BENNETT UNIVERSITY
THE TIMES GROUP

Enrolment No: E22CSEU0827 Name of Student: MADHAV GUPTA
Department/ School: BTECH CSE

END-TERM EXAMINATION, ODD SEMESTER DECEMBER 2023

COURSE CODE	**CSET202**	**MAX. DURATION**	**2 HRS**
COURSE NAME	**Data Structure Using C++**		
PROGRAM	**BTech**	**TOTAL MARKS**	**35**

Mapping of Questions to Course and Program Outcomes										
Q.No.	A1	A2	A3	A4	A5	B1	B2	B3	B4.1	B4.2
CO	2	1	1	3	2	3	4	3	4	4
PO	12	2,4,12	2,4,12	1,12	12	1,12	2,3	1,12	3	4
BTL*[1]	3	3	2	1	1	3	2	3	5	6

GENERAL INSTRUCTIONS: -

1. Do not write anything on the question paper except **name, enrolment number** and **department/school.**
2. Carrying mobile phones, smartwatches and any other non-permissible materials in the examination hall is an act of **UFM.**

COURSE INSTRUCTIONS:

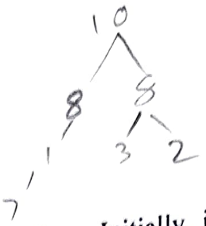
a) Write Batch number in top left corner of the answer copy.

SECTION A **(2X5=10 M)**

A1) Use the following code to express complexity in terms of big-O notation **(2 Marks)**

```
int unknown (int n){
int i,j, k=0;
for (i=n/2; i<n; i++)
          for(j=2; j<n; j=j*2)
                    k = k+n/2;
          retrun (k);
}
```

BENNETT
UNIVERSITY
THE TIMES GROUP



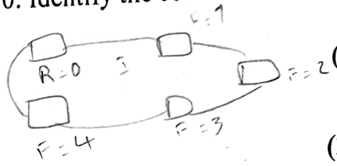
A2) A Priority-Queue is implemented as a Max-Heap. Initially, it has 5 elements. The level-order traversal of the heap is given below: 10, 8, 5, 3, 2 Two new elements "1' and "7' are inserted in the heap in that order. Solve the level-order traversal of the heap after the insertion of the elements.
(2 Marks)

A3) The following postfix expression with single digit operands is evaluated using a stack: **(2 Marks)**

8 2 3 ^ / 2 3 * + 5 1 * -

Note that ^ is the exponentiation operator. Extract the top two elements of the stack after the first * is evaluated?

A4) Suppose a circular queue of capacity (n – 1) elements is implemented with an array of n elements. Assume that the insertion and deletion operation are carried out using REAR and FRONT as array index variables, respectively. Initially, REAR = FRONT = 0. Identify the conditions to detect queue full and queue empty.



(2 Marks)

A5) Consider the following array: A = [8, 3, 6, 1]. **(2 Marks)**

Find the number of swaps required to sort them in ascending order using selection sort.

SECTION B **(5x5 = 25 M)**

B1) Consider the linear array in row major order A[-10;10], B[1925:1990], C[25].
Solve the following
(a) Find the number of elements in each array

0 1 2 3 4 5 6 7 8 9
-1-2-3-4-5-6-7-8-9-10

(2 Marks)
(b) Suppose base (A)=400 and word size (A)=2 words, find the addresses of A[-3], A[0], A[3].
(3 Marks)

2
9)22
8
4

B2) Consider inserting the keys 10, 22, 31, 4, 15, 28, 17, 88, 59 into a hash table (initially empty) of length m=11 using open addressing with the auxiliary hash function H(k) = k % m, where k is key value. Interpret the result of inserting these keys using linear probing. Also Insert the keys given in question using chaining where H(k) = k % 9. **(5 Marks)**

B3) Data is stored in tree format using red-black tree. Assume the tree is empty in the beginning. Insert 2, 1, 4, 5, 9, 3, 6, 7 in a red-black tree. Show all changes with the possible rotations.
(5 Marks)

B4) It's the holiday season, which means one thing; holiday lights! You have unboxed your string of lights to hang, but have one problem; though they are multi-colored, they are not quite in the color order that you would like. Eager to put your CSET 202 skills to work, you try to devise an algorithm to rearrange the light bulbs to be in your preferred order.

You first think about modeling your lights as a **linked list**, where each node represents one light of a certain colour, and contains a link to the next light in the sequence. We could thus define the following struct for a single node in our linked list:

```
struct LightNode {
```

BENNETT
UNIVERSITY
THE TIMES GROUP

```
char colorChar; // color; e.g. 'R' for Red, 'B' for Blue
LightNode *next; // pointer to the next light
};
```

Here's one visual example of such a linked list: (R- red, Y- yellow, W- White, B-Blue)

$$\begin{tikzcd} R \arrow[r] & Y \arrow[r] & W \arrow[r] & B \arrow[r] & W \arrow[r] & R \arrow[r] & Y \end{tikzcd}$$

You may always assume that each node has a set color. Your task is to, for a given linked list of lights, and color string such as "WWGBR" (for White-White-Green-Blue-Red), to try and rearrange the nodes to match the specified color order. **The overall approach we will use is to work from left to right with both the color string and linked list, pulling out the next instance of the light you need from the rest of the list to your right, and moving it to your current place in the list.** For instance, using the linked list illustrated above, here is how the list is modified at each step if we were trying to match to the color string "WRB":

Initially: R-Y-W-B-W-R-Y
// move first W that is to our right (index 2) to index 0
Step 1 (W, index 0): W-R-Y-B-W-R-Y
// the light we are on is already R - no change
Step 2 (R, index 1): W-R-Y-B-W-R-Y
// move first B that is to our right (index 3) to index 2
Step 3 (B, index 2): W-R-B-Y-W-R-Y

Notice how all untouched nodes remain in the same relative order.

Note that the color order string may be any length, including greater than the length of the linked list. If the color string is empty, for instance, the list should be unmodified. If the string is longer than the length of the list, just rearrange lights as long as you can to fit the color string, until you can no longer do so.

B4.1) Write a helper function called findNextLight that, given a reference to a pointer to the head of a linked list of LightNode s, and a color character (such as 'W' or 'B'), removes and returns the leftmost node of that color. In other words, it should rewire the linked list to no longer contain that node, and then return a pointer to that removed node with its next pointer set to nullptr . If no node can be found with that color, you should return nullptr .

(5 Marks)

B4.2) Write a function named rearrangeLights that takes as parameters the head of a linked list of LightNode s by reference, and a color string such as "WWBG", and modifies the linked list to match that color string, as described previously. Your function should return true if it was able to match the entire color string, or false otherwise. It is fine (and encouraged) for your rearrangeLights function to call your findNextLight function from part A.

(5 Marks)

-ALL THE BEST-