Placement Examination – II

*Answer any **ten** questions*

(10 x 10 = 100)

1. Given a string S, write a JAVA program to find the length of the longest substring with all *distinct* characters. For example, for input *"abca"*, the output is 3 as *"abc"* is the longest substring with all distinct characters.

2. Write a JAVA program to find *all permutations* of a given string. For example, if the input string is *"ABC"*, then all permutations will be : ABC, ACB, BAC, BCA, CAB, and CBA.

3. Given an array of N elements, write a JAVA program to find the length of the *common prefix* among all strings in the array. For example, if the input array is [*"apple"*, *"ape"*, *"april"*], then the longest common prefix is *"ap"* and its length is 2.

4. Given two strings consisting of lowercase characters. Write a JAVA program to check whether the two strings are *anagram* to each other or not. An *Anagram* of a string is another string that contains same characters, only the order of characters can be different. For example, *"act"* and *"tac"* are anagram to each other.

5. Given an unsorted array A consisting of N elements, write a JAVA program to find out a *continuous sub-array* which adds to a given number S.

6. Given an array A consisting of N elements, write a JAVA program to find the position where *equilibrium* first occurs in the array. An *equilibrium* position in an array is a position such that the sum of elements before it is equal to the sum of elements after it. For example, if the input array is [1, 3, 5, 2, 2], then the equilibrium position is 3, since sum before it is 1+3=4 and the sum after it is 2+2=4.

7. Given an unsorted array A of N elements, write a JAVA program to find the first element in the array such that all of its left elements are smaller than it, and all of its right elements are larger than it.

8. Given an array containing N words consisting of lowercase characters. Write a JAVA program to find the most frequent word in the array. If multiple words have same frequency, then print the word which occurs last in the english alphabetical series.

9. Write a JAVA program to create a class *Account* with the following specifications:
    1. Class Name – *Account*
    2. Data members – *int accno, float balance*
    3. Member Methods
        1. *Account(int a, int b)* – to initialise accno as a and balance as b
        2. *void withdraw(int w)* – to maintain the balance with withdrawal(balance – w)
        3. *void deposit(int d)* – to maintain the balance with the deposit(balance + d)

Use another class *Calculate* which inherits from Class Account with the following specifications:

1. Data members – *int r, t; float si, amt;*
2. Member Methods
    1. *void accept(int x, int y)* – to initialise r=x and t=y, amt=0
    2. *void compute()* - to find simple interest and amount (formula is known I guess)
    3. *void display()* - to print account number, balance, interest and amount.

10. Write a JAVA program to define a class with the following specifications:
    1. Class Name – *Number*
    2. Data Members / Instance Variables: *int n* – to hold an integer number
    3. Member functions:
        1. *void input()* - to accept an integer number in n
        2. *void display()* - to display the integer number input in n

Now, inherit class *Number* to another class *Check* that is defined with the following specifications:

1. Class Name – *Check*
2. Data Members – *int fact, int revnum*;
3. Member Methods:
    1. *void find()* - to find and print factorial of the number used in base class.
    2. *void palindrome()* - to check and print whether the number used in base class is a *palindrome* number or not.

11. Write a JAVA program to create a class to print the area of a *Square* and a *Rectangle*. The class has two methods with the same name but different number of parameters. The method for printing area of *rectangle* has two parameters which are *length* and *breadth* respectively. The other method for printing area of *square* has one parameter which is its *side*. Display both the areas of the rectangle and the square using a third method defined by the user inside the class.

12. Write a JAVA program to display all the *prime numbers* within a given range.

13. Given an array A consisting of N elements. Write a JAVA program to search for an element *x* by using *Binary Search* algorithm.

14. Given a positive number X. Write a JAVA program to find all *Jumping Numbers* smaller than or equal to X. A number is called *jumping number* if all adjacent digits in it differ by only 1. All single digit numbers are considered as Jumping Numbers. For example, *7*, *8987* and *43434* are Jumping Numbers but *796* and *89098* are not.

15. Given an array C of size N-1 and given there are numbers from 1 to N with one element missing. Write a JAVA program to find out the *missing element* in the array.

----------XXXXX----------