CS 561, HW6

Prof. Jared Saia, University of New Mexico

1. You are buying pizza for n friends. You ask each friend to give you a \$1 coin. When you get to the pizza place, it's closed, so you return the coins to your friends in a random order.

   (a) Consider a single friend. What is the probability that they get back their own coin?

   (b) What is the expected number of friends who will get back their own coins?

   (c) Use Markov's inequality to get an upper bound on the probability that at least 2 of your friends will get their coin back

   (d) What is the expected number of pairs of friends i and j, such that friend i gets j's coin and friend j gets i's coin?

   (e) Say that you lose a random subset of $n/2$ coins on the way back, and you randomly distribute the remaining coins to a random subset of $n/2$ of your friends. Now use a union bound to get an upper bound on the probability that at least one of your friends gets their own coin back.

2. Problem 14-7 (15-7 3rd edition) "Viterbi Algorithm". This is an application of dynamic programming used frequently in machine learning.

3. *Dance, Dance Revolution (DDR)* is played on a platform with 4 squares. You are given an input sequence σ over the symbols A, B, C and D, representing the four squares. In round $i \geq 1$, one of your feet must be on the square $\sigma[i]$. Your feet must always be in different squares, and you can move at most one foot at the start of each round to any new square. Your left foot starts in square A and right foot in square B.

   You are a lazy dancer. So your goal is to maximize the following *laziness score*: the number of rounds in which neither foot moves. Below is an example game play.

σ	A	C	A	D	C	D	B
Feet position	(A,B)	(A,C)	(A,C)	(D,C)	(D,C)	(D,C)	(B,C)
Point?	1	0	1	0	1	1	0

You scored 4 points since there are 4 rounds where neither foot moved.

(a) Write a recurrence relation for the value $m(i, \ell, r)$ which gives the maximum score possible on the first i symbols of σ if your left foot ends in square ℓ and your right foot ends in square r.

(b) Describe a dynamic program to return the max score for any input σ of length n based on your recurrence. What are the dimensions of your table? How do you fill it in? What is the final value returned? What is the runtime of your algorithm?

4. You are given n balloons. Each balloon is painted with a number on it represented by an array $nums$. You are asked to burst all the balloons. If you pop the balloon at index i; and ℓ is the index of the closest un-popped balloon to the left, r is the index of the closes un-popped balloon to right, you get a number of coins equal to $nums[\ell] * nums[i] * nums[r]$. If ℓ or r goes out of bounds of the array, then treat it as if there is a balloon with a 1 painted on it. Your goal is to return the maximum coins you can collect by bursting the balloons wisely.

   (a) Give an example input showing that if in every step, you greedily select the balloon that gives you the largest number of coins for that step, this may not maximize the total number of coins.

   (b) To set up a dynamic program, describe in words the smaller problem(s) whose solutions can help you solve the big problem.

   (c) Write a recurrence relation for the dynamic program.

   (d) Describe the dynamic program and give the runtime.