MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

LABORATORY MANUAL

Department	:	Data Science and Computer Applications
Subject	:	Java Programming Laboratory (MCA 4262)
Semester and Branch :	:	II Semester M.C.A.
Name of the faculty	:	Dr. Dasharathraj K. Shetty and Dr. Vidya Rao
No. of contact hours/week	:	3

CONTENTS

Lab No.	Topics to be covered	Page No.
1	Simple programs using control structures, loops.	3
2	Array manipulation programs	6
3	Basics of Classes and Methods	7
4	Advanced concepts in Classes and Methods	8
5	String handling	9
6	Inheritance	10
7	Interfaces & abstract classes, packages	10
8	Exception handling, multithreaded programming	11
9	Collections	11
10	File handling	12
11	Swings, event handling & applications	12
12	End Semester Examination	12

Reference Books:

1. Herbert Schildt, *Java The Complete Reference*, 11th Edition, McGraw Hill, 2019.
2. Cay S. Horstmann. *Core Java: Volume I - Fundamentals*. 11th Edition, Pearson Education, 2018.
3. Cay S.Horstmann, *Core Java: Volume II – Advanced Features*, 11th Edition, Pearson Education, 2019.
4. Herbert Schildt and Dale Skrien, *Java Fundamentals*, Tata McGraw-Hill Education, 2015.

COURSE OBJECTIVES

Implementing concepts related to Basic Java Programming, Arrays, Strings, Inheritance, Exception handling, Packages, Collections, File handling and Swings.

Instructions to Students

1. **Attending the laboratory sessions**:

   a) You should be regular and come prepared for implementing the specified programs.
   b) In case you miss a session, it is your responsibility to complete the pending work before coming to the next session.
   c) You should have the laboratory manual with you during every laboratory session.

2. **Implementing the programs:**

   a) You should implement the programs individually.
   b) Prescribed text / reference books and class notes can be kept ready for reference, if required.
   c) The programs should meet the following criteria.
      - Programs should be interactive with appropriate messages for inputs and descriptive messages for outputs.
      - Programs should include input validation (data type, range error, etc.) and give appropriate error messages and suggest corrective actions.
      - Comments should be used to specify the statement of the problem and the purpose, inputs and outputs of every function.
      - Statements within the program should be properly indented.
      - Meaningful names should be used for variables and functions.
      - Constants and type definitions should be used wherever needed.
      - Meaningful inputs should be given during execution of programs.

Students SHOULD NOT:

1. Bring mobiles phones or any other electronic gadget to the lab
2. Leave the lab without prior permission from the faculty-in-charge.

Evaluation Plan

1. Continuous Evaluation	60%
Record: 4M, Viva: 6M and Execution: 10M Internal Marks: 3 * 20 = 60 Marks	
2. Lab Examination	40%
• End Semester Lab evaluation: 40 Marks, write Up: 15 Marks, execution: 25 Marks, • Total: 15+25 = 40 Marks Examination of 2 hours duration (Max. Marks: 40)	

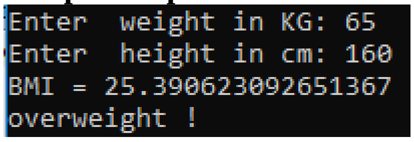
Week 1: Simple programs using control structures, loops

1. Body Mass Index (BMI) is a measure of health on weight. It can be calculated by taking your weight in kilograms and dividing by the square of your height in meters. The interpretation of BMI for different age group is as follows:

BMI	Interpretation
Below 18.5	Underweight
Between 18.5 and 24.9	Normal
Between 25 and 29.9	Overweight
30 or greater	Obese

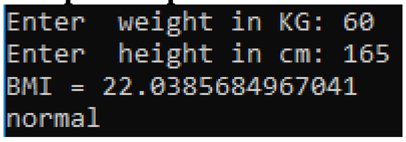
   Write a program that prompts the user to enter a weight in kilograms and height in centimetres. Compute the BMI and display the message accordingly.

