Day-1

1.1 Write a simple program that prints:

Welcome to “PG-DAC J2SE”

Starting my first J2SE Lab Session

1.2 Write a program that prints all numbers starting from a specified number to another number specified for each looping construct.

- Count from 0 to 21 using a *for* loop incrementing by 1 for each iteration
- Print from 9.3 to 10.4 with 0.1 increment using *for-each* loop

[declare - double d[]={9.3,9.4,9.5,9.6,9.7,9.8,9.9,10.0,10.1,10.2,10.3,10.4}]

- Count from 99 to 35 using *while* loop decrementing by 2 for each iteration
- Count from 1 to 400 using *do-while* loop incrementing by product of 2 for each iteration

1.3 Implement a class Employee. An employee has a name and salary.

- Write a default constructor.
- Write a constructor with parameters(name and salary).
- Write a member method printEmpDetails() which prints name and salary.

Implement another class EmployeeDemo which contains the main() and does the following:

- Creates Employee object e with the following details
  - Name : Harry Smith
  - Salary : 10000
- Prints employee “e1” details and employee “e2” details using printEmpDetails() constructed using the no parameter constructor and the parameterized constructor respectively.

1.4 Write a program to determine the sum of all positive numbers less than 100 which are divisible by 3 or 5.

1.5 Write a program which prints the following pattern as output:

```
*   *   *   *   *   *   *
*   *   *   *   *   *   *
*   *                   *   *
*   *                   *   *
*   *                   *   *
*   *   *   *   *   *   *   *
*   *   *   *   *   *   *   *
```

1.6 Write a program that displays the list of all positive integer palindromes which are less than 1000.

1.7 List all the numbers less than 10000 which are equal to the sum of the factorial of all its digits.
For Example: 1!+4!+5! = 1+24+120 = 145

1.8 Write a program to list all the prime numbers less than 1000.

1.9 Write a program to compute the sum of squares of all even numbers equal to or less than 20 ?

$2^2 + 4^2 + \ldots\ldots\ldots\ldots\ldots\ldots 20^2 = ?$

Day-2

2.1 Write a program that accepts two numbers as input and finds the Least Common Multiple[LCM] of those numbers.

Sample Inputs	Sample Outputs
16 28	112
4 6	12

2.2 Write a program which takes input as number of 50 paise coins, number of 25 paise coins, number of 10 paise coins and outputs the total amount of money.

2.3 Write a program that takes as input the elements of 2 square matrices and determines the sum of the matrices.

Sample Inputs	Sample Outputs
2 [This is the size of the square matrices] 1 2 3 4 [elements of the 1st square matrix] 5 6 7 8 [elements of the 2nd square matrix]	6 8 10 12
3 [This is the size of the square matrices] 1 2 3 4 5 6 7 8 9 [elements of the 1st square matrix] 9 8 7 6 5 4 3 2 1 [elements of the 2nd square matrix]	10 10 10 10 10 10 10 10 10

2.4 Write a program that displays a text menu as shown below. The user is prompted to enter his choice of operation from the text menu followed by two integers. Based on the entered choice, an operation will be performed on the two integers. [infinite loop which terminates when “9” is input, use switch-case]

Sample Input	Sample Output
USER MENU ========== 1. Sum 2. Difference 3. Product 4. Average 5. Maximum (larger of two numbers) 6. Minimum (Smaller of two numbers) 9. Exit Enter the menu item number to select an operation 3 OK, you have entered 3 Enter the two numbers 10 20	You have selected 3. Product Numbers entered are 10 and 20 Result is 200

2.5 Write a program that accepts several integers from the user, one integer at one time, till a 0 is input. Add all the integers accepted, display the sum total after each accepted input. Implement using *while* loop.

Sample Input	Sample Output
I will add up the numbers you give me. Number: 6 Number: 9 Number: -3 Number: 2 Number: 0	The total so far is 6 The total so far is 15 The total so far is 12 The total so far is 14 The Grand total is 14

2.6 Write a program which displays a different message depending on the age given. Here are the possible responses:

- age is less than 16, say "You can't drive."
- age is less than 18, say "You can't vote."
- age is less than 25, say "You can't rent a car."
- age is 25 or over, say "You can do anything that's legal."

Remember that a person who is under 16 will display *three* messages, one for being under 16, one for also being under 18, and one for also being under 25. Implement using *if-else* construct.

Sample Input	Sample Output
Hey, what's your name? Harry Ok, Harry, how old are you? 17	 You can't vote, Harry. You can't rent a car, Harry.

2.7 Write a program that prompts user for radius and prints area and circumference of the circle along with radius. [Area: PI*R*R; Circumference: 2*PI*R]. Define a java constant for PI=3.14.

