# Sorting Practice – Count sort, Radix sort, Bucket sort

For all sorting algorithms: **Time and Space complexity. Stable? Adaptive? (Data moves)**

## Non-comparison sorting

**NCS1.** (6 points) You are using count sort to sort an array of N numbers, where each number is from the range [0,M]. What is the time complexity (as Theta) of the number of data moves? (For example swapping two records requires 3 data moves.). Briefly justify your answer.

**NCS2.** (9 points**)**  (Radix sort)

Show how **LSD radix sort** sorts the following numbers in the given representation (base 10). Show the numbers after each complete round of count sort.

| Index: | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Original Array: | 513 | 145 | 320 | 235 | 141 | 433 | 2 |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

**NCS3.** (4 points) What is the operation you do to map/scale values from range [A,B] to range [X,Y]? You can assume that A < B and X < Y. (E.g. [47,49] -> [20,30], [5,10] -> [21,23])

**NCS4.** (5 pts) Assume you want to use bucket sort to sort an array A, that has integers in the range [-100, 350]. (i.e. A[i]≥-100 and A[i]≤350, for all valid i ). You will use 50 buckets. Write the formula to find the index, `bucketIdx`, for the bucket where A[i] should go.
Make sure you indicate any rounding (up or down) if necessary.

**NCS5.** (6 pts) Fill in the arrays to show the required processing with count sort for the data below.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Original array | C, Alice | B, Jane | A, Jane | F, John | A, Matt | D, Sam | B, Tom |

Counts array after part 1 (counts of each key):

| | |
|---|---|
| Index: | |
| Counts array: | |

Counts array after part 2 (after cumulative sum):

| | |
|---|---|
| Index: | |
| Counts array: | |

Show the counts array and the copy array after each of the next 2 big steps of count sort as shown in slide 6 (i.e. after a first element is placed in the copy array, and after a second element is placed in the copy array). Create columns as needed in the tables below.

| | |
|---|---|
| Index: | |
| Counts array: | |
| Counts array: | |

| | |
|---|---|
| Index: | |
| Copy array: | |
| Copy array: | |

**NCS6.** a) We run bucket sort (version covered in class) on the array [0.3, 0.15, 0.27, 0,8, 0.61].

How many buckets will be created?

What are the elements in each bucket? Here we assume the numbers in the array are in the range [0,1) and we map these to the bucket indexes (like in the pseudocode). We do NOT use the min and max from the array.
When giving the elements in a bucket, give them in SORTED order, separated by commas and with no extra spaces. Say *empty* if the bucket is empty.

b) What if 10 buckets were created?