Note: This exam has 4 questions for a total of 100 points. Answer all questions. Write all your answers on the blue books.

Question I (25 points total)

(a) Let \( A = \{1, 2, \ldots, n\} \), where \( n \geq 2 \). We carry out two successive trials on \( A \), with each trial consisting of selecting an integer uniformly randomly from \( A \). Show that the expectation of the sum of the two values chosen is \( n + 1 \). (9 points)

(b) Let \( B = \{1, 2, \ldots, 10\} \). Each Bernoulli trial on \( B \) consists of choosing an integer uniformly randomly from \( B \). A trial is considered a success if the chosen integer is either 7 or 9. Let \( X \) denote the random variable that gives the number of successes in 25 successive independent trials. Find the variance of \( X \). Show work. (8 points)

(c) Recall the Hire-Assistant function whose pseudocode is given below.

1. Current-Best = 0. /* Initialization: Dummy Candidate 0. */
2. for \( i = 1 \) to \( n \) do
   if Candidate \( i \) is better than Current-Best
      then Hire candidate \( i \) and set Current-Best = \( i \).

Assume that all possible permutations of the \( n \) candidates are equally likely. Compute the probability that the last candidate in the list won’t be hired when we execute the above algorithm. Show work. (8 points)

Question II (30 points total)

(a) Consider the binary Max-Heap represented by the first seven elements of an array \( B \) of size eight.

\[
\]

We want to insert the new value 105 into the heap. Show the array \( B \) after the new value is inserted. (8 points)

(b) Show that a binary heap with \( n \) nodes has exactly \( \lceil n/2 \rceil \) leaves. (10 points)

(c) Suppose we have a Max-Heap which is initially empty. We perform a mix of \( n \) INSERT operations and \( n \) EXTRACT-MAX operations on the heap. Show that, in the worst-case, the total time for these \( 2n \) operations is \( \Omega(n \log n) \). (12 points)
Question III (30 points total)

(a) Use Radix-Sort to sort the following collection of strings into dictionary order: RAT, CAT, WAR, ASK, RIG and PIT. Be sure to show the partially sorted sequence after sorting by each column. (12 points)

(b) Pseudocode for Bucket-Sort is given at the bottom of this page. Assume that to sort $t$ numbers, the time used by Insertion-Sort is at most $ct^2$ for some constant $c > 0$. Prove that the worst-case running time of Bucket-Sort is $O(n^2)$. (18 points)

Note: An answer of the form “The worst-case occurs when all the keys end up in the same bucket” is not rigorous and hence not acceptable.

Question IV (15 points)

Suppose each row of an $n \times n$ matrix $A$ consists of 0’s and 1’s such that in any row of $A$, all the 1’s come before any 0’s in that row. Assuming that $A$ is already in memory, design an algorithm with a running time of $O(n \log n)$ (not $O(n^2)$) to compute and output the total number of 1’s in $A$.

Your answer must include the following: (i) A description of the algorithm and (ii) an explanation of why the running time of the algorithm is $O(n \log n)$. You won’t receive any credit if your algorithm is incorrect or its running time is asymptotically worse than $O(n \log n)$.

Pseudocode for Bucket-Sort:

Bucket-Sort($A$)

// Each element of $A[1..n]$ is a real number in the interval [0,1).
// $B[0..n-1]$ is an array of pointers.

1. for $i = 1$ to $n$ do
   Insert $A[i]$ into the list pointed to by $B[\text{floor}(n*A[i])]$.

2. for $i = 0$ to $n-1$ do
   Sort list $B[i]$ into increasing order using Insertion-Sort.

3. Concatenate lists $B[0], B[1], ..., B[n-1]$ together in order.

Note: In Step 1 above, we have used $\text{floor}(n*A[i])$ to denote $\lfloor n \ast A[i] \rfloor$. 