Unique Paper Code	**:**	**32341301_OC**
Name of the Course	**:**	**B. Sc. (Hons.) Computer Science – CBCS (Old Course)**
Name of the Paper	**:**	**Data Structures**
Semester	**:**	**III**
Year of Admission	**:**	**2017, 2018**
Duration	**:**	**3 Hours**
Maximum Marks	**:**	**75**

Instructions for Candidates

Attempt Any Four questions. All questions carry equal marks.

1. A theatre has a capacity of *500* seats arranged in *50* rows and *10* seats per row. However, due to the prevailing conditions, very few customers turn up for watching the show. The theatre needs to devise an efficient data structure to store the details of the booked tickets to optimise the memory requirement. Suggest an appropriate data structure to store the details of the booked tickets. Justify the chosen data structure and give the complete class definition for the same. Also, give the code for the following functions in the class definition:
    a. function to add the details of a new booking (single seat),
    b. function to reserve multiple seats in one booking,
    c. function to print the availability of a given seat,
    d. function to display the booked seats,
    e. function for the cancellation of a booked ticket.

2. a) A college has made two sections of BSc. (Hons.) Computer Science I year and arranged the student names of each section in a singly linked list in the lexicographical order. However, in the II year, the two sections should be combined into one section and hence, the lists of student names should also be combined into one single list of names in lexicographic order. Give an efficient algorithm to do the same.

    If the college had used doubly linked lists to store the student names of each section in the lexicographic order, would combining the two lists into one list be more time efficient than using the singly linked lists for the same? Justify the answer.

    Now, one of the students decides to withdraw the admission. A pointer to the node containing the details of that student is given. Would the choice between a singly and a doubly linked list to store the list of names impact the time complexity of this operation? Justify the answer.

b) Explain the working of the given function *f*. Trace and explain the output on the given linked list, when the function is invoked as *f(head,0,0)*, such that *head* is pointing to 12.

Linked List - 12->14->61->18->5->13->10->17

```
void f (Node* n, int x1, int x2)
{
   if (n == NULL)
   {
        cout<<x1<<endl<<x2;
        return;
   }

   if(n-> data % 2 == 0)
   {
        x1++;
        f(n-> next, x1, x2);

   }


   else
   {
        x2++;
        f(n-> next, x1, x2);
    }
}
```

3. a) Given '*n*' temperature readings for a day, write an algorithm to find the duplicate temperature readings in the entire day. Also, show the sample run of your algorithm on the following temperature readings 38, 39, 40, 41, 42, 40, 39, 37.

b) Evaluate the following postfix expression using stack. Show the stack after each step.

CAB+-DBC+ - * where A=1, B=2, C=3, D=4

4. a) A lottery game generates *100* tickets each having a *10*-digit ticket number. These *100* tickets are to be hashed into a square matrix (10X10) based on the ticket number. The summation of the first five digits of a ticket number (modulo the number of rows in the square matrix) is used to identify the row number and the summation of last five digits of the ticket number (modulo the number of columns in the square matrix) is used to identify the column number of the square matrix to store that particular ticket. Two tickets may hash to the same location in the square matrix, in which case, the location becomes a bucket that can hold multiple tickets. Demonstrate the working of the above game on the following ticket numbers:

   1234567890, 0123456789, 0234516789, 0134526789, 0124536789, 0123546789, 0123465789, 9012345678

   What restrictions, if added in the ticket generation code, can reduce the chances of collision? Justify the answer.

   b) A memory block can store maximum *4* values and minimum *2* values in the ascending order and gives way to *5* more such blocks for creating hierarchical data storage. However, the top block of this hierarchical data storage is allowed to have minimum *1* value and maximum *4* values. Considering these restrictions, store the following elements in a B tree. Show the data structure after each insertion.

   <57, 72, 32, 14, 19, 22, 10, 30, 25, 7>

   Then, delete values “14” and “10” one by one. Show the data structure after each deletion.

5. Construct a binary search tree for the following keys 100, 50, 20, 30, 25, 58, 75, 64. Show all the steps and the resulting BST after each step. Thereafter, all the keys are to be deleted one by one sequentially, such that the root node is deleted every time. Show all the steps while deleting each node. Use deletion by copy method to delete a node with two children.

6. An event management team has to organise a prize distribution ceremony for a college. A pile of *100* books is placed on a table. These books, starting from the topmost book, are to be awarded to the students as they arrive on the ceremony stage. The student who arrives first gets the book first. Suggest the suitable data structures that should be used to maintain the pile of books and the list of students to be rewarded. Give the complete class definition with the necessary data members, constructors for implementing this scenario with function definitions to pick a book from the table and presenting it to the arriving student, adding *100* books originally to the pile and *100* students in the list of rewarded students.