Roll Number:_____

B. E. (2nd Yr. COE/CSE)
3rd Oct., 2023
Tuesday, Time– 11:30 To 1:30 PM
Time: 2 Hours, Max Marks: 45 (Weightage: 35)

Course Code: UCS301
Course Name: Data Structures
Name of Faculty: Tarunpreet Bhatia, Simranjit Kaur, Swati Kumari, Vaibhav Pandey, Gourav Jain, Chinmay Panigrahy, Manisha Kaushal

Note: Attempt all Questions in sequence. Answer all sub-parts of each question at one place. Do mention Page No. of your attempt at front page of your answer sheet. Assume missing data (if any).

Q.1	a) Determine the worst case time complexity of the following functions by finding the frequency i.e. the number of times each statement executes:	(3)

```
i) fun1(n)                              ii) fun2 (n)
{                                       {
   for(int i=1; i<=n; i++)                 int i, j, k, s=0;
      for(int j=1; j<=i; j++)              for (i = 1; i <= n; i = i + 1)
         for(int k=1; k<=n; k++)              for (j = 1; j <= n; j = 2*j)
            break;                              for (k = 1; k <= 2^n; k = 2*k)
}                                                  s = s + 1;
                                        }
```

	b) Write the recursive function of the binary search algorithm. Show the recursive calls made for searching the key element "8" in the given array A = {2, 5, 6, 7, 8, 9, 10, 15, 16, 18, 19, 20}.	(3+2)

Q.2	Let Q be a variable of the structure named "queue", which is used to maintain a circular queue. Q contains three data members, namely, a character array arr[] with MAXSIZE = 6 (Indexing starts from 0) and two integers, viz., front and rear. Front points to the first element and rear points to the last element of the queue. The first element is inserted at MAX-1 position in an initially empty queue and then queue grows towards left. a) Write the function Enqueue() which takes Q and an element to be inserted in Q as its input and Dequeue() which takes Q as its input and returns an element to be deleted from Q if any. b) Let Q.arr[] = {'D', 'A', 'M', 'A', 'G', 'E'}, where Q.front=5 and Q.rear=0. isEmpty(Q) function returns true if Q is empty. Execute the given code snippet **(Fig 1)** and give the contents of queue along with the values of front and rear after each of the first three iterations of the while loop within main().	(3+5)

```
void Fun(struct queue Q, char x)
{
    if (isEmpty(Q) || Q.arr[Q.front]>x)
    {
        Enqueue(Q, x)
        return;
    }
    char y = Dequeue(Q)
    Fun(Q, x);
    if(y != Q.arr[Q.front])
        Enqueue(Q, y);
}
int main()
{
    while(!isEmpty(Q))
        Fun(Q, Dequeue(Q));
    return 0;
}
                Fig 1
```

Q3.	Convert the infix expression A*(B&D/E)-F*(G%H/K) to postfix expression using stack as an intermediate data structure. Show contents of the stack at each intermediate step. Use the precedence table as shown in **Table 1**.	(4)

Table 1

Operator	Precedence
%	Highest
*, /	↓
+, -	
&	Lowest

Q.4	a) Given an array A[]= {8, 7, 6, 1, 0, 9, 2} you have to sort the array in ascending order using Quick Sort algorithm by considering last element as Pivot. Show the intermediate steps. Determine which kind of input will lead to worst case and what will be the worst-case complexity?	(4+2)
	b) Given an array A consisting of 6s, 7s, and 8s such as A[]= {6,7,7,6,7,8,7,8,7,8,6,7,8,6,6} you have to sort the array of 6's, 7's and 8's in ascending order. You need to modify the partition procedure of quicksort to place the elements at their correct position in the sorted order by iterating through array only once. Give suitable pseudocode/code for the same.	(4)
Q.5	a) Consider space efficient array representation (in row-major order) of sparse matrix with a header row. You need to multiply given sparse matrices: A[][] = {{5, 5, 5}, {0, 1, 5}, {0, 2, 18}, {2, 1, 14}, {3, 1, 15}, {3, 3, 4}} and B[][] = {{5, 5, 5}, {0, 1, 5}, {1, 2, 20}, {2, 0, 8}, {3, 1, 15}, {3, 3, 24}} and show the output sparse matrix using space efficient array representation.	(3)
	b) You are given two arrays A[] = {5, 6, 7, 8, 9} and B[] = {7, 8, 9, 6, 5}. Array A[] represents the input queue and the array B[] represents the output queue. You need to determine if an array B[] is the valid stack permutation of an array A[] or not. Justify your answer by showing the proper sequence in which the operations will be called.	(2)
Q.6	Consider a group of children playing hide-and-seek choose IT (the "chor"). They stand in a circle, and the child (called the counter) recites some kind of a rhyme consisting of k words, starting with himself/herself and moving on to the next child after reciting each word in a circular fashion. The child at whom the rhyme ends is eliminated from subsequent rounds of counting. The circular linked list should shrink as children are eliminated from counting. Counting for the next round starts from the child following the eliminated child (in circular order). The process continues until one child remains, who becomes IT. The children, numbered 1 ... N, should be represented by nodes in a circular linked list. The node of circular linked list has data field as data and next pointer as next.	

a) Your first task is to complete the following code to simulate the above activity. You need not to write entire code just mention the statements corresponding to the fill in the blanks (1-14) only. Each blank should consist of single statement only. (7)

```
node * make_circle(node *head, int N)          void find_IT(node *head, int counter, int k)
{                                              {
    for (int i = 1; i ≤ N; i++)                    node *p= head, *q;
    {                                              for(int i = 1; __6__ ; i++)
        node * nn = new node();                        p=p->next;
        nn->data = i;                              while (__7__)
        nn->next = NULL;                           {
        if (head == NULL)                              for (int j = 0; __8__ ; j++)
            ___1___ ;                                  {
        else                                               ___9___ ;
        {                                                  ___10___ ;
            node* r = head;                            }
            while (___2___)                            ___11___ ;
                ___3___ ;                              delete(__12__);
                ___4___ ;                              ___13___ ;
        }                                          }
    }                                          
    ___5___ ;                                      cout<<"Child who becomes IT is "<<__14__ ;
    }                                          }
    return head;
}
```

b) Your second task is to apply the above code to a group of 5 children with k = 7 and counter = 3. You need to find the order in which the children are eliminated and then the child who becomes IT. (3)