Probability and Computing
Hilary Term 2019
Exercise Sheet 1

1. The problem of verifying multiplication of square matrices takes as input three $n \times n$ matrices A, B, C, with integer entries and returns whether $C = A \cdot B$.

   Give an efficient co-RP randomised algorithm for the problem. Your algorithm should do $O(n^2)$ arithmetic operations (multiplications and additions of integers). You can assume that the entries of the matrices are small nonnegative integers, say in the range $\{0, \ldots, 2^n - 1\}$.

   Hint: The straightforward matrix multiplication of two $n \times n$ matrices needs $O(n^3)$ operations, but multiplication of an $n \times n$ matrix and a vector needs only $O(n^2)$ operations.

2. We have a function $F : \mathbb{Z}_n \to \mathbb{Z}_m$. We know that for $0 \leq x, y \leq n-1$, $F((x+y) \bmod n) = (F(x)+F(y)) \bmod m$. The only way to evaluate F is to use a look-up table that stores the values of F. Unfortunately an Evil Adversary has changed the value of $1/5$ of the table entries–but we don't know which ones.

   Describe a simple randomised algorithm that, given an input z, outputs a value that equals $F(z)$ with probability at least $1/2$. Your algorithm should use as few lookups and as little computation as possible.

   Hint: The (deterministic) algorithm that looks up the value of $F(z)$ in the look-up table and returns that value is not a correct solution, because it returns the correct value with probability 0 if the Adversary has changed the table entry for z.

   Suppose I allow you to repeat your algorithm three times. What should you do in this case, and what is the probability that your enhanced algorithm returns the correct answer?

3. Consider the following program:

```
Input: integer array A[1..n]
z := infinity
for i = 1 to n
  do if A[i] < z then z:=A[i]
return z
```

   If the input array is chosen uniformly at random from all permutations of $\{1, \ldots, n\}$ find the expected number of times that variable z changes its value over a run of the algorithm.

4. Suppose there is a stream of items passing by one at a time. We can either save or discard each item as it passes, but we cannot recover a discarded item and we only have room to store k items at any one time. We don't know beforehand the length of the stream. Describe a randomised procedure so that after the whole stream has passed the set of items held in memory is uniformly distributed among all k-element subsets of all items seen. Carefully justify the correctness of your proposed solution.

5. You throw a fair die until you see the first 6. What is the expected number of throws (including the throw giving 6) conditioned on the event that all throws gave even numbers?

   Hint: It is not 3.