Homework 1

Carl Eastlund

Due **Wed., Jan. 16** at **9:00pm**.

1. You need to sign up for a Github account in order to submit homework in this class.
   (a) Make an account on `github.com`.
   (b) Send an email to the instructor at `cce@ccs.neu.edu` with the subject "CS4800 Github Account". The body of the email must contain your name and your Github account name. You will receive a response within 24 hours with the URL to a private Github account for your work in this course.
   (c) Your private repository will start with only some of the files you need for this assignment. The public repository at `github.com/neu-cs4800s13/public` has the files you need. Use `git merge`, `git rebase`, or some similar tool to import the revision history from the public repository into your private repository. You will *not* get full credit for just uploading new copies of each file into your repository.
   (d) Complete the other problems as instructed in your private repository. Whatever you submit to your private repository via `git push` by the due date will be graded.

2. Programming in this course will be done in Racket, and will use additional Racket libraries provided for the course. Write your solutions for this exercise in Racket. Put them in a file named `solution.rkt` in the same directory as this PDF.
   (a) Import the course software from `software/cs4800.rkt` into your solution using `require`.
   (b) Write a function `flatten-lists` to "flatten" a nested list structure into a single list. For example, the following expressions:

```
(flatten-lists (list))
(flatten-lists (list 2 3))
(flatten-lists (list 1 (list 2 3) 4))
(flatten-lists
  (list
    (list 1 (list 2 3) 4)
    (list 5 (list 6 7) 8)))
```

   ...should produce, respectively:

```
(list)
(list 2 3)
(list 1 2 3 4)
(list 1 2 3 4 5 6 7 8)
```

   You may not use any built-in or library functions other than `empty?`, `cons?`, `cons`, `first`, and `rest`, or their near-synonyms `null?`, `pair?`, `car`, and `cdr` (note that `cons` is always `cons`).
   (c) For full credit, `flatten-lists` must run in $O(n)$ time, where n is the combined length of the lists in its input.
   Use `define/cost` to define `flatten-lists` and any helper functions. Define a `cost-model` for the built-in functions that you use. Use the function `O?` we defined in `lectures/lecture-2013-01-10.rkt` to demonstrate that `flatten-lists` runs in $O(n)$ time.
   Note: The `O?` function cannot guarantee that your function runs in $O(n)$ time. Getting credit for using `O?` correctly does not guarantee credit for writing `flatten-lists` correctly, and vice versa.

(d) For extra credit, write `flatten-lists` so that every `cons` it creates is part of its final output. In other words, make sure it never copies a `cons` unnecessarily.

3. Prose and math in this course will be done in LaTeX. Write your solutions for this exercise in LaTeX. Put them in a file named `solution.tex` in the same directory as this PDF. Also submit a rendered PDF of your solution named `solution.pdf` in the same directory as this PDF.

   Sample formula: $2x_i^{2x_j^{2x_k^{\cdots}}}$

   (a) Demonstrate that $f(n) = 2n^2 - 3n + 4 \in O(n^2)$. That is, choose a specific $c > 0$ and a specific $n_0 > 0$. Then, show that for any possible $n > n_0$, it must be true that $f(n) \leq cn^2$.

   (b) Demonstrate that $f(n) = 3\sqrt{n} \in O(n)$. That is, choose a specific $c > 0$ and a specific $n_0 > 0$. Then, show that for any possible $n > n_0$, it must be true that $f(n) \leq cn$.

   (c) Demonstrate that $f(n) = \Sigma_{i=1}^{n}(2i^2 + 3i + 5) \in O(n^3)$. That is, choose a specific $c > 0$ and a specific $n_0 > 0$. Then, show that for any possible $n > n_0$, it must be true that $f(n) \leq cn^3$.

   (d) Demonstrate that $f(n) = \sqrt{n} \notin O(\log_2 n)$. That is, for any possible $c > 0$ and for any possible $n_0 > 0$, give a formula for some $n > n_0$ and show that n always satisfies $f(n) > c \log_2 n$.