**Lab 1: For Java Covered until now**

Exercise 1: Write a Java program that reads a line of integers and then displays each integer and the sum of all integers. (Use StringTokenizer class)?

Exercise 2: Create a method that accepts a String and checks if it is a positive string. A string is considered a positive string, if on moving from left to right each character in the String comes after the previous characters in the Alphabetical order. For Example: ANT is a positive String (Since T comes after N and N comes after A). The method should return true if the entered string is positive.

Exercise 3: Create a method to accept date and print the duration in days, months and years with regards to current system date.

Exercise 4: You are asked to create an application for registering the details of jobseeker. The requirement is:

Username should always end with \_job and there should be at least minimum of 8 characters to the left of \_job. Write a function to validate the same. Return true in case the validation is passed. In case of validation failure return false.

Exercise 5: Create a method to find the sum of the cubes of the digits of an n digit number

Exercise 6: Create a method that can accept an array of String objects and sort in alphabetical order. The elements in the left half should be completely in uppercase and the elements in the right half should be completely in lower case. Return the resulting array.

Note: If there are odd number of String objects, then (n/2) +1 elements should be in UPPPERCASE

Exercise 7: Create a method which accepts an integer array, reverse the numbers in the array and returns the resulting array in sorted order

|  |  |
| --- | --- |
| Method Name | getSorted |
| Method Description | Return the resulting array after reversing the numbers and sorting it |
| Argument | int [] |
| Return Type | int |
| Logic | Accept and integer array, reverse the numbers in the array, sort it and return the resulting array.  Hint  Convert the numbers to String to reverse it |

Exercise 8: Create a method which accepts an array of integer elements and return the second smallest element in the array

|  |  |
| --- | --- |
| Method Name | getSecondSmallest |
| Method Description | Get the second smallest element in the array |
| Argument | int[] |
| Return Type | int |
| Logic | Sort the array and return the second smallest element in the array |

Exercise 9: Create a class with a method which can calculate the sum of first n natural numbers which are divisible by 3 or 5.

|  |  |
| --- | --- |
| Method Name | calculateSum |
| Method Description | Calculate Sum |
| Argument | int n |
| Return Type | int-sum |
| Logic | Calculate the sum of first n natural numbers which are divisible by 3 or 5. |

Exercise 10: Create a class with a method to find the difference between the sum of the squares and the square of the sum of the first n natural numbers.

|  |  |
| --- | --- |
| Method Name | calculateDifference |
| Method Description | Calculate the difference |
| Argument | int n |
| Return Type | int - Sum |
| Logic | Find the difference between the sum of the squares of the first n natural numbers and the square of their sum.  For Example if n is 10,you have to find  (1^2+2^2+3^2+….9^2+10^2)-  (1+2+3+4+5…+9+10)^2 |

Exercise 11: Create a method to check if a number is an increasing number

|  |  |
| --- | --- |
| Method Name | checkNumber |
| Method Description | Check if a number is an increasing number |
| Argument | int number |
| Return Type | boolean |
| Logic | A number is said to be an increasing number if no digit is exceeded by the digit to its left.  For Example : 134468 is an increasing number |

Exercise 12: Create a method to check if a number is a power of two or not

|  |  |
| --- | --- |
| Method Name | checkNumber |
| Method Description | Checks if the entered number is a power of two or not |
| Argument | int n |
| Return Type | boolean |
| Logic | Check if the input is a power of two.  Ex: 8 is a power of 2 |

**Lab2 : Inheritance and Polymorphism**

Using an inheritance hierarchy, design a Java program to model items at a library (books, journal articles, videos and CDs.) Have an abstract superclass called Item and include common information that the library must have for every item (such as unique identification number, title, and number of copies). No actual objects of type Item will be created - each actual item will be an object of a (non-abstract) subclass. Place item-type-specific behavior in subclasses (such as a video's year of release, a CD's musical genre, or a book's author).  
More in detail:

1. Implement an abstract superclass called Item and define all common operations on this class (constructors, getters, setters, equals, toString, print, checkIn, checkOut, addItem, etc). Have private data for: identification number, title, and number of copies.

2. Implement an abstract subclass of Item named WrittenItem and define all common operations on this class. Added private data for author.

3. Implement 2 subclasses of WrittenItem: Book and JournalPaper.

3.1. Class Book: no new private data. When needed, override/overload methods from the superclass.  
3.2. Class JournalPaper: added private data for year published. When needed, override/overload methods from the superclass.

4. Implement another abstract subclass of Item named MediaItem and define all common operations on this class. Added private data for runtime (integer).

5. Implement 2 subclasses of MediaItem: Video and CD.

5.1. Class Video: added private data for director, genre and year released. When needed, override/overload methods from the superclass.  
5.2. Class CD: added private data for artist and genre. When needed, override/overload methods from the superclass.

Write the definitions of these classes and a client program (your choice!) showing them in use.