Problem 1. In this problem we will study balanced non-binary vectors of different kinds. A vector $v \in \{0, \ldots, q - 1\}$ will be called:

- *symbol-balanced* if the number of times each symbol appears is the same ($n/q$ and $n$ is a multiple of $q$),
- *weight-balanced* if $\sum_{i=1}^{n} v_i = n(q - 1)/2$ and $n$ is even,
- *polarity-balanced*, where $q$ and $n$ are even, if the number of times the symbols $0, \ldots, (q - 1)/2$ appear is the same as the number of times the symbols $q/2, \ldots, q - 1$ appear.

We let $A, B, C$ be the set of all symbol-balanced, weight-balanced, polarity-balanced vectors, respectively.

(a) Calculate the size of each set $A, B, C$ and conclude on the minimum redundancy of any code for symbol-balanced and polarity-balanced vectors (you don’t need to calculate the minimum redundancy for the weight-balanced vectors).

(b) Use Knuth’s algorithm in order to design codes for symbol-balanced, weight-balanced, and polarity-balanced vectors. Prove correctness, analyze the number of redundancy symbols, and, for symbol-balanced and polarity-balanced vectors, compare with the lower bound on the redundancy. You can assume that the redundancy symbols do not need to satisfy the constraint in each case.
Problem 2. The signature of a permutation $\sigma = (\sigma_1, \ldots, \sigma_n) \in S_n$, is a binary vector of length $n - 1$, $s(\sigma) = (s_1, \ldots, s_{n-1})$, defined as follows. For all $1 \leq i \leq n - 1$, $s_i = 1$ if and only if $\sigma_i < \sigma_{i+1}$. For $k \leq n - 1$, let $A(n, k)$ be the value $A(n, k) = |\{\sigma \in S_n \mid w_H(s(\sigma)) = k\}|$.

(a) Prove that for all $k \leq n - 2$,
$$A(n, k + 1) = (k + 2)A(n - 1, k + 1) + (n - k - 1)A(n - 1, k).$$

(b) For an odd integer $n$, a permutation $\sigma \in S_n$ will be called balanced if the Hamming weight of its signature is $\frac{n-1}{2}$, that is, $w_H(s(\sigma)) = \frac{n-1}{2}$. Construct a code of balanced permutations with efficient encoder and decoder while minimizing the redundancy of the code. Hint: You can ideas from Knuth’s algorithm. (Note that here you need to guarantee that the outcome permutation after encoding is balanced).

Problem 3.

(a) For a permutation $\sigma \in S_n$, we define $W_\tau(\sigma) = \{(i, j) \mid i < j, \sigma^{-1}(i) > \sigma^{-1}(j)\}$ (lecture 5, slide 12). Prove that $d_\tau(\sigma, \pi) = |W_\tau(\sigma) \triangle W_\tau(\pi)| = |W_\tau(\sigma) \setminus W_\tau(\pi)| + |W_\tau(\pi) \setminus W_\tau(\sigma)|$.

(b) In the weighted Kendall’s $\tau$ scheme with weights $w = (w_1, \ldots, w_{n-1}) = (1, 2, \ldots, n - 1)$, the cost of changing the adjacent elements in locations $i$ and $i+1$ is $i$. Then, the weighted-Kendall’s $\tau$ distance between two permutations $\sigma$ and $\pi$, denoted by $d_{w\tau}(\sigma, \pi)$, is the minimum cost in order to change $\sigma$ to $\pi$. For example, for $\sigma = (1, 2, 3, 4)$ and $\pi = (3, 2, 1, 4)$, $d_{w\tau}(\sigma, \pi) = 4$ for the path
$$\begin{align*}
(3, 2, 1, 4) &\rightarrow^1 (2, 3, 1, 4) \rightarrow^2 (2, 1, 3, 4) \rightarrow^1 (1, 2, 3, 4).
\end{align*}$$

Note that we could choose a path with higher cost (5),
$$\begin{align*}
(3, 2, 1, 4) &\rightarrow^2 (3, 1, 2, 4) \rightarrow^1 (1, 3, 2, 4) \rightarrow^2 (1, 2, 3, 4).
\end{align*}$$

Design an algorithm that calculates the weighted-Kendall’s $\tau$ distance between two given permutations $\sigma, \pi \in S_n$. Prove its correctness and complexity.

(c) Bonus: Repeat the previous part for the weights $w_1 = (1, 2, \ldots, n/2 - 1, n/2, n/2 - 1, \ldots, 1)$ and $w_2 = (n/2, n/2 - 1, \ldots, 2, 1, 2, \ldots, n/2 - 1, n/2)$, where $n$ is even.

Problem 4. For all $n \geq 2$, let $C_n$ be the code
$$C_n = \left\{ x = (x_1, \ldots, x_n) \in \{0, 1\}^n \mid \sum_{i=1}^{n} ix_i \equiv 0 \mod (n + 1) \right\}.$$

An insertion in a vector $x = (x_1, \ldots, x_n) \in \{0, 1\}^n$ is the event where a bit is inserted in one of the locations of $x$ so the outcome is a vector of length $n + 1$. The set $I(x)$ is the set of all vectors of length $n + 1$ that can be received as an outcome of a single insertion in the vector $x$.

(a) Prove that the code $C_n$ can correct a single insertion.

(b) Prove that for all $x \in \{0, 1\}^n$, $|I(x)| = n + 2$.

(c) Prove that an upper bound on a code correcting a single insertion is $\frac{2^{n+1}}{n+2}$. 

2