Homework Assignment #1
Due: May 30, 2018 at 10pm
Please typeset the assignment in LaTeX, and upload the pdf file on Brightspace (D2L, online.mun.ca).

[10] 1. **Asymptotic notation**
Let f, g be functions on natural numbers. Recall the definitions of $f(n) \in O(g(n))$ $(\exists n_0, c > 0 \forall n > n_0 f(n) \leq c \cdot g(n))$ and a symmetric definition of $\Omega(g(n))$ (same except $f(n) \geq c \cdot g(n)$). We say $f(n) \in \Theta(g(n))$ if both $f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n).)$ Consider the following 12 functions (here, all logs are base two): $3n^2$, $n \cdot \log^3 n$, $n \cdot \log n$, $2^{n/3}$, 2^n, $n^2 - \log n$, $\log(n^{3n})$, $2^{n^{0.2}}$, $\sqrt{n^4}$, $n \cdot \log(n^3)$, $\frac{1}{100}2^n$, $\log(2^{10n \cdot n})$.

(a) List all groups of functions above that have the same asymptotic complexity (that is, they are in Θ of each other).

(b) Arrange these functions in the order of increasing asymptotic complexity (that is, functions earlier in your sequence should be $\in O()$ of later functions).

[25] 2. **Finite automata and Turing machines** We talked in class about encoding all kinds of possible inputs (graphs, Turing machines, sequences of numbers) as binary strings. In this problem, you will play with one possible encoding of pairs of binary strings. This is not the most efficient encoding, but it is simple to describe. The idea is to use even tape cell positions to hold the strings themselves, with odd cell positions indicating whether we are on a number or on a "delimiter". You can also think about it as treating each pair of cells as a symbol, with "00" and "01" corresponding to "0" and "1", respectively, and say "11" corresponding to a delimiter. More specifically, suppose the input is a pair of strings $x = x_1 \ldots x_n$ and $y = y_1 \ldots y_m$ over $0, 1$ (for example, $x = 010$ and $y = 01$). Then the pair $\langle x, y \rangle$ will be represented as $0x_1 0x_2 0 ... 0x_n 110y_1 0y_2 0 \ldots 0y_m$. For example, $\langle 010, 01 \rangle$ will be encoded as 000100110001.

(a) Give a deterministic finite automaton accepting a set of valid pairs of binary strings in the above encoding. Note that strings can be empty.

(b) Give a full description of a Turing machine which takes a pair of binary strings encoded as above, and accepts if and only if the two strings are equal. (First give a high-level description of how your Turing machine works, then give a formal description specifying Σ, Γ, Q with q_0, q_a, q_r and a table for δ).

(c) Give a description of a multi-tape Turing machine which starts with two numbers (possibly with leading 0s) encoded as a pair of binary strings as above, and outputs a single number (without any extra encodings) which is equal to the sum of the two input numbers. If the input is not valid, finish in q_r; otherwise finish in q_a pointing to the first symbol of the output with nothing other than the answer on the first tape (for this problem, ignore that there might be something on the other tape(s) when your TM stops).

[35] 3. **Computability of languages**.
For each of the following languages determine whether the language is decidable, semi-decidable, co-semi-decidable, or is higher in the arithmetic hierarchy. For the easiness proofs, give algorithms; for the hardness proofs, give reductions (ideally, from A_{TM}.) For each problem, start by writing the language descriptions using quantifiers.

(a) $A_{Composite} = \{< M >|$ the language of M contains only composite (non-prime) numbers $\geq 2\}$.

(b) $INF = \{< M >|$ the language of M is an infinite set $\}$.

(c) $Halt_{110} = \{< M >| M$ halts on *some string* containing a substring 110$\}$.

(d) $A_{10Gb} = \{< M >| M$ halts running on a mobile device with 10Gb of memory $\}$

[10] 4. **Closure of the class of semi-decidable languages under Kleene star.**
Suppose that L is semi-decidable. Define a closure L^* of L as $L^* = \{x_1 \dots x_k \mid k \in \mathbb{N}, \forall i, 1 \leq i \leq k,\ x_i \in L\}$ (that is, strings in L^* are finite sequences of strings from L). Show that L^* is also semi-decidable using the definition of semi-decidable in terms of existence of a verifier.