Programming Fundamentals
Lab #6

Exercise 1 (The simplest Java method with a time complexity of O(X^3)):

```
void cubicTimeMethod(int N) {
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            for (int k = 0; k < N; k++) {

            }
        }
    }
}
```

Exercise 2:

The given Java method foo(int N) has a time complexity of O(N), as it contains two loops where the first loop iterates N times and the second loop iterates a constant number of times (1000000), which is not dependent on N.

Exercise 3:

The given Java method bar(int N) has a time complexity of O(log N). The loop iterates logarithmically with base 2 with respect to N. In each iteration, the value of i is doubled, hence the logarithmic behavior.

Exercise 4:

To perform a binary search for the number 73 in the given sorted array:
{1, 14, 15, 24, 55, 59, 73, 90, 94, 99}

1. Iteration 1: Middle element = 55, Left bound = 0, Right bound = 9
2. Iteration 2: Middle element = 90, Left bound = 6, Right bound = 9
3. Iteration 3: Middle element = 73, Left bound = 6, Right bound = 7

Exercise 5:

For the insertion and selection sort algorithms, let's trace the execution with the given array {1, 29, 14, 15, 94}:

Insertion Sort:

- ❖ Start with the second element (index 1) of the array.
- ❖ Compare the current element with the one before it.
- ❖ If the current element is smaller, swap it with the previous element and continue comparing until the current element is in its correct position relative to the sorted elements before it.
- ❖ Move to the next element and repeat steps 2-3 until all elements are sorted.

1. After 1st iteration: {1, 29, 14, 15, 94}
2. After 2nd iteration: {1, 14, 29, 15, 94}
3. After 3rd iteration: {1, 14, 15, 29, 94}
4. After 4th iteration: {1, 14, 15, 29, 94}
5. After 5th iteration: {1, 14, 15, 29, 94}

Selection Sort:

```
For i = 1 to N-1 do:
    ❖ a. Set current_element to A[i]
    ❖ b. Set j to i-1
    ❖ c. While j >= 0 and A[j] > current_element do:
            ➢ i. Swap A[j] and A[j+1]
            ➢ ii. Decrement j
    ❖ d. Set A[j+1] to current_element
```

1. After 1st iteration: {1, 29, 14, 15, 94} → {1, 14, 29, 15, 94}
2. After 2nd iteration: {1, 14, 29, 15, 94} → {1, 14, 15, 29, 94}
3. After 3rd iteration: {1, 14, 15, 29, 94} → {1, 14, 15, 29, 94}
4. After 4th iteration: {1, 14, 15, 29, 94} → {1, 14, 15, 29, 94}
5. After 5th iteration: {1, 14, 15, 29, 94} → {1, 14, 15, 29, 94}