CSE341 – Programming Languages (Fall 2020)
Assignment #0 – Introduction to Lisp Programming

Handed out: Tuesday October 13, 2020.

Due: 11:55pm Friday October 25, 2019

Hand-in Policy:

- Source codes must be uploaded to Moodle with a compressed format. File name must be studentID.zip
- No late submissions will be accepted.
- Your submission must contain a source file for each part and each file name should correspond to the related task. (Example: 181045999_part1.lisp, 181045999_part2.lisp…) etc

Collaboration Policy: No collaboration is permitted. Any cheating (copying someone else's work in any form) will result in a grade of -100 for the first offense and -200 for the subsequent attempts.

Grading: Each homework will be graded on the scale 100.

This assignment is about Common Lisp programming and a soft introduction. Our aim is to make you ready for the assignments to come and to make you gain familiarity with an unconventional (for you) programing language paradigm.

Part 1. You will write a program called **flattener**. Flattener will read a file called “nested_list.txt” and converts it into a single list without any sub-lists. Output will be written into a file called “flattened_list.txt”. **Expected code file name is 181045999_part1.lisp**

Part 2. You will write a program called **primecrawler**. This program will read two integers from a file called “boundries.txt”. Then you will implement a function that will find primes and semi-primes between the two integers (Both ends are included) (Semi-prime number is a number that have only two prime divisor). You will print the results into a file called, “primedistribution.txt”. For example: for given the input file that contains 2 and 10 we expect your output as follows:

```
2 is Prime
3 is Prime
4 is Semi-prime
5 is Prime
6 is  Semi-prime
7 is  Prime
9 is  Semi-prime
```

```
10 is Semi-prime
```

Expected code file name is 181045999_part2.lisp

Part 3. You will read at most 5 integers (File may contain fewer numbers) from a given file called "integer_inputs.txt". For each integer you will calculate the **collatz sequence** and print the results into a file called "collatz_outputs.txt" as follows. For example, given integers 6, 8 and 17 your program should have the output. You can find lots of reading material online about the collatz sequence.

```
6: 6 3 10 5 16 8 4 2 1
8: 8 4 2 1
17: 17 52 26 13 40 20 10 5 16 8 4 2 1
```

Expected code file name is 181045999_part3.lisp

Part 4. You will write a program that will calculate the Huffman codes for a given paragraph from a file called "paragraph.txt". We will supply you with a paragraph. And for each character (including whitespaces) you will construct the Huffman tree and determine the Huffman codes. For each character you will print the character and the codes into a file called "huffman_codes.txt". Note codes in the files should be ordered according to their length. (Shorter codes should appear at the top of the file)

Ex:

```
E: 0 0 0
B: 0 0 1 0
H: 0 1 0 1
  .
  .
  .
```

Expected code file name is 181045999_part4.lisp