DAA Assignment IV

1. Trace the list with values 10, 20, 10, 20, 10, 5 using following algorithms:

    A. Sort by comparison counting and discuss Stable property of the algorithm. Also Workout worst case and best case analysis.

    B. Sort by Distribution counting and discuss the stable property of the algorithm. Also Workout worst case and best case analysis.

Show the trace in the form of a table showing the intermediate values at each iteration till you get the final sorted array.

2.

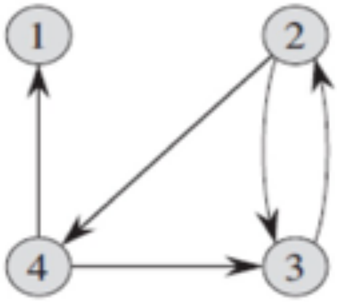
    A. Compute C(6, 3) by applying the dynamic programming algorithm.

    $C(n, k) = C(n - 1, k) + C(n - 1, k - 1)$

    $C(n, n) = C(n, 0) = 1$

n / k	0	1	2	3
0	1			
1	1	1		
2	1	2	1	
3	1	3	3	1
4	1	4	6	4
5	1	5	10	10
6	1	6	15	20

B. **Find the transitive closure of the following digraph using Warshall's algorithm.**



A_0	1	2	3	4
1	0	0	0	0
2	0	0	1	1
3	0	1	0	0
4	1	0	1	0

A_1	1	2	3	4
1	0	0	0	0
2	0	0	1	1
3	0	1	0	0
4	1	0	1	0

A_2	1	2	3	4
1	0	0	0	0
2	0	0	1	1
3	0	1	**1**	**1**
4	1	0	1	0

A_3	1	2	3	4
1	0	0	0	0
2	0	**1**	1	1
3	0	1	1	1
4	1	**1**	1	**1**

A_4	1	2	3	4
1	0	0	0	0
2	**1**	1	1	1
3	**1**	1	1	1
4	1	1	1	1

3. **Apply Boyer-Moore pattern search algorithm to search a pattern** `MAHARAJ` **in the text** `RAJA–KI–RAJA–MAAHARAJA`**. Also find number of character comparisons.**

BAD SYMBOL TABLE							
–	A	H	I	J	K	M	R
7	1	4	7	7	7	6	2

Index:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Text:	R	A	J	A	-	K	I	-	R	A	J	A	-	M	A	A	H	A	R	A	J	A
Pattern:	M	A	H	A	R	A	J															

No match at 0, 1 comparisons so far, using bad character rule to advance by 7.

Index:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Text:	R	A	J	A	-	K	I	-	R	A	J	A	-	M	A	A	H	A	R	A	J	A
Pattern:	M	A	H	A	R	A	J															

No match at 7, 2 comparisons so far, using bad character rule to advance by 6.

Index:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Text:	R	A	J	A	-	K	I	-	R	A	J	A	-	M	A	A	H	A	R	A	J	A
Pattern:								M	A	H	A	R	A	J								

No match at 13, 3 comparisons so far.

Index:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Text:	R	A	J	A	-	K	I	-	R	A	J	A	-	M	A	A	H	A	R	A	J	A
Pattern:														M	A	H	A	R	A	J		

No match at 14, 10 comparisons so far.

Index:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Text:	R	A	J	A	-	K	I	-	R	A	J	A	-	M	A	A	H	A	R	A	J	A
Pattern:															M	A	H	A	R	A	J	

No match at 15, 11 comparisons so far.

Index:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Text:	R	A	J	A	-	K	I	-	R	A	J	A	-	M	A	A	H	A	R	A	J	A
Pattern:																M	A	H	A	R	A	J

Character comparisons = 11
Match(es) = 0

4. For the input `12, 8, 30, 20, 56, 75, 31, 19, 90, 33, 80` **and hash function** `h(K) = K mod 9`

A. Construct the open hash table.

```
BUILD-MAX-HEAP(A)
A.heap-size = A.length
for i = ⌊A.length/2⌋ downto 1
      MAX-HEAPIFY(A, i)
```

```
MAX-HEAPIFY(A, i)
l = LEFT(i)
r = RIGHT(i)
if l ≤ A.heap-size and A[l] > A[i]
      largest = l
else largest = i
if r ≤ A.heap-size and A[r] > A[largest]
      largest = r
if largest ≠ i
      exchange A[i] with A[largest]
      MAX-HEAPIFY(A, largest)
```

```
HEAPSORT(A)
BUILD-MAX-HEAP(A)
for i = A.length downto 2
      exchange A[1] with A[i]
      A.heap-size = A.heap-size − 1
      MAX-HEAPIFY(A, 1)
```

0 →	90
1 →	19
2 →	20 → 56
3 →	12 → 27 → 75

4 →	31
5 →	
6 →	33
7 →	
8 →	8 → 80

B. **Find the largest number of key comparisons in a successful search in this table.**

3, for key = 75.

C. **Find the average number of key comparisons in a successful search in this table.**

(7 * 1 + 3 * 2 + 1 * 3) / (7 + 3 + 1)

= 16/11
= 1.45?

5.

A. **Write an algorithm for Heapsort.**

B. **The assignment problem can be stated as follows: There are n people who need to be assigned to execute n jobs, one person per job. (That is, each person is assigned to exactly one job and each job is assigned to exactly one person.) The cost that would accrue if the i-th person is assigned to the j-th job is a known quantity C[i, j] for each pair i, j =1,...,n. The problem is to assign the people to the jobs to minimize the total cost of the assignment. Express the assignment problem as a linear programming problem.**

Let x_{ij} be a 0-1 variable indicating an assignment of the ith person to the jth job (or, in terms of the cost matrix C, a selection of the matrix element from the ith row and the jth column). The assignment problem can then be posed as the following linear programming problem:

$$\begin{array}{lll}
\text{minimize} & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} & \text{(the total assignment cost)} \\
\text{subject to} & \sum_{j=1}^{n} x_{ij} = 1 \text{ for } i = 1,\dots n & \text{(person } i \text{ is assigned to one job)} \\
& \sum_{i=1}^{n} x_{ij} = 1 \text{ for } j = 1,\dots n & \text{(job } j \text{ is assigned to one person)} \\
& x_{ij} \in \{0,1\} \quad \text{for } i = 1,\dots,n \text{ and } j = 1,\dots,n &
\end{array}$$