1. Consider the Uniqueness Test Algorithm:
   Check whether all the elements in a given array are distinct
   • Input: An array A[0…n-1]
   • Output: Return “true” if all the elements in A are distinct and “false” otherwise

   **ALGORITHM** \texttt{UniqElements}(A[0..n-1])
   
   \begin{algorithmic}
   \FOR {i \leftarrow 0 \text{ to } n-2}
   \FOR {j \leftarrow i+1 \text{ to } n-1}
   \IF {A[i] = A[j]} \text{return} \text{false}
   \ENDIF
   \ENDFOR
   \ENDFOR
   \text{return} \text{true}
   \end{algorithmic}

   Analysis gives:

   \[ T(n) = \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} 1 = \sum_{i=0}^{n-2} (n-1-i) = \sum_{i=1}^{n-1} i = n(n-1)/2 = \Theta(n^2) \]

   Another algorithm would first sort the array and check for equal adjacent elements (duplicates). Write this algorithm using a sorting function \texttt{Sort}(A[0..n-1]). If the sorting function does \( O(n \log n) \) comparisons, what would be the number of comparisons made by this algorithm in the worst case?

2. The multiplication of two square matrices \( A_{n \times n} \) and \( B_{n \times n} \) produces a matrix \( C_{n \times n} = A \times B \) whose elements are given by:

   \[ C_{ij} = \sum_{k=0}^{n-1} A_{ik} B_{kj}, \quad i, j = 0,...,n-1 \]

   Write the algorithm to receive \( A, B \), and return \( C \) using the above definition. Find the number of arithmetic operations done by this algorithm as a function of \( n \).

3. The discrete Fourier Transform of an image \( f(x,y) \) of size \( N \times N \) pixels is computed as follows:

   \[ F(u,v) = \left( \frac{1}{N^2} \right) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \exp{-2\pi j (ux + vy)/N} \]

   where \( u = 0, 1, \ldots, N-1 \), and \( v = 0, 1, \ldots, N-1 \), and \( j = \text{sqrt}(-1) \).

   If \( f(x,y) \) is generally complex, and the evaluation of \( \exp{\ldots} \) costs one complex multiplication, how many complex multiplications are needed to compute the Fourier Transform of the whole image?

4. The sine of an angle \( x \) (in radians) can be computed using the \( n \)-term expansion:

   \[ \sin(x) = x - \left( \frac{x^3}{3!} \right) + \left( \frac{x^5}{5!} \right) - \left( \frac{x^7}{7!} \right) + \ldots \quad (n \text{ terms}) \]

   \[ = \sum_{1 \leq i \leq n} (-1)^{i+1} x^{2i-1} / (2i-1)! \]

   • Write an \( O(n) \) algorithm that uses the above expansion to compute \( \sin(x) \) for arbitrary values of \( n \geq 1 \), with \( x \) of type \texttt{float}.
   • Show that the number of \texttt{float arithmetic operations} needed is:

   \[ T(n) = 3(n-1) + 2 \]
5. Suppose a sequence $A$ of $n$ numbers should already be in ascending order. Thus for the sequence $(6,2,9,5,8,7)$ there are 6 pairs that are out of sequence:

$(6,2), (6,5), (9,5), (9,8), (9,7), (8,7)$

- Implement an algorithm to return the number of these out-of-order pairs. **Hint:** An out-of-order pair $(A_i, A_j)$ is such that: $(i < j)$ AND $(A_i > A_j)$.
- Determine $T(n) =$ number of array element comparisons.
- What is the Big-O of this algorithm?