CS 696 Functional Design & Programming
Spring Semester, 2015
Assignment 2

San Diego State University -- This page last updated 9/16/15

Due Sept 30 11:59 PM

Each problem is worth 10 points. In problem 1-7 make sure that your function names and arguments are as given in the problem description as unit tests will be used to validate your answers. Each question asks you to write a function. You can write more that one function if you find it useful.

1. Write a function, divisors, with one argument a positive integer. The function returns a sequence of the divisors of N.

2. An abundant number is an integer for which the sum of its proper divisors is greater than twice number it self. For example 12 is an abundant number as its divisors are 1, 2, 3, 4, 6, 12 which totals 28. Write a function, abundance, that has one argument an integer and returns the sum of the proper divisors of the number minus the number it self. For example (abundance 12) returns 4.

3. Find all the abundant numbers less than 300.

4. Write a function, pattern-count with two arguments. The first arguments is a string, lets call it text, and the second argument is also a string, call it pattern. The function pattern-count return the number of times the pattern occurs in the text. For example

   (pattern-count “abababa” “aba”) returns 3
   (pattern-count “aaaaa” “aa”) returns 4
   (pattern-count “Abcde” “abc”) returns 0

5. Write a function, most-frequent-word, which has two arguments. The first argument is a string, the second argument is an integer, call it n. most-frequent-word returns a sequence word(s) of length n that occurs most in the string. For example

   (most-frequent-word “TCGAAGCTAGACGCTAGTAGCTAGTGTGCA” 4) returns
   (“CTAG” “GCTA”)

6. Given integers L and t, a string Pattern forms an (L, t)-clump inside a (larger) string Text if there is a contiguous substring of Text of length L in which Pattern appears at least t times. For example, TGCA forms a (25,3)-clump in the following Text: gatcagcataagggtcccTGCAATGCATGACAAGCCTGCAgttgttttac. Write a function findclumps with four arguments Text, k (integer), L (integer), t (integer) that returns a sequence strings of length k that form a (L, t)-clump in Text. For example:

   (def text
   “CGGACTCGACAGATGTGAAGAAATGTGAAGACTGAGTGAAGAGAAGAGGAAACAC
   GACACGACATTGCGACATAATGTACGAATGTAATGTGCCTATGGC”)

(find-clumps text 5 75 4) returns (“CGACA” “GAAGA” “AATGT”)

7. The file weather.dat is a tab separated file with a month of weather data. You will find a link to the file on the assignment page of the course website. The first column is the day number in the month, the second column is the maximum temperature of the day, the third column is the minimum temperature of the day. Write a function maximum-spread with one argument. The argument is the path to a file of the same format as weather.dat. maximum-spread returns the day number of the day that has the largest temperature spread. There are Clojure libraries that parse tab separated files. You are not to use them.

What to Turn in

Answer all questions in a single Clojure file (file extension .clj). Use a comment to separate and label each questions. Place the questions in order in your file. Zip up the file and turn in your zipped file using assignment 1 link on the course portal.

Late Penalty

An assignment turned in 1-7 days late, will lose 5% of the total value of the assignment per day late. The eight day late the penalty will be 40% of the assignment, the ninth day late the penalty will be 60%, after the ninth day late the penalty will be 90%. Once a solution to an assignment has been posted or discussed in class, the assignment will no longer be accepted. Late penalties are always rounded up to the next integer value.