Sample Output-1:



```
Enter  weight in KG: 65
Enter  height in cm: 160
BMI = 25.390623092651367
overweight !
```

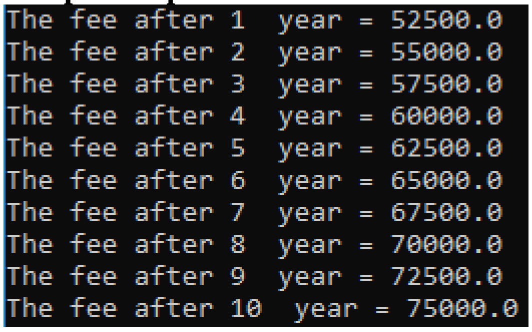
Sample Output-2:



```
Enter  weight in KG: 60
Enter  height in cm: 165
BMI = 22.0385684967041
normal
```

2. Suppose that the tuition for a university is Rs 50,000 this year and increases 5% every year. Write a program to compute and display the tuition fee for 10 consecutive years.

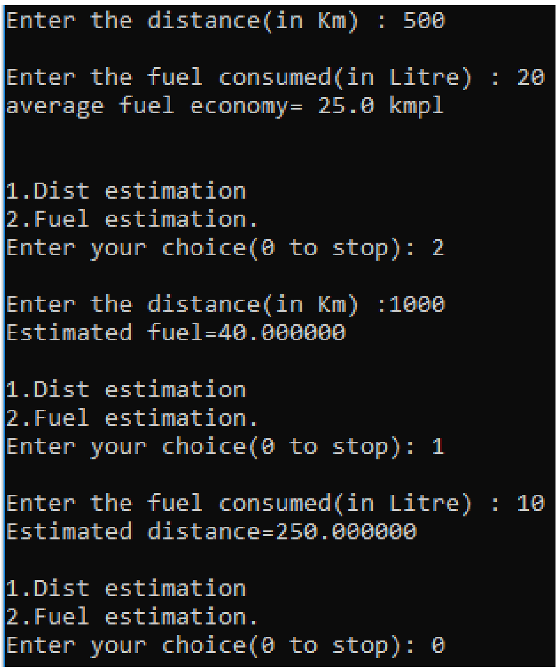
Sample Output:



```
The fee after 1  year = 52500.0
The fee after 2  year = 55000.0
The fee after 3  year = 57500.0
The fee after 4  year = 60000.0
The fee after 5  year = 62500.0
The fee after 6  year = 65000.0
The fee after 7  year = 67500.0
The fee after 8  year = 70000.0
The fee after 9  year = 72500.0
The fee after 10  year = 75000.0
```

3. Write a java program to accept distance travelled (in kilometre) and the quantity of fuel used (in litre) for a journey and determine the average fuel economy for that journey. Based on this, perform the following two operations for the user:
   - **Fuel estimation:** Estimate the fuel needed for the distance entered by the user.
   - **Distance estimation**: Estimate the distance that can be traveled for the quantity of fuel entered by the user.

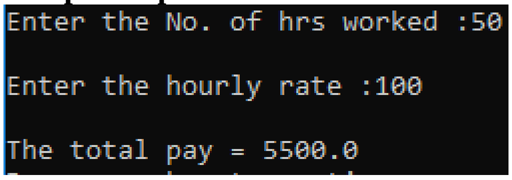
   Here is a sample output:



```
Enter the distance(in Km) : 500

Enter the fuel consumed(in Litre) : 20
average fuel economy= 25.0 kmpl


1.Dist estimation
2.Fuel estimation.
Enter your choice(0 to stop): 2

Enter the distance(in Km) :1000
Estimated fuel=40.000000

1.Dist estimation
2.Fuel estimation.
Enter your choice(0 to stop): 1

Enter the fuel consumed(in Litre) : 10
Estimated distance=250.000000

1.Dist estimation
2.Fuel estimation.
Enter your choice(0 to stop): 0
```

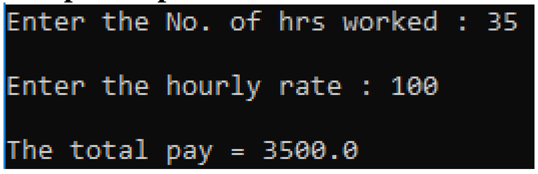
4. Develop a Java application to accept number of hours worked, hourly rate and determines the gross pay for an employee according to the following criteria:
   The company pays straight time for the first 40 hours worked by each employee and time and a half for all hours worked in excess of 40 hours.