Sample Input	Sample Output
Enter radius of circle (in cm): 10	Radius(in cm): 10 Area: 314 cm^2 Circumference: 62.8 cm

2.8 Write a program that implements overloaded methods for each statement given below:

- Find area of either rectangle or square. [func(int, int) and func(int)]
- Find sum of two numbers(int or double). [calc(int, int) and calc(double, double)]

2.9 Complete the following code snippet by adding a ***continue*** statement(with label) and a ***break*** statement(with label) in appropriate places. Finally, the below code should search for a substring in a given string. Here the given string is *`Look for a substring in me,`* substring to be searched is *`sub`* and the label-name is ***test***.

```
class ContinueWithLabelDemo {
       public static void main(String[] args) {

             String searchMe = "Look for a substring in me";
             String substring = "sub";
             boolean foundIt = false;
           int max = searchMe.length() -  substring.length();

      test:
            for (int i = 0; i <= max; i++) {
          int n = substring.length();
          int j = i;
          int k = 0;
          while (n-- != 0) {
             if (searchMe.charAt(j++) != substring.charAt(k++)) {
             }
          }
          foundIt = true;
       }
       System.out.println(foundIt ? "Found it" : "Didn't find it");
```

```
        }
    }
```

Note:

- The unlabeled *continue* statement skips the current iteration of the innermost *for, while*, or *do-while* loop and evaluates the boolean expression that controls the loop. A labeled continue statement skips the current iteration of an outer loop marked with the given label.
- An unlabeled break statement terminates the innermost switch, for, while, or do-while statement, but a labeled break terminates an outer statement. The break statement terminates the labeled statement; it does not transfer the flow of control to the label. Control flow is transferred to the statement immediately following the labeled (terminated) statement.

2.10 Write a program that reads an unspecified number of integer arguments from the command line and adds them together to print the result.

For example, suppose that you enter the following to run your java program *Adder.java*:

java Adder 1 3 2 10

The program should display 16, which is the addition of 1, 3, 2 and 10. The program should display an error message if the user enters only one argument.

2.11 Implement a class EmployeeDemo which contains

- an inner class Employee. An employee has a name and salary. Write a default constructor and parametrised constructor with parameters(name and salary). Also, write a member method printEmpDetails() which prints name and salary.
- main() which creates an Employee object e with the following details
  - Name : Harry Smith
  - Salary : 10000
- using parameterized constructor and prints employee details using printEmpDetails().

Day-3

3.1 Create two java classes SCounter (with static integer *count* variable) and NSCounter (with integer *count* variable).

- Implement constructors which increments count variable by 1 and prints the same (in both classes). Use *this* keyword to refer to *count* in constructor.
- Write a CounterDemo class which contains the main() and tests the same by creating 3 objects each for SCounter and NSCounter classes.
- Write a static block which prints
  Static block invoked in CounterDemo in CounterDemo Class.
  Static block invoked in SCounter in SCounter Class.

Compile and run the program to check the same.

3.2 Following is a list of keywords. Create a class diagram showing the appropriate inheritance hierarchy. Make the required classes abstract wherever necessary.

Lion
Dog
Canine
Cat
Feline
Hippo
Tiger
Animal
Wolf

Place the following animal traits in appropriate classes obtained from the above inheritance hierarchy.

makeNoise()	*eat()*	*sleep()*
roam()	*food*	*hunger_level*

Note: Canine means dog family and they always roam in packs/groups. Feline means cat family and they tend to avoid others of their kind while roaming.

3.3 Write a program which models a library which contains many kinds of items:

Printed: Books, Journals, Magazines, and Documents
Multimedia: CDs, and DVDs.

Every item in the library must have an ID number and title. Every printed item must have a number of pages, and every multimedia item must have a length, in seconds. Define the classes Item, Printed and Multimedia, making sure they have the appropriate relationships in the class hierarchy, and the appropriate private fields.
Create constructors for the three classes, using super() where necessary.
Create toString() methods for each of the three classes.
For an item, the string should have its id followed by its title.
For printed and multimedia items, the id and title should be followed by the number of pages or running length with the appropriate unit (pages or seconds).

Sample Output:

```
7985 Alice in Wonderland (105 pages)
3565 In a Sentimental Mood (597 seconds)
```

Remember that the id and title fields in the Item class are private, so you'll need to call the toString() method belonging to the Item class for the other two toString() methods.

3.4 Create a class *OuterClass* with the following:

- a private integer variable outerX with value 10
- printOuterX()
- sayHello() with a local inner class named LocalInnerClass and an anonymous inner class.

