Exercise 1 - DSA TA 2024

Part 1: Identifying Big-O Notations

1. Identify the Big-O of the below Java code

```java
public void no1(int n) {
    System.out.println(n*2);
}
```

2. Identify the Big-O of the below Java code

```java
public void no2(int[] n) {
    for (int i = 0; i <= n.length; i++) {
        System.out.print(n[i]);
    }
}
```

3. Identify the Big-O of the below Java code **(Extra points for identifying the algorithm name)**

```java
public void no3(int[] arr, int n) {
    int first =  0;
    int last = arr.length - 1;
    while (first <= last) {
        int mid = (last - first / 2) + first;
        if (mid == n) {
            System.out.println(mid);
        }
        else if (mid < n) {
            last = mid - 1;
        }
        else {
            first = mid + 1;
        }
    }
}
```

4. Identify the Big-O of the below Java code **(Extra points for identifying the algorithm name)**

```java
public int partition(int[] n, int begin, int end) {
    int pivot = n[end];
    int i = begin - 1;
    for (int j = begin; j < end; j++) {
        if (n[i] <= pivot); {
            i++;
            int temp = n[i];
            n[i] = n[j];
            n[j] = temp;
        }
    }
    int temp = n[i+1];
    n[i+1] = n[end];
    n[end] = temp;
    return i+1;
}

public void no4(int[] n, int begin, int end) {
    if (begin < end) {
        int partition = partition(n, begin, end);
        no4(n, begin, partition-1);
        no4(n, partition+1, end);
    }
}
```

5. Identify the Big-O of the below Java code **(Extra points for identifying the algorithm name)**

```java
public void no5(int[] n, int no) {
    int i,j,temp;
    boolean swap;
    for (i = 0; i < no - 1; i++) {
        swap = false;
        for (j = 0; j < no - i - 1; j++) {
            if (n[j] > n[j+1]) {
                temp = n[j];
                n[j] = n[j+1];
                n[j+1] = temp;
                swap = true;
            }
        }
        if (swap == false) {
            break;
        }
    }
}
```

Part 2 - Programming

Create an Animal Class that satisfies 3 out of 4 OOP principles (Inheritance, Encapsulation, Polymorphism, and Abstraction). Ex = the Animal class can be the parent of a Lion class (Inheritance). Make a constructor, setters, and getters for the Animal class. Create a driver class to run your code. Attach screenshots of the code and results along with a GitHub Repo link for your code. In your GitHub Repo, make a readme.md to explain your code (search online on how to properly document your repository in a readme.md file).