Week-4 solution

(1) Counting sort is a non comparison sort. so number of comparison is 0.

Ans → (a)

(2) The auxiliary space complexity of counting sort is $O(n+k)$ where k = range of input

Ans → (a)

(3) Radix sort is generally implemented as an out of place algorithm, since it needs to create a second, copied array in order to handle the work of sorting.

ans → (b) False

(4) Bucket sort is not an inplace sorting algo.

ans → (d)

(5) Bucket sort is most efficient in the case when the input is uniformly distributed.

ans → (b)

(6) The run time of SELECT algorithm is optimal for group size 5. & for group size 3 $T(n) > O(n)$ ∴ $T(n) = O(n)$ for group size 5.

Ans → (b)

(7) Counting sort algorithm is efficient when range of data to be sorted is fixed.

Since the range is from 0 to 255 (for ASCII)

so the range is fixed.

Ans → (d)

(8) We can find kth smallest element of an unsorted array in O(n) time in worst case using SELECT method.

Ans → (a)

(9) Radix sort is a non comparison based integer sort.

ans → (a)

(10) For the number in the range from 0 to $n^d - 1$,

the radix sort runs in $\Theta(dn)$ time.

Since we know $T(n, b) = O(bn/\log n)$.

Where $r = \log n$ in the original formula

$$T(n,b) = O\left(\frac{b}{r}(n + 2^r)\right)$$

then $\Rightarrow b = d \log n \Rightarrow \underline{T(n) = O(dn)}$ ✱

Where each b-bit word is broken in b/r equal pieces.

Ans → (b)