, i.e. cache could've been used to load useful data to hide memory latency instead.
4. **Instruction level parallelism** – Parallelism obtained by executing sequence of low-level instructions simultaneously through the use of hardware & software techniques such as dynamic instruction issue (hardware) or compilers(software).
5. **Cache line** – Unit of data fetched from main memory to cache
6. **Data dependency** – The issue where the result of one particular instruction may be required to execute some subsequent instruction.
7. **Very Long Instruction Word (VLIW) Processor** – Processors that can execute a group of instruction packed into a single long instruction word. (Used to exploit instruction level parallelism by resolving dependencies through software techniques).
8. **Spatial locality** – A computation-centric view of memory access wherein the consecutive words in memory are used by successive instructions
9. **Single instruction multiple data** – A parallel computing architecture where one instruction is executed by all the processing elements on all the data (each processing element works on one piece of the data). This execution is performed in lockstep operation.
10. **Implicit parallelism** – Parallelism inherent in the program because of the nature of operations/computations being performed.

2. Prove by induction that the sum of the first n odd positive integers is n^2. Clearly label each step.

3. Suppose you owe someone v cents and must repay using only coins. You wish to use the fewest coins possible. If we're using United States currency and have an effectively unlimited supply of each coin type, you can achieve this by repaying with quarters until you owe less than 25 cents, then dimes until you owe less than ten cents, then a nickel if you owe at least five cents, and finally pennies to finish squaring the debt. A similar strategy also works if you include fifty-cent coins and dollar coins.
Suppose instead you were in a country that mints coins with the following values: one cent, 10 cents, 30 cents, and 40 cents. Does the strategy of repeatedly giving the most valuable coin whose worth is less than or equal to the debt minimize the number of coins given? Why or why not?

4. Suppose you have an array A of n integers. You wish to produce a two-dimensional array B that is $n \times n$, where $B[i][j]$ (for $i < j$) holds the sum of elements $A[i..j]$ (inclusive). $B[i][j]$, for $i \geq j$, is undefined - you may use those spots in B however you want (or leave garbage/defaults there).

(a) Write the pseudo-code for an algorithm that takes A as input and produces array B in time $O(n^3)$

(b) Write the pseudo-code for an algorithm that takes A as input and produces array B in time $O(n^2)$. If you are certain of your answer for this, you may skip part (a) and instead write, for that part, "see (b)."