National University of Computer & Emerging

Sciences (NUCES) Islamabad

School of Computing

DATA STRUCTURES – FALL 2023

Cyber Security Department

LAB 02

Learning Outcomes

In this lab you are expected to learn the following:

- Algorithm Optimization
- Sorting Algorithms

Time Complexity

This is a very good resource to understand the concept of time complexity:

https://www.geeksforgeeks.org/understanding-time-complexity-simple-examples/

Instead of measuring actual time required in executing each statement in the code,

Time Complexity considers how many times each statement executes.

Example 1:

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Hello World";
    return 0;
}
```

Time Complexity: In the above code "Hello World" is printed only once on the screen.

So, the time complexity is **constant: O (1)** i.e., every time a constant amount of time is required to execute code, no matter which operating system or which machine configurations you are using.

Auxiliary Space: O (1)

Example 2:

```
#include <iostream>
using namespace std;

int main()
{

    int i, n = 8;
    for (i = 1; i <= n; i++) {
        cout << "Hello World !!!\n";
    }
    return 0;
}
```

Time Complexity: In the above code "Hello World !!!" is printed only **n times** on the screen, as the value of n can change.

So, the time complexity is **linear: O(n)** i.e., every time, a linear amount of time is required to execute code.

Auxiliary Space: O (1)

1. Algorithm Optimization

Question1:

Print all positive integer solutions to the equation $a^3 + b^3 = c^3 + d^3$ where a, b, c, and d are integers between 1 and 1000.

A brute force solution will just have four nested for loops. Something like:

```
n = 1000

for a from 1 to n

      for b from 1 to n

            for c from 1 to n

                  for d from 1 to n

                        if a³ + b³ == c³ + d³

                              print a, b, c, d
```

This algorithm iterates through all possible values of a, b, c, and d and checks if that combination happens to work. The time complexity of the algorithm is $O(n^4)$, reduce the runtime from $O(n^4)$ to $O(n^3)$.

Hint: $\boldsymbol{d = \sqrt[3]{a^3 + b^3 - c^3}}$

Question2:

Reverse the contents of an integer array **in-place** i.e., without using any other data structure in O(n) runtime.

```
void reverse (int array [], int arr_length) {

//Your code here

}
```

2. Sorting Algorithms

Question3:

Write a program to perform **Selection Sort** on a 1D Array. Take size and elements as inputs from the user.

Algorithm:

Step 1 − Set MIN to location 0

Step 2 − Search the minimum element in the list

Step 3 − Swap with value at location MIN

Step 4 − Increment MIN to point to next element

Step 5 − Repeat until list is sorted