Sample output-1:



```
Enter the No. of hrs worked :50

Enter the hourly rate :100

The total pay = 5500.0
```

Sample output-2:

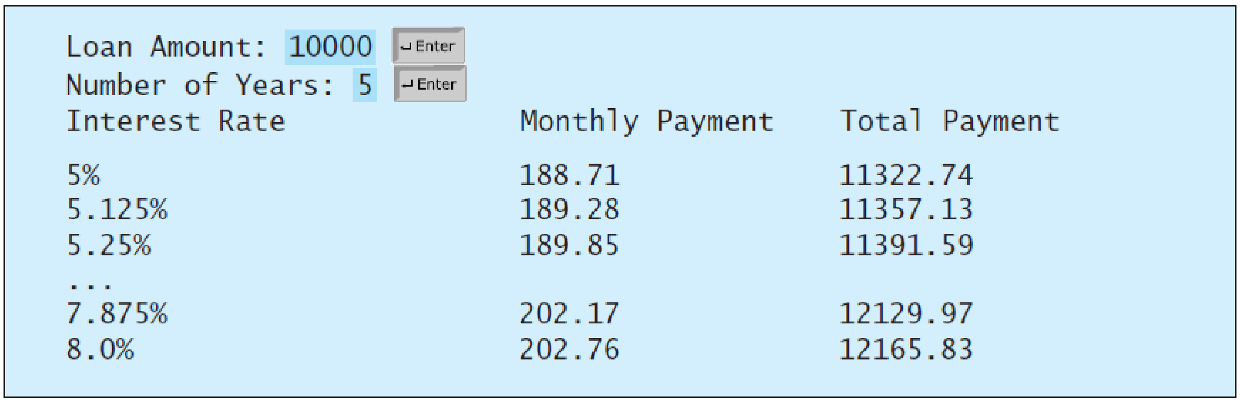


```
Enter the No. of hrs worked : 35

Enter the hourly rate : 100

The total pay = 3500.0
```

5. Write a program that lets the user enter the loan amount and loan period in number of years and displays the monthly and total payments for each interest rate starting from 5% to 8%, with an increment of 1/8.

$$monthlyPayment = \frac{loanAmount \times monthlyInterestRate}{1 - \dfrac{1}{(1 + monthlyInterestRate)^{numberOfYears \times 12}}}$$

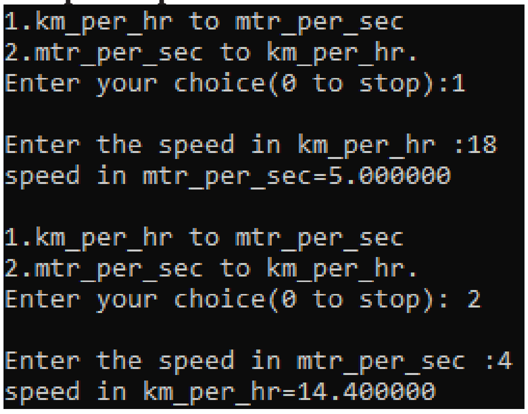
Where, monthlyInterestRate = annualInterestRate/12.
Here is a sample run:



```
Loan Amount: 10000
Number of Years: 5
Interest Rate              Monthly Payment    Total Payment

5%                         188.71             11322.74
5.125%                     189.28             11357.13
5.25%                      189.85             11391.59
...
7.875%                     202.17             12129.97
8.0%                       202.76             12165.83
```

6. Write a java program to convert the input speed from kilometre per hour to meters per second and vice versa. Hint: 1 Kmph = 1000/3600 mps

Sample output:



```
1.km_per_hr to mtr_per_sec
2.mtr_per_sec to km_per_hr.
Enter your choice(0 to stop):1

Enter the speed in km_per_hr :18
speed in mtr_per_sec=5.000000

1.km_per_hr to mtr_per_sec
2.mtr_per_sec to km_per_hr.
Enter your choice(0 to stop): 2

Enter the speed in mtr_per_sec :4
speed in km_per_hr=14.400000
```

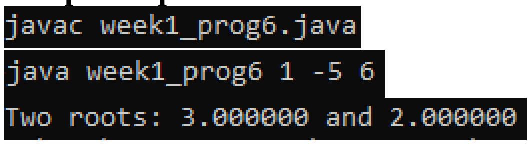
Additional Exercises:

