Due 29th September 2022 at 5pm Sydney time

Your solutions must be typed, machine readable PDF files. *All submissions will be checked for plagiarism!*

For each question requiring you to design an algorithm, you *must* justify the correctness of your algorithm. If a time bound is specified in the question, you also *must* argue that your algorithm meets this time bound.

Partial credit will be awarded for progress towards a solution.

Question 1   *The Cheapest Fridge*

[**30 marks**] Song wants to buy a fridge with volume at least V cubic centimetres. The shop sells a large variety of fridges. More precisely, for each positive integer x, the shop sells a fridge for x dollars with the following dimensions:

- width $3x$ centimetres,
- depth $2x + 1$ centimetres and
- height 2^x centimetres.

1.1   [12 marks] Design an algorithm which runs in $O(\log V)$ time and finds the minimum amount that Song must spend to buy a suitable fridge.

1.2   [18 marks] Design an algorithm which runs in $O(\log(\log V))$ time and finds the minimum amount that Song must spend to buy a suitable fridge.

You may choose to skip 1.1, in which case your answer to 1.2 will be marked as your answer to 1.1 also.

Question 2   *Counting Occurrences*

[**30 marks**] Answer the following:

> Be very careful with the requested time bounds.

2.1   [6 marks] Let $X = [x_1, \ldots, x_n]$ be a **sorted** array of n positive integers. Design an algorithm to count the number of occurrences of each *distinct* integer in X in worst case $O(n)$ time.

2.2   [12 marks] Let $Y = [y_1, \ldots, y_n]$ be an **unsorted** array of n positive integers. Design an algorithm to count the number of occurrences of each *distinct* integer in Y in *expected* $O(n)$ time.

2.3   [12 marks] Let $Z = [z_1, \ldots, z_n]$ be a **sorted** array of n positive integers. You are also given an integer $k \leq n$, the number of *distinct* integers in this array. Design an algorithm to count the number of occurrences of each distinct integer in Z in worst case $O(k \log n)$ time.

Question 3   *Counting Inversions Between Arrays*

[**20 marks**] Let A and B be two arrays of length n, each containing a random permutation of the numbers from 1 to n. An inversion between the two permutations A and B is a pair of values (x, y) where the index of x is less than the index of y in array A, but the index of x is more than the index of y in array B.

Design an algorithm which counts the total number of inversions between A and B that runs in $O(n \log n)$ time.

Question 4 *Red & Yellow Flowers*

[**20 marks**] You are given an array of n flowers that are either red or yellow. Your goal is to find the number of subarrays where there are more red flowers contained within the subarray, than there are yellow flowers outside the subarray.

> Subarrays must be contiguous.

4.1 [**6 marks**] Describe a method that runs in $O(n)$ time, which pre-processes the input array such that you can then calculate the number of red flowers within any contiguous subarray in $O(1)$ time.

4.2 [**14 marks**] Design an algorithm that achieves your goal in $O(n \log n)$ time.