CMS-A-CC-5-12-P: Object Oriented Programming Lab
OOPs Lab Using JAVA

Sample Questions:
1. Write a program in java that sorts half of element in ascending and rest half of the elements in descending order.
2. Write a program in java that accepts a 2D matrix and prints the matrix with row minimum and column minimum values.

4	3	5	**3**
1	0	7	**0**
8	4	6	**4**

1   0   5

3. Write a program in java to delete all consonants from an input string and print the result string.

4. A class called MyPoint, which models a 2D point with x and y coordinates. It contains:
   - Two instance variables x (int) and y (int).
   - A default (or "no-argument" or "no-arg") constructor that construct a point at the default location of (0, 0).
   - A overloaded constructor that constructs a point with the given x and y coordinates.
   - A method setXY() to set both x and y.
   - A method getXY() which returns the x and y in a 2-element int array.
   - A toString() method that returns a string description of the instance in the format "(x, y)".
   - A method called distance(int x, int y) that returns the distance from *this* point to another point at the given (x, y) coordinates, Write the MyPoint class. Also write a test driver (called TestMyPoint) to test all the public methods defined in the class.

5. Create a superclass 'Person' and two subclasses 'Student' and 'Staff'. The following are the instance variables and methods:
   a. For 'Person' instance variables: *name:String, address:String*. Initiate variable through constructor, incorporate one method setPerson() that updates Person variables , another method tostring() that shows Person details as "Person[name=?,address=?".
   b. For 'Student' sub class instance variables: *program:String, year:String, fees:double*. Initiate both 'Student' and 'Person' variables through constructor, incorporate one method setStudent() that updates both student and 'Person' data, another method tostring() that shows 'Person-Student' details as "Person[name=?,address=?,Program=?,Year=?,Fees=?".
   c. For 'Staff' subclass instance variables: *school:String, pay:double*. Initiate both 'Staff' and 'Person' variables through constructor, incorporate one method setStaff() that updates both 'staff' and 'Person' data, another method tostring() that shows 'Person-Staff' details as "Person[name=?,address=?,School=?,Pays=?".

   Write the classes and a test driver main class to test all functions mentioned above.

6. Create a base class 'Square' having instance variable *side:double*. Initiate variable using constructor, a method 'getVolume() : double' that calculates volume and print it. Create a derived class 'Cylinder' having instance variable *height:double*. Initiate variables of both classes through constructor, override method 'getVolume() : double' to calculate volume of cylinder taking 'side' variable of base class as 'radius' and print it.

7. Consider you are designing vehicles engine with '*speed:int, gear:int*'. you can define your engine functionalities 'speedUp(value)' and 'changeGear(value) in an interface. The class which is implementing the interface should implement all the methods in the interface.

8. Write a program in java that handles both 'ArrayIndexOutOfBoundsException' and 'ArithmeticException'.

9. Write a program in java that inputs students data(Regno, Sname, City, ContactNo) from user and insert into table 'Student_info' using JDBC connectivity. Also view all records in tabular format.

10. Write a Java program to remove a specific element from an array.

11. Write a Java program to insert an element (specific position) into an array.

12. Write a Java program to find all pairs of elements in an array whose sum is equal to a specified number.

13. Write a Java program to remove the duplicate elements of a given array and return the new length of that array.

14. Write a Java program to find the length of the longest consecutive elements sequence from a given unsorted array of integers.
    Sample array: [49, 1, 3, 200, 2, 4, 70, 5]
    The longest consecutive elements sequence is [1, 2, 3, 4, 5], therefore the program will return its length 5.

15. Write a java program to compare two strings lexicographically.

16. Write a Java program to find whether a region in the current string matches a region in another string.
    Sample Output:
    str1[0 - 7] == str2[28 - 35]? true
    str1[9 - 15] == str2[9 - 15]? false

17. Write a Java program to print all permutations of a given string with repetition.
    Sample Output:
    The given string is: PQR
    The permuted strings are:
    PPP, PPQ, PPR, RRP, RRQ, RRR

18. Write a Java method to count all words in a string.
    Test Data:
    Input the string: The quick brown fox jumps over the lazy dog.
    Expected Output:
    Number of words in the string: 9

19. Write a program in java to create Box class with parameterized constructor with an object argument to initialize length, breadth and height also create a function volume which returns the volume of the box and print it in main method.

20. Write a program in java with class Employee and do the following operations on it
    a) Create two constructor default and with Object as parameter to initialize class variables.
    b) Create a function Calculate which calculates the pf and allowances on the salary of employee and return the all values as an object.

21. Write a program to create your own exception as NegativeSizeException whenever negative values are put in an array.

22. Create a class Student with following operations
    a) create parameterized constructor to initialize the objects.
    b) create a function isEqual() to check whether the two objects are equal or not which returns the Boolean value and gets two objects.
    c) print the result in main method if objects are equals or not (take variables as your assumption)

23. Create an abstract class employee, having its properties and abstract function for calculating net salary and displaying the information. Derive manager and clerk class from this abstract class and implement the abstract method net salary and override the display method.