CS115 **Assignment 08** Fall 2019

Due: Wednesday, November 20 @ 10:00 AM (No late submissions)

Assignment Guidelines:

- Solutions to these questions are expected to follow the requirements of the Style Guide (https://www.student.cs.uwaterloo.ca/~cs115/coursenotes1/styleguide.pdf) This includes all relevant design recipe elements, proper use of constants, and proper use of helper functions.
- Submission details:
  - Solutions to these questions must be placed in files **a08q1.rkt**, **a08q2.rkt**, **a08q3.rkt**, respectively, and must be completed in Racket.
  - All solutions must be submitted through MarkUs. Solutions will **not** be accepted through email.
  - Verify your basic test results using MarkUs to ensure that your files were submitted properly and are readable on MarkUs. *Note, however, that passing the basic tests does not guarantee that you will pass all our correctness tests*.
- Download the interface file from the course Web page to ensure that all function names are spelled correctly, and each function has the correct number and order of parameters.
- Restrictions:
  - You may only use the built-in functions and special forms introduced in the lecture slides up to and including the module covered by this assignment. A list of these functions can be found on the Assignments web page: https://www.student.cs.uwaterloo.ca/~cs115/#allowed
  - Read each question carefully to see if any additional restrictions apply.
  - Test data for correctness tests will always meet the stated assumptions for consumed values.
- The solutions you submit must be entirely your own work. Do not look up either full or partial solutions on the Internet or in printed sources.

Plagiarism: The following applies to all assignments in CS115.

All work in CS 115 is to be done individually. The penalty for plagiarism on assignments (first offense) is a mark of 0 on the affected question and 5 marks off the final grade, consistent with School of Computer Science policy. In addition, a letter detailing the offense is sent to the Associate Dean of Undergraduate Studies, meaning that subsequent offenses will carry more severe penalties, up to suspension or expulsion.

To avoid inadvertently incurring this penalty, you should discuss assignment issues with other students only in a very broad and high-level fashion. Do not take notes during such discussions, and avoid looking at anyone else's code, on screen or on paper. If you find yourself stuck, contact the ISA or instructor for help, instead of getting the solution from someone else. Do not consult other books, library materials, Internet sources, or solutions (yours or other people's) from other courses or other terms.

Read more course policies at: https://www.student.cs.uwaterloo.ca/~cs115/#policies

Language level: Intermediate Student with lambda
Coverage: Module 7

CS115 **Assignment 08** Fall 2019

Due: Wednesday, November 20 @ 10:00 AM (No late submissions)

General Guidelines

- For all questions on this assignment, you are not allowed to use explicit recursion. All of your solutions must use abstract list functions. There will be a severe penalty in correctness marks for a solution that uses explicit recursion.
- All helper functions must be defined as `local` functions or as `lambda` expressions.
- When writing your solutions, look for opportunities to use locally defined constants where appropriate.

Question 1: Secret Code

When sending coded messages, the individual characters are normally changed. In addition, cues from the text, like spaces, punctuation, and capitalization are often removed. Write a function, `encode`, that consumes a string, `message`, and produces an encoded version of that string according to the following rules:

- all non-alphanumeric characters are removed
- all numeric characters are unchanged
- all alphabetic characters appear as uppercase letters, where they are swapped with the letter at the same relative position at the other end of the alphabet. For example, an `A` would become a `Z` and a `Z` would become an `A`; a `B` would become a `Y` and a `Y` would become a `B`, etc.

Note: You may assume all alphabetic characters are English letters and all numeric characters are the digits from 0 – 9.

For example: `(encode "I love CS115!") => "ROLEVXH115"`

Submit your solution in the file **a08q1.rkt**.

Question 2: IATA Airport-Country Codes

Recall Question 2 from Assignment 06. For this question, you will write the function, `same-country`, that follows the same rules as the question on Assignment 06 with the following exceptions:

- You must follow the General Guidelines for this assignment (i.e. A08).
- You are allowed to use the built-in function `sort`
- The IATA code may or may not be in the association list. If it is not in the list, then your function should produce `empty`.
- You are not allowed to use any named helper functions. In other words, you must use `lambda`

For example, if you have defined the constant `alist` as described in A06,

```
(same-country alist "ABC") => empty
(same-country alist "YUL") => (list "YUL" "YVR" "YWG" "YYZ")
```

You may use the design recipe components from the model solutions of assignment 06 in your solution. You will not be marked for testing on this question. However, we recommend that you include tests to ensure your solution is correct.

Submit your solution in the file **a08q2.rkt**.

Due: Wednesday, November 20 @ 10:00 AM (No late submissions)

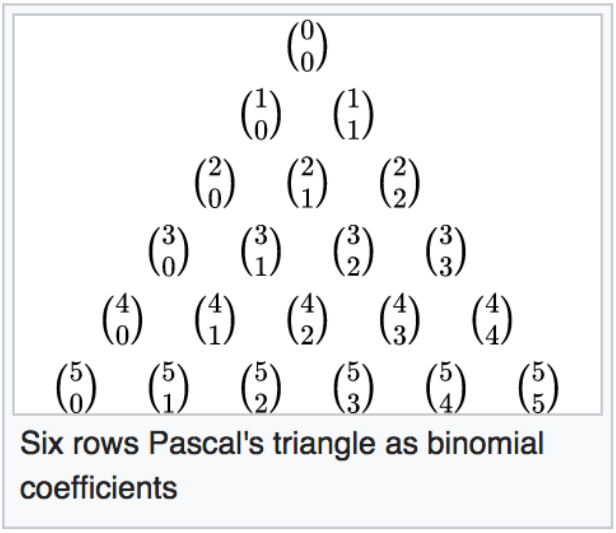
Question 3: Pascal's Triangle

Pascal's Triangle is a triangle formed by positive integers. The peak of the triangle is a 1. The next row of the triangle has two 1s. Each row below starts and ends with a 1, and each of the other entries in that row is formed by adding the numbers in the row above that appear to the left and right of the spot being filled. See the Wikipedia entry for more details https://en.wikipedia.org/wiki/Pascal's_triangle

If we consider the peak of the triangle row 0, and the leftmost entry in each row position 0, then each entry in the triangle can be calculated as $\binom{n}{k}$ which means "*n* choose *k*". In this case, *n* is the row number and *k* is the row position.

You can expand $\binom{n}{k}$ as $\frac{n!}{(n-k)!\,k!}$. Note that *n*! = *n* • (*n*-1) • (*n*-2) … • 3 • 2 • 1, and that 0! = 1.

Here is an example of how to calculate Pascal's triangle with six rows (source Wikipedia.org)



Six rows Pascal's triangle as binomial coefficients

Write the function, `pascals-triangle`, that consumes a natural number, `size`, and produces a list of lists. There will be `size` number of inner lists, each of which represent one row of Pascal's Triangle. If `size` is `0`, the function should produce the `empty` list. For example,

```
(pascals-triangle 4)
=> (list (list 1) (list 1 1) (list 1 2 1) (list 1 3 3 1))
```

Notes:

- Your solution should use `range` and/or `build-list`
- Your solution must **NOT** use `cond`

Submit your solution in the file **a08q3.rkt**.