Assignment 6: Hash Tables
Software Engineering for Scientists
CU Boulder, Fall 2019

Objectives: Implement and benchmark hash functions and collision resolution strategies.

Tasks:

1. Develop modules and scripts using test-driven development. Tests DO NOT need to be committed individually. Commits should still follow the best practices.
2. Starting with the `hash_functions.py` module
   a. Implement the `h_ascii` hash function that takes a string key and a hash table size and returns a hash that is based on the sum of the ASCII values for the characters in the key
   b. Implement the `h_rolling` hash function that takes a sting key and a hash table size and returns a hash that is based on the polynomial rolling hash algorithm
   c. FOR EXTRA CREDIT implement another hash function
3. Starting with the `hash_tables.py` module
   a. Implement `LinearProbe` and `ChainedHash` classes that
      i. Take a table size and hash function as constructor parameters (NOTE: you can pass a function name as a parameter)
      ii. Have `add` and `search` functions
   b. FOR EXTRA CREDIT implement another hash table that uses a different collision resolution strategy
4. Perform experiments that test the properties of your hash functions and collision resolution strategies on different inputs. Discuss experiments and results, including figures in your README.md
   a. Feel free to use the data files and graphic scripts in https://github.com/swe4s/lectures/blob/master/hash_tables
   b. In keeping with best practices, document exactly how run all experiments and generate all figures
5. Add functional tests and unit tests to `.travis.yml`. Your tests should all pass.
6. Create a release from master tagged as 1.0