1. Write a java program to add two dates input by the user.
2. Write a program to find the LCM and GCD of two numbers.
3. The two roots of a quadratic equation $ax^2 + bx + c = 0$ can be obtained using the following formula:

$$r_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \qquad r_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

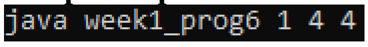
   b^2 - 4ac is called the discriminant of the quadratic equation. If it is positive, the equation has two real roots. If it is zero, the equation has one root. If it is negative, the equation has no real roots.
   Write a program that reads values for a, b, and c and displays the result based on the discriminant. The coefficients a, b and c must be passed as command-line arguments. If the discriminant is positive, display two roots. If the discriminant is 0, display one root. Otherwise, display "The equation has no real roots".

   Sample output-1:

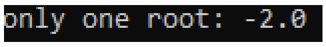


```
javac week1_prog6.java
java week1_prog6 1 -5 6
Two roots: 3.000000 and 2.000000
```

   Sample output-2:

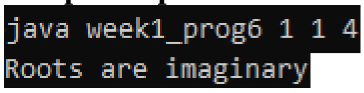


```
java week1_prog6 1 4 4
```



```
only one root: -2.0
```

Sample output-3:

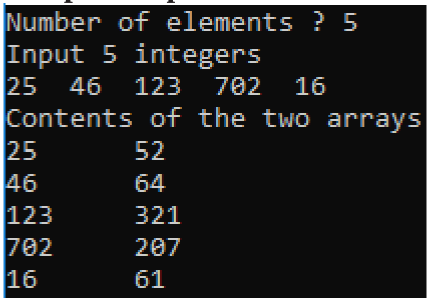


```
java week1_prog6 1 1 4
Roots are imaginary
```

Week 2: Array manipulation programs

1. Input an array of *n* integers. Reverse all elements of the array and store them in another array.

Sample Output:



```
Number of elements ? 5
Input 5 integers
25  46  123  702  16
Contents of the two arrays
25      52
46      64
123     321
702     207
16      61
```

2. Write a program to read an array of n distinct numbers, and display all the pairs (x,y) such that x>y.

Sample output:

Input: {2,1, 5, 3}

Output: { (2,1), (5,2), (5,1), (5,3), (3,2), (3,1) }

3. Write a java program to find the largest and smallest element in a two-dimensional array. Display the row & column indices of the largest and the smallest element in the given two-dimensional array.

Sample output:

```
Enter the number of rows and columns of the array: 3   4
Enter the array:
 23.5      35       2        10
 4.5       3        45       3.5
 35        44       5.5      9.6

The largest element = 45, location = (1, 2)
The smallest element = 2, location = (0, 2)
```

Additional Exercises:

1. Write a java program to copy all the unique elements from the given 2-D array into an 1-D array.

2. Write a program to subtract each element of the given array from the mean value of the same array.
3. Given a matrix containing only binary numbers. Sort each row based on the total number of one's present in the row.

Example:

$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Week 3: Basics of classes and Methods

1. A shop provides discount to customers based on their total bill amount as follows:

Total bill amount	**Discount**
<500	No discount
500-1000	10%
1000-2000	12%
Above 2000	15%

Create a class called **Shop** with data members for storing item id, item name, quantity, unit price and total bill amount. Implement the above using a menu-driven program.

2. Create a class named "Employee" to store employee information with data: employee code, employee name, Basic, HRA, IT, PF, net salary and gross salary. The net salary and gross salary are calculated as follows:
   Gross salary = Basic + HRA
   Net salary = gross salary – (IT+PF)
   where HRA is 7.5% of basic, IT = Rs.200, PF = 12% of basic.
   Write appropriate methods to read the data, calculate net salary and display the net salary.

3. Create a class called Box with width, height and depth as data members. Include a member function to find the volume of the box. Use parameterized constructors to initialize the objects.

Additional Exercise:

Define a class **IntArr** which holds an array of integers as its data members. Provide the following methods:

- A ***default constructor.***
- A ***parameterized constructor*** which initializes the array of the object.
- A method called ***display*** to display the array contents.
- A method called ***search*** to search for an element in the array.
- A method called ***compare*** which compares 2 **IntArr** objects for equality.