The LocalInnerClass is as shown below:

```
class LocalInnerClass {
      public void greetInLocal() {
            System.out.println("Hello World in LocalInnerClass!!");
      }
      public void printXInLocal() {
            System.out.println("outerX :" + outerX + " in LocalInnerClass");
      }
}
```

The anonymous inner class AnonymousInnerClass as shown below:

```
AnonymousInnerClass hello = new AnonymousInnerClass(){
public void printHelloWorld() {
      System.out.println("Hello World in sayHello!!");
      System.out.println("outerX :" + outerX + " in AnonymousInnerClass");
}
```

The main() should make a call to OuterClass's sayHello() method.
Modify the sayHello() such that the following output is obtained:

```
Hello World in LocalInnerClass!!
outerX :10 in LocalInnerClass
Hello World in sayHello!!
outerX :10 in AnonymousInnerClass
```

Day-4

4.1 Define a String "`This is lab session on String Handling`". Do the following
Write a main method to

- Prints the length of the String
- Prints "String Handling"
- Prints "LAB SESSION"
- Prints "This is J2SE lab session on String Handling"
- Replaces 'J2SE' with 'J2SE PG-DAC' in above string.
- Generates and prints a random password by taking any one random letter from a-z,one random letter from A-Z and a random digit from 0-9.
- Define a String filePath and prints the extension,file-name and file-path.

Example: `filePath="/home/user/index.html"` should print

```
extension = html
file-name = index
file-path = /home/user
```

Use inbuilt methods in java.lang.String.
Use Random class of java.lang.Math for random password generation. [Hint : String is immutable]

4.2 Write a program that prompts the user to enter a String and prints whether it is a palindrome or not. Use StringBuilder class.

4.3 Write a program that defines following variables.

```
double a = -191.635;
double b = 43.74;
int c = 16, d = 45;
```

Use *java.lang.Math* class to find the following:

- Absolute value of a
- Ceiling of b
- Floor of b
- Maximum and Minimum of c and d
- A random integer number between 0 and 100(>=0 and <100)

Also, use static import of *Math* class in your code.

4.4 Write a class that implements the *CharSequence* interface found in the java.lang package. Your implementation should output the reverse of a string. Write a small *main* method to test your class.

4.5 An anagram is a word made by transposing the letters of another word. For example, "software" is an anagram of "swearoft". Write a program that figures out whether one string is an anagram of another string.

4.6 Create a Person Class which contains two member variables firstName and lastName.Write a program that prompts the user to enter details of several persons. Model the collection of persons using Vector.

- Print the vector containing person details.
- Remove a person from vector and print.

(Prompt the user to enter first name of the person to be deleted).

- Insert a new person at some position entered by user.
- Print details of first person.Replace it with "Harry Smith"
- Convert the vector of Person objects to an array of Person objects.

4.7 Write a JAVA Program which accepts text of words separated by commas and gives the output of words printed on separated lines with commas removed.

Sample Output:

```
Enter the string to parse
Monday,Tuesday,Wednesday,Thursday,Friday,Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Sunday
```

Day-5

5.1 Write a method:

public static int[] append(String[] a, String[] b)

that appends one array(of Strings) to another. For example, if

a = {qqq, www, eee} and b = {aaa, sss, ddd}

result = {qqq, www, eee, aaa, sss, ddd}

Hint: Use java.util.Arrays

Write a program that takes an array of words and does the following:

- Prints out any duplicate words
- Prints the number of distinct words
- Prints the list of the words with duplicates eliminated.

Sample Input	Sample Output
i came i saw i conquered	Duplicate words: i Distinct words: 4 List of the distinct words: [saw, came, conquered, i]

5.2 Write a program that gets an array of words and does the following:

- Prints each word and it's frequency
- Prints only the words
- Prints only the frequency of all words

Here's a sample run assuming the array of words as {i, came, i, saw, i, conquered}

saw : 1
came : 1
conquered : 1
i : 3

Note: Order doesn't matter in any of the above outputs

5.3 Create a Student class with name, id and age. Write a program that gets some students from user and then sorts by:

Ascending : order of Name (Use Comparable)
Descending : order of Name
Both age and name. (Use Comparator)

5.4 Write a program that creates a list, set and map of items. The user should not be allowed to do any modifications (add/delete) to these once they are created.

5.5 Implement a PhoneBook using HashMap.

Hint: Create the following classes:

Person	Class contains firstName and surName.
PhoneNumber	Class contains areaCode and number.
BookEntry	contains Person object and PhoneNumber object.
PhoneBook	contains a HashMap<Person,BookEntry>.
TestPhoneBook	Class contains main() to do the following:

The user should be prompted with the following menu:

Enter-1: to add a new phonebook entry
Enter-2: to find the number for a name
Enter-3: to find name for a number

Enter-9: to quit

5.6 Extend the above question so as to print the sorted list the entries in the phonebook. Hint: Store the entries in HashMap to a linked list and use Collections.sort().

5.7 Write a program that accepts an input string array/s and prompts the user to select a task from the menu as shown below. *[use the compareTo() method of the String class]*

```
Menu
====
1. Sort array
2. Perform binary search in array
3. Check if two arrays are equal
4. Replace/fill array with a default value
```

5.8 Write a program that gets an array of words and does the following:

- Prints each word and it's frequency
- Prints only the words
- Prints only the frequency of all words

Here's a sample run assuming the array of words as {i , came , i , saw , i , conquered}

came : 1
conquered : 1
i : 3
saw : 1

[came, conquered, i, saw]
[1, 1, 3, 1]

Note: Print in the same order as shown above.