CS 240: Data Structures and Data Management **Winter 2022**

Tutorial 3: May 30

1. Give the expected run-time for the randomized insertion sort algorithm below. You can assume that the $random(k)$ operation takes $O(1)$ time and produces an integer in $\{0, 1, ..., k-1\}$ with equal probability, where $k \geq 1$ is an integer.

Algorithm 1: $randomized - insertion - sort(\mathcal{A})$

Input: Array $\mathcal{A}$
Output: None (Array A is sorted in-place)

```
1  for i from 0 to n − 1 do
2  |  j = random(i + 1)
3  |  swap(A[i], A[j])
4  end
5  for i from 1 to n − 1 do
6  |  j = i − 1
7  |  while (j ≥ 0) and (A[j] > A[j + 1]) do
8  |  |  swap(A[j], A[j + 1])
9  |  |  j = j − 1;
10 |  end
11 end
```

2. Let $0 < \epsilon < 1$. Suppose that we have an array A of n items such that the first $n - n^{\epsilon}$ items are sorted. Describe an $O(n)$ time algorithm to sort A.

3. Let A and B be two bitstrings of length n (modelled here as arrays where each entry is 0 or 1). A *string-compare* tests whether A is smaller, larger, or the same as B and works as follows:

Algorithm 2: $str\text{-}cmp(A, B, n)$

```
1 for (i = 0; i < n; i++) do
2 |  if (A[i] < B[i]) then return "A is smaller"
3 |  if (A[i] > B[i]) then return "A is bigger"
4 end
5 return "They are equal"
```

Show that the average-case run-time of *str-cmp* is in $O(1)$. You may use without proof that $\sum_{i \geq 0} \frac{i}{2^i} \in O(1)$.