CIS 455 – Homework #3

Due on myCourses *before* 11:59pm on Sunday, April 1st, 2018

Instructions: There are 7 problems worth a total of 100 points.

Submit on myCourses: solutions to the written parts, hard copies of all source code and screenshots of sample runs which thoroughly verify the correctness of your Rosalind code.

Document and indent your programs properly. You will be graded on both your solutions and your ability to show their correctness. Please rename and upload your source code as a .txt file. Do **not** zip up your code, since I can't comment directly in your .doc/.pdf on myCourses.

If you feel it would help, you are encouraged to work together on homework. But remember that you must submit your own work, as the point of the homework is to learn the material. *If you do work with others on homework, you must write the names of those you worked with on your homework.*

Submitting modified versions of other people's work as your own (or copying from websites) is considered cheating and will result in an "F".

Late Homework will be penalized!

Problem 3-1. (5+5=10 points) Jones & Pevzner, (*modified version* of Problem 5.1, page 143). Suppose you have a maximization algorithm, *A*, that has an approximation ratio of ¼. When run on some input π, *A*(π) = 12.

1. What can you say about the true (correct) answer OPT = OPT(π)? *Explain your answer.*

    a. OPT(π) >= 3;
    b. OPT(π) <= 3;
    c. OPT(π) >= 12;
    d. OPT(π) <= 12;
    e. OPT(π) >= 48;
    f. OPT(π) <= 48;

2. What if A is a minimization algorithm? (*Hint:* This is a trick question)

Problem 3-2. (20+3+3=26 points) Jones & Pevzner, (*modified version* of Problem 5.4, page 143).

1. Perform the *ImprovedBreakpointReversalSort* algorithm with π = 3 4 6 5 8 1 7 2
   (Remember to start with **0** 3 4 6 5 8 1 7 2 **9** and to follow the algorithm below, **particularly line 3**).

   For each step (similar to the class activity):
    - Show the reversal you have chosen
    - Show the resulting sequence
    - Show the number of breakpoints that remain after performing the reversal

```
ImprovedBreakpointReversalSort(π)
1  while b(π) > 0
2      if π has a decreasing strip
3          Among all possible reversals, choose reversal ρ that minimizes b(π·ρ)
4      else
5          Choose a reversal ρ that flips an increasing strip in π
6      π ← π·ρ
7      output π
8  return
```

2. The *if-test* in line 2 of the *ImprovedBreakpointReversalSort* algorithm ensures that the algorithm never gets stuck in a situation were there is no way to eventually decrease the number of breakpoints.

   Again using π = 3 4 6 5 8 1 7 2, construct a permutation σ where this if-test is needed. (Find a step that *ImprovedBreakpointReversalSort* might have ended up with where there are no decreasing strips, and no reversal that reduces the number of breakpoints). If you found such a step in (1), you can use that one. You **do not** need to solve it from this step on, just give an example of such a permutation.

3. Since this is an approximation algorithm, there might be a sequence of reversals that is shorter than the one found by *ImprovedBreakpointReversalSort*.

   Again using π = 3 4 6 5 8 1 7 2, construct a permutation σ for which this is the case. (Instead of following the *ImprovedBreakpointReversalSort* algorithm, perform any series of reversals—starting from π to the identity permutation—in fewer steps than it took you in part 1 when you used *ImprovedBreakpointReversalSort*).

Problem 3-3. (10 points) Jones & Pevzner, (*modified version* of Problem 5.5, page 143).
Find a permutation with no decreasing strips for which there exists a reversal that reduces the number of breakpoints. Use that reversal, and further reversals afterwards, to reach **this identity permutation: π = 1 2 3 4 5**

Problem 3-4. (9 points) Jones & Pevzner, Problem 5.13, page 144 (*read the description between 5.11 and 5.12*).

Given permutations π and σ, a breakpoint between π and σ is defined as a pair of adjacent elements π_i and π_{i+1} in π that are separated in σ. For example, if $\pi = 143256$ and $\sigma = 123465$, then $\pi_1 = 1$ and $\pi_2 = 4$ in π form a breakpoint between π and σ since 1 and 4 are separated in σ. The number of breakpoints between π=01432567 and σ=01234657 is three (14, 25 and 67), while the number of breakpoints between σ and π is also three (12, 46 and 57).

Given permutations π^1 = 124356, π^2 = 143256, and π^3 = 123465, compute the number of breakpoints between:

(1) π^1 and π^2
(2) π^1 and π^3
(3) π^2 and π^3

(Use the first permutation as the given sorting order. For example, given π^3 = 123465, 4 and 6 are adjacent and 4 and 5 are not.)

Problem 3-5. (10 points) <u>Jones & Pevzner, Problem 6.4, page 212.</u>
Modify *DPChange* (on page 151 and below) to return not only the smallest *number* of coins but also the correct combination of coins.

```
DPCHANGE(M, c, d)
bestNumCoins_0 ← 0
for m ← 1 to M
    bestNumCoins_m ← ∞
    for i ← 1 to d
        if m ≥ c_i
            if bestNumCoins_{m−c_i} + 1 < bestNumCoins_m
                bestNumCoins_m ← bestNumCoins_{m−c_i} + 1
return bestNumCoins_M
```

Problem 3-6. (15 points) **<u>Rosalind:</u>** *no late credit will be given for Problem 3-6 after 3/31*

1. Solve **Problem 3: Compute the Number of Breakpoints in a Permutation**
2. (10 points) Submit it successfully on *Rosalind* by **March 31st**
3. What is your Rosalind username?
4. (5 points) Share your solution on *Rosalind* with the rest of the class, after you have submitted your correct answer. Direct link is here: http://rosalind.info/classes/problems/12199/solutions/

Problem 3-7. (20 points) **<u>Rosalind:</u>** *no late credit will be given for Problem 3-7 after 3/31*

1. Solve **Problem 4: Find the Minimum Number of Coins Needed to Make Change**
2. (10 points) Submit it successfully on *Rosalind* by **March 31st**
3. (5 points) Share your solution on *Rosalind* with the rest of the class, after you have submitted your correct answer. Direct link is here: http://rosalind.info/classes/problems/12198/solutions/
4. Go back to the Shared Solutions for *Rosalind Problem 2: Compute the Number of Times a Pattern Appears in a Text*. Direct link is here: http://rosalind.info/classes/problems/12041/solutions/
5. (5 points) Vote for your **three** favorite algorithms (please do not vote for your own) by clicking on the *upvote arrow*. Give the username of the 3 authors whose code you preferred and explain what you liked about them and why they were your favorite (in just a few sentences). You will not be graded on who you select, as long as it's not yourself!