Q1

a. To prove it, we know that in Huffman short codewords are given, with highest frequencies, smallest key words or encrypt number.

Suppose 'ch' with highest freq, $f(ch) > 2/5$. Ch should be given shorter code. But ch not given code word of length 1.

To prove by contradiction, $Ch != 2/5$ or $< 2/5$ characters given code word of length $2/5 >$ or 1. So, it is proved by contradiction that no codeword is given a keyword of length 1.

Therefore, this assumption leads to contradiction of proved that there should be atleast 1 codeword of length 1.

---

b. let's prove by contradiction, ≥ 1 Ch with freq. $< 1/3$ in Huffman coding.

In any optimal Huffman encoding, shorter words are assigned more frequent ch. If ch with freq $< 1/3$ had codeword of length 1, it should be inefficiency, contradicting optimal encoding.

2.

b. queue operations take n logn. because sorting is involved

Traversel is in $O(n)$.

$O(n \log n) + O(n) = O(n \log n)$.

c. This algo has a nature of giving freq. characters shorter keywords & opposite. & this extends coding upto (0, 1, 2) which maximizes compression. so, this code proves that freq. ch are given shorter keywoods with no prefix repeated with maximizing encoding using (0, 1, 2).

---

3. e. sorting takes $= O(n+k)$.

guard positions $= O(n)$.

Total $= O(n+k)$.

f. Calculation ensures min overlapping coverage while guarding all objects. Guard pos are determined by incrementing $(2d+1)$ until range. This guarantees the min no. of guard protecting the objects in a hallway.

4. b.

Counting sort $= O(n+K)$

so, $O(n+K)$.

c. This algo sorts on basis of swim time. Contestants with less swimming time are scheduled earlier. Since bike & run simultaneously. so, they can reach destination with having shorter swimming time.

---

5. b. sorting $= O(n+K)$.

schedule $= O(n)$.

Total $= O(n+K)$

c. Sort videos based on time duration Then checks according to r. This approach (according to ques) ensures that connection's bandwith constraint is met for all videos in schedule, allowing smooth streaming.

---

6. b. Sorting $= O(n+K)$.

Total weight time $= O(n)$.

Total $= O(n+K)$.

c. This algo calculates the time for each customer, which is the product of their job time & priority. Then, sorts on basis of total weight time. This ensures that highest priority of customers are served earlier, reducing wait time. This leads to efficient scheduling strategy.

---

7. b. Sorting $= O(n+K)$.

Total time calc. $= O(n)$.

Total. $= O(n+K)$

c. It sorts on basis of job time. It minimizes wait time. This approach guarantees customer satisfaction by prioritizing shorter job times & optimizing the printing schedule accordingly.