Assignment Guidelines:

- For this and all subsequent assignments, you are expected to use the design recipe when writing functions from scratch, including helper functions.
- For full marks, it is not sufficient to have a correct program. Be sure to follow all the steps of the design recipe. Read the Style Guide carefully to ensure that you are following the proper conventions. In addition, your solution must include the definition of constants and helper functions where appropriate.
- Unless otherwise indicated in the question you may use only the built-in functions and special forms introduced in the lecture slides from CS115 up to and including the modules covered by this assignment. A list of functions described in each module of the lecture slides can be found at https://www.student.cs.uwaterloo.ca/~cs115/built_in
- 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.
- Read each question carefully for restrictions.
- Test data for all questions will always meet the stated assumptions for consumed values.
- Do not copy the purpose directly from the assignment description. The purpose should be written in your own words and include references to the parameter names of your functions.
- 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.
- Do not send any code files by email to your instructors or tutors. Course staff will not accept it as an assignment submission. Course staff will not debug code emailed to them.
- You may post general assignment questions using the discussion groups on Waterloo LEARN. Choose Connect Discussions. Read the guidelines for posting questions. Do NOT post any code as part of your questions.
- Check Markus and your basic test results to ensure that your files were properly submitted. In most cases, solutions that do not pass the basic tests will not receive any correctness marks.
- Read the course web page for more information on assignment policies and how to organize and submit your work. Follow the instructions in the Style Guide.
- Your solutions should be placed in files a8qY.rkt, where Y is a value from 1 to 3.

Plagiarism: Read https://www.student.cs.uwaterloo.ca/~cs115/assignments#Plagiarism
Language level: Intermediate Student with Lambda
Coverage: Module 08

You are not allowed to use recursion in this assignment. Your solutions must use abstract list functions. You may use lambda, but it is not required. All helper functions must be local definitions.

Question 1
(a.) The factorial of a natural number $n$ is denoted by $n!$ and it is formally defined as follows:

$$n! = \prod_{k=1}^{n} k$$
$$= 1 \cdot 2 \cdot 3 \cdots (n - 2) \cdot (n - 1) \cdot n$$
$$= n \cdot (n - 1) \cdot (n - 2) \cdots 2 \cdot 1$$

where $0! = 1$, and for natural numbers $n \geq 1$, $n! = n \cdot (n - 1)!$

Write a function `factorial` that consumes a natural number $n$ and produces the factorial of $n$.

(b.) Write a function `sum-b-to-n` that consumes two natural numbers $b$ and $n$, and produces the sum of all natural numbers between $b$ and $n$ inclusive. Note that $n \geq b$.

For example:
- `(sum-b-to-n 5 5) => 5`
- `(sum-b-to-n 5 10) => 45`, i.e., $5+6+7+8+9+10$

(c.) Write a function `select-sum` that consumes `alon`, a list of natural numbers, and $b$, another natural number, and produces a list of natural numbers created by applying the `sum-b-to-n` function described in (b) above to all values that are greater than or equal to $b$ in `alon`.

For example:
- `(select-sum (list 1 7 8) 5) => (list 18 26)`
- `(select-sum (list 10 0 3 14 5 6 17 9 10) 3)`
  => `(list 52 3 102 12 18 150 42 52)`
Question 2:
Rotokas is a language spoken by few people. It is spoken by about 4,320 people on the island of Bougainville, Papua New Guinea. The language uses a Latin-based alphabet that contains only 12 characters, which is the smallest character-set for a language in use today. The characters are \{A E G I K O P R S T U V\} in upper case, and \{a e g i k o p r s t u v\} in lower case. For this assignment, \#\space is an acceptable Rotokas character.

(a.) Write a function is-rotokas? that consumes a string s and produces true if all characters in s are members of the Rotokas alphabet, and false otherwise. An empty string is a member of the Rotokas alphabet, but non-alphabetic characters are not.