Week 4: Advanced concepts in classes and methods

1. A telephone company contains the records of all customer payment details of a "Customer" like customer_id, name, phone number, Bill number, Number of calls and bill amount. The monthly telephone bills are calculated as per the following criteria:

   Minimum Rs. 100 for the first 100 calls.
   + Rs. 0.60 per call for next 50 calls.
   + Rs. 0.50 per call for next 50 calls.
   + Rs. 0.40 per call for any call beyond 200 calls.

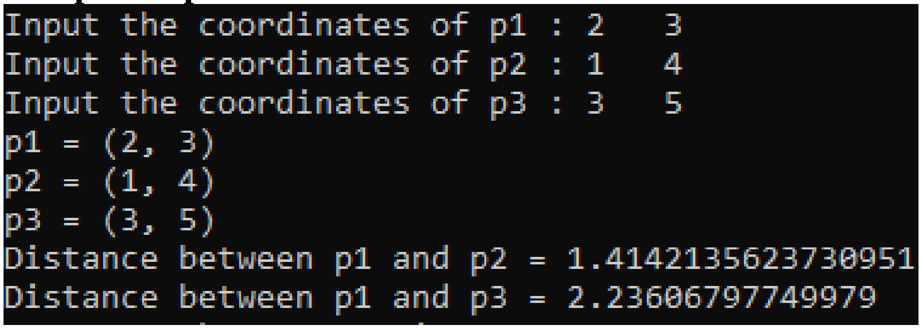
   Create an array of objects to process the details of 'n' customers. Use appropriate constructors, and member functions to accept and display details of a customer. Create a function Calculate_bill_amt() to calculate bill amount.

2. Consider a class called Points with *x-coordinate* and *y-coordinate* as its data members. Create three objects P1, P2 and P3, which represent three points in a plane.
   - Calculate and the distance between P1 and P2 (by passing the coordinates of one point as arguments).
   - Calculate and the distance between P1 and P3 (by passing one point itself as argument).

   Note: The distance between two points P(x_1, y_1) and Q(x_2, y_2) is calculated as:

$$d(P,Q) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Sample output:



```
Input the coordinates of p1 : 2   3
Input the coordinates of p2 : 1   4
Input the coordinates of p3 : 3   5
p1 = (2, 3)
p2 = (1, 4)
p3 = (3, 5)
Distance between p1 and p2 = 1.4142135623730951
Distance between p1 and p3 = 2.23606797749979
```

3. Create a class called Stack with necessary fields and constructors. Use static methods for push and pop operations.

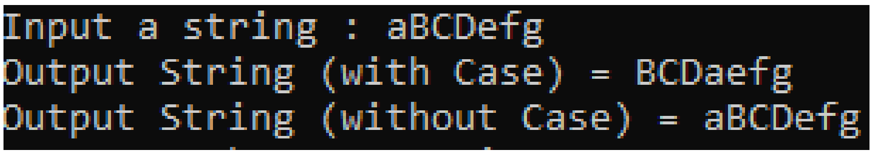
Week 5: String handling

1. Input a string. Reverse it. Check whether it is a palindrome or not.

Note: Write a separate function for reversing the string.

