**EZINWA-OBI, CHIDIMMA**

**21CJ029210**

**COMPUTER ENGINEERING**

**CEN434 ASSIGNMENT**

**Quick-sort Algorithm**

* **Definition**: Quick-sort is a divide-and-conquer sorting algorithm that works by selecting a 'pivot' element from the array and partitioning the other elements into two sub-arrays, according to whether they are less than or greater than the pivot.
* **Process**:
  1. Choose a pivot element.
  2. Partition the array into two halves: elements less than the pivot and elements greater than the pivot.
  3. Recursively apply the same process to the left and right sub-arrays.
* **Time Complexity**:
  1. Best and average case: O(nlog⁡n)O(n \log n)O(nlogn)
  2. Worst case: O(n2)O(n^2)O(n2) (occurs when the smallest or largest element is always chosen as the pivot)
* **Space Complexity**: O(log⁡n)O(\log n)O(logn) due to recursion stack.
* **In-place**: Does not require additional storage beyond the input array.

**Divide and Conquer Algorithms**

* **Definition**: Divide and conquer is an algorithm design paradigm that works by recursively breaking down a problem into two or more sub-problems of the same or related type until these become simple enough to be solved directly.
* **Process**:
  1. **Divide**: Split the problem into smaller sub-problems.
  2. **Conquer**: Solve the sub-problems recursively.
  3. **Combine**: Combine the solutions of the sub-problems to form a solution for the original problem.
* **Characteristics**: Efficient for problems that can be broken down into smaller, independent sub-problems. Examples include Merge Sort and Quick Sort.
* **Applications**: Used in various domains such as numerical algorithms, searching algorithms, and computational geometry.

**Min-Max Problem Algorithms**

* **Definition**: The Min-Max problem involves finding the minimum and maximum values in a given dataset. It is often addressed in decision-making and game theory.
* **Algorithms**:
  1. **Linear Search**: Traverse the entire dataset to find the min and max values.
  2. **Min-Max Algorithm**: A more efficient approach that compares pairs of elements, reducing the number of comparisons.
     + For an even number of elements, compare them in pairs and keep track of the current min and max.
     + For an odd number of elements, handle the last element separately.
* **Time Complexity**:
  1. Linear Search: O(n)O(n)O(n)
  2. Min-Max Algorithm: O(n)O(n)O(n) with fewer comparisons (approximately 3n/23n/23n/2).
* **Applications**: Useful in statistics, game theory (like finding optimal moves), and data analysis.