For example:
• (is-rotokas? "Perogies Tree") => true
• (is-rotokas? "Meat") => false

(b.) Write a function filter-and-sort-rotokas that consumes alos, a list of strings, and produces the list of all Rotokas words contained in alos, sorted in case-insensitive ascending order. If alos contains duplicate strings (which may or may not be case-sensitive), the strings should be arranged in the produced list in the same order they appear in alos.

For example:
• (filter-and-sort-rotokas empty) => empty
• (filter-and-sort-rotokas (list "tea" "team" "Meat" "Giros" "Veggies")) => (list "Giros" "tea" "Veggies")

You may find the built-in function string-ci<=? useful.

(c.) Write a function filter-and-deepsort-rotokas that behaves like filter-and-sort-rotokas in (b) above except that every string in the list would first be sorted in an ascending lexicographical order of their characters.

For example:
• (filter-and-deepsort-rotokas empty) => empty
• (filter-and-deepsort-rotokas (list "tea" "team" "Meat" "Giros" "Veggies")) =>
  (list "aet" "eeggisV" "Giors")

You may find the built-in function char-ci<=? useful.
Question 3

In statistics, histograms are used to represent the frequency distribution of data. They provide a count of the number of observations of variables. For example, an ornithologist may count the number of times they observe birds from each of several species while studying the population of an ecosystem. Often the range of observed values is "binned" into a set of intervals, e.g., to group closely related bird species together. The histogram then counts how many of the observed values fall into each bin/interval. This is also helpful for analyzing student grades, e.g., to answer questions like "how many students had a grade in the 90's?" In the past, the University of Waterloo used 10 letter grade bins to count percentage grades that range from 0 to 100. The table below shows the relationship between letter grades and percentage ranges of grades.

<table>
<thead>
<tr>
<th>Letter grade</th>
<th>Percentage ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>A+</td>
<td>90-100</td>
</tr>
<tr>
<td>A</td>
<td>85-89</td>
</tr>
<tr>
<td>A-</td>
<td>80-84</td>
</tr>
<tr>
<td>B+</td>
<td>77-79</td>
</tr>
<tr>
<td>B</td>
<td>73-76</td>
</tr>
<tr>
<td>B-</td>
<td>70-72</td>
</tr>
<tr>
<td>C+</td>
<td>67-69</td>
</tr>
<tr>
<td>C</td>
<td>63-66</td>
</tr>
<tr>
<td>C-</td>
<td>60-62</td>
</tr>
<tr>
<td>F</td>
<td>0-59</td>
</tr>
</tbody>
</table>

In this question, we will use a definition of an association and association list slightly different from what was discussed in class.

;; An association (As) is (list Sym Nat),
;; where
;; * the first item is the key, and
;; * the second item is the associated value.

;; An association list (AL) is one of
;; * empty
;; * (cons As AL)
;; Note: All keys must be distinct.
Write a function called `histogram` that consumes percentage-grades, a list of natural numbers between 0 and 100 inclusive, where each number represents a student's grade. The function produces the histogram corresponding to these grades using the letter grade bins from the table.

Use an association list to represent the histogram produced. The keys in the association list should be the letter grades {'A+', 'A', 'A-', 'B+', 'B', 'B-', 'C+', 'C', 'C-', and 'F'}. The associated value for each letter grade should be the number of students whose percentage grade fell into the interval for that letter grade.

For example:
```scheme
(define list20 (list 85 77 96 88 74 59 69 81 0 78 85 75 100 92 91 68 62 72 80 79))
```

```scheme
(histogram list20) =>
  (list (list 'A+ 4) (list 'A 3) (list 'A- 2)
       (list 'B+ 3) (list 'B 2) (list 'B- 1)
       (list 'C+ 2) (list 'C 0) (list 'C- 1) (list 'F 2))
```

The association list produced should be in the same order of the keys in letter grades, i.e., starting at the 'A+' key and ending at 'F' key as shown in the example provided.