Exercise 2
Due: 3/12/2020

1. Show that 5 multiplications are sufficient to compute the square of a 2×2 matrix. Can we use this observation recursively and obtain a $O(n^{\log 5})$ time algorithm for computing A^2 for general $n \times n$ matrices?

2. Multiply the polynomials $1 - 4x - 3x^2$ and $2 - 5x$ using the DFT algorithm shown in class. (That is, evaluate the polynomials on the appropriate roots of unity, find the values of the product polynomial on these, and interpolate to obtain its coefficients. Do not use the recursive version, FFT.)

3. Consider the deterministic Multiplicative Weights algorithm shown in class (pick an action by weighted majority).

   (a) Show that for $0 < \epsilon < 1/2$, using the following update rule:

   $$w_i^{t+1} = \begin{cases} w_i^t & x_i^t = o^t \\ (1-\epsilon)w_i^t & x_i^t \neq o^t \end{cases},$$

   if the algorithm makes M mistakes, then $M \leq 2(1+\epsilon) \cdot m + 2\ln n/\epsilon$ (where m is the number of mistakes the best expert does, and n the number of experts).

   (b) Show that any deterministic algorithm must make at least $2m$ mistakes in the worst case.

4. Suppose we have a set $X = \{x_1, \ldots, x_n\}$ of elements, each with a label $\ell_i \in \{0,1\}$ (think of x_i as a picture, and the label indicates whether it is a cat or not). We also have a set of classifiers H, and an algorithm A that given any distribution $\mathcal{D}$ on X, outputs $h \in H$ such that

   $$\Pr_{i \sim \mathcal{D}}[h(x_i) = \ell_i] \geq 0.51 \ .$$

   Show an algorithm that produces a set of $T = O(\log n)$ classifiers $h^{(1)}, \ldots, h^{(T)} \in H$, such that the majority vote among these T classifiers yields the correct label for all $1 \leq i \leq n$.

5. **Bonus:** (2 points to course grade)

   A 3-term arithmetic progression is 3 integers $a < b < c$ such that $a + c = 2b$. Given a set of integers $A \subseteq \{1, \ldots, n\}$, find an $O(n \log n)$ time algorithm that computes the number of 3-term arithmetic progressions in A.