# 7th tutorial



**Definition** (*Discrete Fourier Transform (DFT)*): Fix $\omega$ to be a primitive $n$-th root of 1. DFT is the mapping $\mathcal{F} : \mathbb{C}^n \to \mathbb{C}^n$ defined as $\mathbf{y} = \mathcal{F}(\mathbf{x})$, $y_j = \sum_{k=0}^{n-1} x_k \omega^{jk}$, where $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$.

Notice that in FFT, we are computing the graph representation of a polynomial $P(\mathbf{x})$ given by a coefficient vector $\mathbf{p} = (p_0, \ldots, p_{n-1})$ (i.e., $P(\mathbf{x}) = \sum_{j=0}^{n-1} p_j x^j$) with respect to the points $\omega^0, \omega^2, \ldots, \omega^{n-1}$ which is exactly $\mathcal{F}(\mathbf{p})$.

**1. Fourier Images.** Compute the Fourier images of the following vectors:

(1) $(x, x, \ldots, x)$ for $x \in \mathbb{R}$ (try first $x = 1$)
(2) $(1, -1, 1, -1, \ldots, 1, -1)$
(3) $(1, 0, 1, 0, \ldots, 1, 0)$
(4) $(\omega^0, \omega^1, \omega^2, \ldots, \omega^{n-1})$
(5) $(\omega^0, \omega^2, \omega^4, \ldots, \omega^{2n-2})$

**2. Properties.** Which properties are expressed by the 0-th and the $n/2$-th coefficient of the Fourier image?

**3. Basis Image.** What is the Fourier image of the unit vector $\mathbf{e}_j$, i.e., the vector whose $j$-th coordinate is 1 and all other coordinates are 0?

**4. Basis Inversion.** For each $j$ find a vector whose Fourier image is $\mathbf{e}_j$. How to use this to construct the inversion of $\mathcal{F}$, i.e., $\mathcal{F}^{-1}$?

**5. DFT of a Real Vector.** Show that the Fourier image $\mathbf{y}$ of a real vector $\mathbf{x}$ is *antisymmetric*, i.e., $\mathbf{y}_j = \overline{\mathbf{y}}_{n-j}$ for all indices $j$. What will be the Fourier image of an antisymmetric vector?

---

**6. Fast Multiplication.** Derive an $\mathcal{O}(n \log n)$ algorithm for multiplying two $n$-bit numbers.

**7. Pasha's Task.** You're given an $n$-bit binary string $\mathbf{s} \in \{0, 1\}^n$. Define a pair of indices $l < r$ with $l + r = 2m$ (i.e., $m$ is the middle index between $l, r$) to be *nice* if $s_l = s_r = s_m$. Count the number of nice pairs $l, r$. The naive algorithm runs in $\mathcal{O}(n^2)$ (try all pairs). Try to get $\mathcal{O}(n \log n)$.
*Hint1:* If you can solve the task for all nice pairs with $s_l = s_r = s_m = 1$, then you can also solve it for al pairs with $s_l = s_r = s_m = 0$. Focus only on the first case.
*Hint2:* Recall the multiplication of two polynomials: what is the coefficient at $x^k$ in $P \cdot Q$?