**St. Francis Xavier University**
**Department of Computer Science**

# CSCI 435: Algorithms and Complexity
# Assignment 1
# Due January 24, 2023 at 8:15am

---

**Assignment Regulations.**

- This assignment must be completed individually.
- Please include your full name and email address on your submission.
- You may either handwrite or typeset your submission. If your submission is handwritten, please ensure that the handwriting is neat and legible.

---

[6 marks] 1. Recall the many liars problem from the lecture notes. In our discussion of the problem, we only considered the case where the total number of people in the room was even.

Describe how to solve the problem in the case where the total number of people in the room is odd. Explain your modification to the procedure described in the lecture notes, and argue that your modification is sufficient to solve the problem.

[10 marks] 2. Let $G = (V, E)$ be a directed graph where $V = \{v_1, v_2, \ldots, v_n\}$. We say that $G$ is an *ordered graph* if it satisfies the following two properties:

- Each edge exits a vertex with a lower index and enters a vertex with a higher index (i.e., every directed edge is of the form $(v_i, v_j)$ where $i < j$); and
- Each vertex, except for $v_n$, has at least one outgoing edge (i.e., for all vertices $v_i$, $1 \leq i \leq n-1$, there exists at least one edge $(v_i, v_j)$).

Suppose that we are given an ordered graph $G$, and we want to find the length of the longest path that begins at vertex $v_1$ and ends at vertex $v_n$.

(a) Explain why the following algorithm doesn't correctly solve the ordered graph longest path problem by giving an example of an ordered graph on which the algorithm returns a suboptimal answer.

**Algorithm:** Ordered graph longest(?) path

```
w ← v_1
L ← 0
while there is an outgoing edge from w do
    choose the edge (w, v_j) for which j is smallest
    w ← v_j
    L ← L + 1
return L
```

(b) Design an efficient algorithm that takes as input an ordered graph $G$ and returns the length of the longest path beginning at vertex $v_1$ and ending at vertex $v_n$.

[6 marks] 3. Recall that, in the secretary problem, the $k = n/\mathrm{e}$ stopping strategy is asymptotically optimal for large values of $n$. However, it's possible to compute directly the best value of $k$ for small values of $n$.

Let $\mathbb{P}[n, k]$ denote the success probability of the stopping strategy where you observe the first $k$ secretaries out of $n$ secretaries. Design an algorithm that computes $\mathbb{P}[n, k]$ for all $1 \leq k \leq n$ and returns the best value of $k$. Your algorithm should have a time complexity of $O(n)$.

[8 marks] 4. Suppose we want to implement the LRU algorithm for the paging problem. To do so, we must implement an appropriate data structure to efficiently maintain both page IDs and timestamps for each page in the cache. Describe a data structure that maintains a cache index in this way and provides the following three operations:

- EXISTS($p$): checks whether a page with ID $p$ exists in the cache and returns true or false.
- OLDEST(): returns the page ID corresponding to the page with the oldest timestamp.
- UPDATE($p, t$): changes the timestamp of the page with ID $p$ to time $t$ (where $t$ is always larger than the existing timestamp of page $p$).

Your data structure should implement each of these operations with an average time complexity of $O(1)$, and it should have a space complexity of $O(k)$ for a cache of size $k$. You can make use of as many standard data structures as you need (e.g., arrays, hash tables, heaps, linked lists, etc.).