# EAST WEST UNIVERSITY
## Department of Computer Science and Engineering
## B.Sc. in Computer Science and Engineering Program
## Mid 1 Examination, Fall 2021 Semester

| | |
|---|---|
| **Course:** | **CSE246 Algorithm, Section-03** |
| **Instructor:** | **Jesan Ahammed Ovi, Senior lecturer, CSE Department** |
| **Full Marks:** | **25 (15 will be considered for final grading)** |
| **Time:** | **1 Hour 20 Minutes (Writing) plus 10 Minutes (Uploading)** |

**Note:** There are FIVE questions, answer ALL of them. Course Outcome (CO) and Mark of each question are mentioned at the right margin.

---

1. Consider the following sentence. **Apply** Hoffman algorithm to generate variable length code for each character. Show details of your calculation. [CO3, C3, Marks: 5]

   *"TOO HOT TO RUN"*

2. Consider the following recursive relation. **Analysis** the complexity of the relation using a suitable technique. [CO4, C4, Marks: 5]

$$T(n) = \begin{cases} C \ ; n = 1 \\ 4 * T\left(\frac{n}{2}\right) + n/2 + C \ ; n > 1 \end{cases}$$

   Here C is constant and n is the input.

3. Xander Budnick is a famous you tuber. Currently he is in a mission to visit several tourist spots and stay distinct number of days in each spot. So while he is visiting a particular spot, he is a problem to find out the number of days he should stay in that spot. You are given a sorted non-negative distinct integers, your task is to **design** an algorithm that finds the smallest missing non-negative element on it. Your solution should not takes more than logn instructions in worst case. Consider following examples for better understanding. [CO3, C4, EP1, Marks: 5]

   Example:
   Input: nums[] = [0, 1, 2, 6, 9, 11, 15]
   Output: The smallest missing element is 3

   Input: nums[] = [1, 2, 3, 4, 6, 9, 11, 15]
   Output: The smallest missing element is 0

   Input: nums[] = [0, 1, 2, 3, 4, 5, 6]
   Output: The smallest missing element is 7

4. Consider the following code segment. **Analysis** the complexity of the code in terms of big-O. Show details of your calculation. [CO4, C3, Marks: 5]

```
for(i = 2 ; i <= n/2 ; i++){
        j = 1 ;
        while(j <= m ){
        sum++ ;
        j *= 10 ;
       }
}
```

5. Consider the following String S and Pattern P. **Apply** Rabin-Karp algorithm to find the each occurrence of pattern P in string S. [CO2, C3, Marks: 5]

S = 0011011011000101
P = 1101

Use Horner's rule for generating integer.