CSE 431/531: Algorithm Analysis and Design	Fall 2022

Homework 2

Instructor: Shi Li **Deadline: 10/9/2022**

Your Name: ______________________ Your Student ID: ________________

Problems	1	2	3	Total
Max. Score	20	30	30	80
Your Score				

Problem 1 Construct the Huffman code (i.e, the optimum prefix code) for the alphabet $\{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{e}, \mathsf{f}, \mathsf{g}\}$ with the following frequencies:

Symbols	a	b	c	d	e	f	g
Frequencies	50	20	27	25	29	85	55

Also give the weighted length of the code (i.e, the sum over all symbols the frequency of the symbol times its encoding length).

Problem 2 We are given an array A of length n. For every integer i in $\{1, 2, 3, \cdots, n\}$, let b_i be median of the sub-array $A[1..i]$. (If the sub-array has even length, its the median is defined as the lower of the two medians. That is, if i is even, b_i is the $i/2$-th smallest number in $A[1..i]$.) The goal of the problem is to output $b_1, b_2, b_3, \cdots, b_n$ in $O(n \log n)$ time.

For example, if $n = 10$ and $A = (110, 80, 10, 30, 90, 100, 20, 40, 35, 70)$. Then $b_1 = 110, b_2 = 80, b_3 = 80, b_4 = 30, b_5 = 80, b_6 = 80, b_7 = 80, b_8 = 40, b_9 = 40, b_{10} = 40$.

Hint: use the heap data structure.

Problem 3 In the interval covering problem, we are given n intervals $(s_1, t_1], (s_2, t_2], \cdots, (s_n, t_n]$ such that $\bigcup_{i\in[n]}(s_i, t_i] = (0, T]$. The goal of the problem is to return a smallest-size set $S \subseteq \{1, 2, 3, \cdots, n\}$ such that $\bigcup_{i\in S}(s_i, t_i] = (0, T]$.

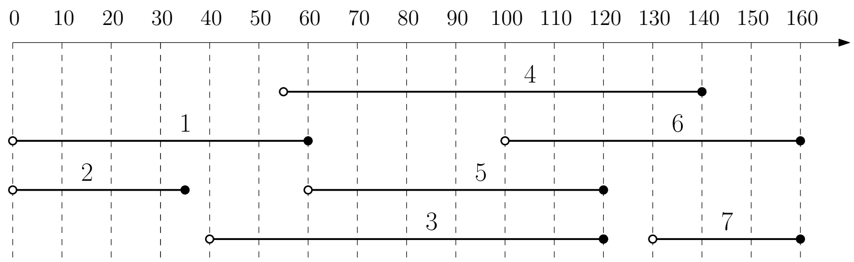

Figure 1: An instance of the interval covering problem.

For example, in the instance given by Figure 1. The intervals are $(0, 60]$, $(0, 35]$, $(40, 120]$, $(55, 140]$, $(60, 120]$, $(100, 160]$, $(130, 160]$. Then we can use 3 intervals indexed by 1, 4, 7 to cover the interval $(0, 160]$. This is optimum since no two intervals can cover $(0, 160]$.

Design a greedy algorithm to solve the problem. it suffices for your algorithm to run in polynomial time. To prove the correctness of the algorithm, it is recommended that you can follow the following steps:

- Design a simple strategy to make a decision for one step.
- Prove that the decision is safe.
- Describe the reduced instance after you made the decision.