timothy allec

**Name, SID, Date** .....

# In Class Assignment 14: Bucket Sort

**Benjamin Sanders, MS** November 25, 2020

## 1 Introduction

You will need to work individually to complete this assignment. Write your name at the top of all pages for this assignment. Turn in all work to Blackboard on or before the deadline to receive credit.

You may use additional libraries and online resources, if you get them approved in writing, over email, from the instructor first. If you have received approval from the instructor, write the approved libraries and any references in the space below.

.....
.....
.....
.....

## 2 Assignment Description

### 2.1 Big Picture

Professor Sanders used to use Bucket Sort to sort student work by last name when he used to have physical printouts to grade.

### 2.2 Algorithm Implementation

Implement the following algorithm in Java, using the Vector data structure for any 1-D array, 2-D array, or linear algebra purposes.

BUCKET-SORT($A$)

1 let $B[0..n-1]$ be a new array
2 $n = A.length$
3 **for** $i = 0$ **to** $n-1$
4 make $B[i]$ an empty list
5 **for** $i = 1$ **to** $n$
6 insert $A[i]$ into list $B[\lfloor nA[i] \rfloor]$
7 **for** $i = 0$ **to** $n-1$
8 sort list $B[i]$ with insertion sort
9 concatenate the lists $B[0], B[1], \ldots, B[n-1]$ together in order

```
public class BucketSort {
    public static void bucketSort(int[] A) {
        int n = A.length;
        int max = getMax(A); // Find the maximum value in A
        int[] B = new int[max + 1]; // B needs to accommodate values from 0 to max
        for (int i = 0; i < n; i++) {
            B[A[i]]++;
        }
        int index = 0;
        for (int i = 0; i < B.length; i++) {
            for (int j = 0; j < B[i]; j++) {
                A[index++] = i;
            }
        }
    }

    public static int getMax(int[] A) {
        int max = A[0];
        for (int i = 1; i < A.length; i++) {
            if (A[i] > max) {
                max = A[i];
            }
        }
        return max;
    }
```

```
    public static void main(String[] args) {
        int[] A = {56, 23, 90, 51, 29, 34,
55};
        bucketSort(A);
        for (int i = 0; i < A.length; i++) {
            System.out.println(A[i]);
        }
    }
}
```

### 2.3 Time Complexity Analysis

Where $n$ is the number of data points in $A$, analyze the time complexity of the given algorithm with respect to $n$. Write the result of your analysis in big-$O$ notation, i.e. $O(n^2)$ in the space below.

O(n)
.....

### 2.4 Space Complexity Analysis

Where $n$ is the number of data points in $A$, analyze the space complexity of the given algorithm with respect to $n$. Write the result of your analysis in big-$O$ notation, i.e. $O(n \cdot log(n))$ in the space below.

O(n^2)
.....

**Name, SID, Date** .....

## 2.5 New Algorithm Implementation

Create a new version of the given algorithm that handles String input of length 32 characters. Implement this in Java.

```
/*Create a new version of the given algorithm that handles String input
of length 32 characters. Implement this
in Java
 * After sorting, the strings are ordered based on their lengths,
 * with shorter strings appearing before longer ones.
 * If two strings have the same length, their order remains the same
 *  as in the original array.
 */

import java.util.ArrayList;
import java.util.List;

public class StringBucketSort {
    public static void stringBucketSort(String[] A) {
        int n = A.length;
        int maxStringLength = getMaxStringLength(A); // Find the maximum string length in A
        List<List<String>> buckets = new ArrayList<>();
        for (int i = 0; i < maxStringLength; i++) {
            buckets.add(new ArrayList<>());
        }

        // Distribute strings into buckets based on their lengths
        for (int i = 0; i < n; i++) {
            String str = A[i];
            int strLength = str.length();
            buckets.get(strLength - 1).add(str);
        }

        // Concatenate the strings from the buckets to form the sorted array of strings
        int index = 0;
        for (int i = 0; i < maxStringLength; i++) {
            List<String> bucket = buckets.get(i);
            for (String str : bucket) {
                A[index++] = str;
            }
        }
    }

    public static int getMaxStringLength(String[] A) {
        int max = 0;
        for (String str : A) {
            if (str.length() > max) {
                max = str.length();
            }
        }
        return max;
    }

    public static void main(String[] args) {
        String[] A = {"abcdefg", "zxy", "mnopqrstuvwxyz", "pqrstu", "ijk"};
        stringBucketSort(A);
        for (String str : A) {
            System.out.println(str);
        }
    }
}
```

# 3 What to Turn In

*Turn in one PDF or Word document on Blackboard, containing the following items.*

1. All pages scanned or photographed of the In Class Assignment completed document.
2. Any additional pages you used to complete the assignment.
3. All code created for the assignment, along with test cases.
4. One statement indicating which parts of your implementation(s) are working, and which parts are not.
5. Screenshots demonstrating the code working, if it is working.