Series No. 02 – Sorting Algorithms

Exercise 01: A Special Sort (Competition – Solve it 2023)

Write a function called ***"maxValue"*** that takes an array ***"T"*** of positive integers (or zero) as a parameter and rearranges its elements to form the largest possible value. Since the maximum value can be very large, return an array of characters (string) instead of an integer.

Examples of inputs and outputs:

```
Input : T = {1, 2, 3, 4, 5, 6, 7, 8, 9, 0}
Output : "9876543210"

Input : T = {54, 14, 89, 2, 301, 45, 7}
Output : "8975432143012"
```

The function should have the following header:

```
char* maxValue(int T[], int n);
```

Hint: Use any sorting algorithm with a custom comparator.

Exercise 02: The Minimum Possible Value

Write a function called ***"minSameDigits"*** that takes an integer as a parameter and rearranges its digits such that:

- The value of the result is the smallest possible while keeping the same digits.
- There should be no zero at the beginning of the number (except if the number is equal to 0).

Hints:

- Think about using a slightly modified version of a sorting algorithm (assume you already have a sorting algorithm implemented).
- Be careful with negative numbers (in this case, you should maximize the absolute value).

Exercise 03: The Missing Value

Let ***T*** be an array of integers of size n. We assume that ***T*** contains all integers from 0 to n except one. The goal of this exercise is to write a function that returns the missing value.

- The naive solution is to iterate through the array and test for each integer from 0 to n if it is present in the array. Provide the implementation of this solution. What is its complexity?
- How can we reduce the complexity of this algorithm? Implement this solution.

Exercise 04: Finding Two Numbers with a Given Sum

The goal of this exercise is to write a function called "twoSum" that takes an array ***T*** and an integer ***S*** as arguments and returns ***true*** if the array ***T*** contains two numbers whose sum is equal to ***S***. For example, if ***T = [1, 2, 3, 4, 5]*** and ***S = 7***, the function should return ***true*** because 3 + 4 = 7.

- Start by implementing a naive version of this function that tests all possible pairs of numbers in the array. What is the complexity of this function?
- How can we use sorting algorithms to improve the complexity of this function to sub-quadratic complexity? Implement this version.