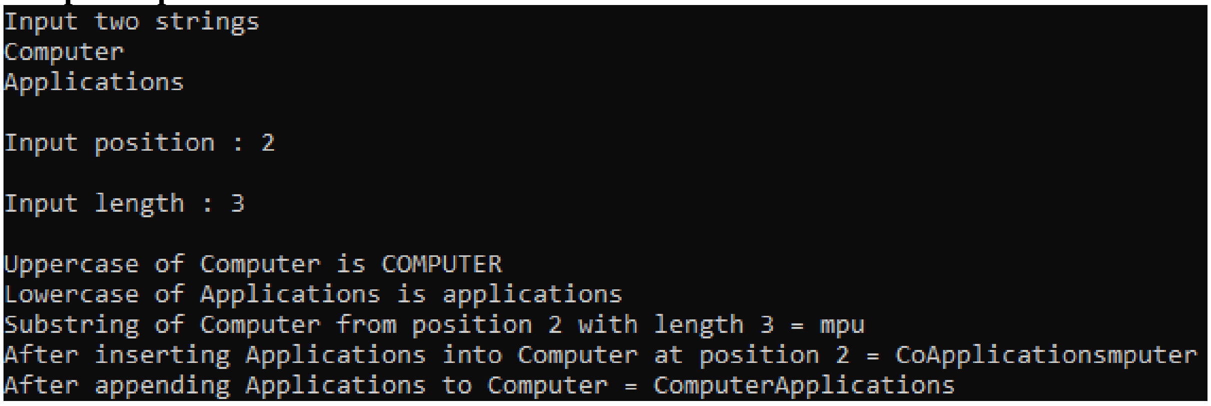
2. Input a string. Rearrange the characters in it in alphabetic order (E.g., COMPUTER should be written in the form of CEMOPRTU).Rearranging should be done for the two cases:
   • Considering the upper/lower case of alphabets
   • Without considering the upper/lower case of alphabets

Sample output:


```
Input a string : aBCDefg
Output String (with Case) = BCDaefg
Output String (without Case) = aBCDefg
```

3. Input two strings (S1 and S2) and two integers (N1 and N2). Perform the following string operations.
   - Display the upper-case equivalent of S1.
   - Display the lower-case equivalent of S2.
   - Extract a substring of length N2 from S1 starting at position N1.
   - Insert S2 into S1 at position N1.
   - Append S2 to S1.

Sample output:


```
Input two strings
Computer
Applications

Input position : 2

Input length : 3

Uppercase of Computer is COMPUTER
Lowercase of Applications is applications
Substring of Computer from position 2 with length 3 = mpu
After inserting Applications into Computer at position 2 = CoApplicationsmputer
After appending Applications to Computer = ComputerApplications
```

Additional Exercise:

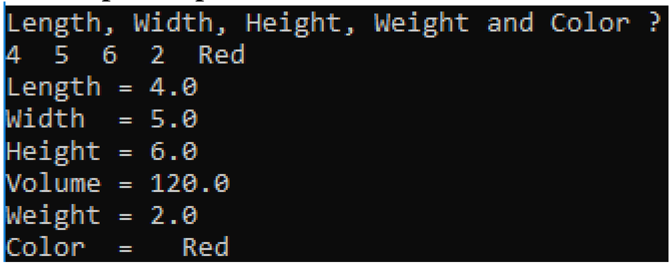
1. Write a java program to print the given number in words.

Example- Input number: 123, Output: One hundred Twenty Three.

Week 6: Inheritance

1. Create a class called **Box** having length, width, and height as attributes. Use a method along with all possible constructors to calculate volume. Create a derived class with additional data members namely weight and color. Include a method to display the volume and all the data members used in both the classes.

   Sample output:


```
Length, Width, Height, Weight and Color ?
4  5  6  2  Red
Length = 4.0
Width  = 5.0
Height = 6.0
Volume = 120.0
Weight = 2.0
Color  =   Red
```

2. Create three classes as per the following specifications:
   • Student: Data members are Register Number, Name, Course and Semester.
   • Exam (derived from class Student): Data members are the marks scored in three subjects.
   • Result (derived from class Exam): Data members are Total Marks and Grade.

Implement get_data() and display() methods using the concept of method overriding.

Week 7: Interfaces & abstract classes, packages

1. Create an abstract class Figure with abstract method area and two integer dimensions. Create three more classes Rectangle, Triangle and Square which extend Figure and implement the area method. Show how the area can be computed dynamically during run time for Rectangle, Square and Triangle to achieve dynamic polymorphism. (Use the reference of Figure class to call the three different area methods)

2. Create a class that performs conversion of temperatures – from Celsius to Fahrenheit and vice versa. Include the following two packages:
   • CToF with a class CelsToFahr containing a method called ConvertCToF (double C)
   • FToC with a class FahrToCels containing a method called ConvertFToC (double F).
   Use a menu to select the type of conversion, input the relevant data, calculate and display the result.

3. Create a class called Employee with data members: name, id, designation, salary. Write appropriate constructors and methods for initializing the object. Implement Cloneable interface to create a copy of the Employee object.

Week 8: Exception handling, multithreaded programs

1. Write a program to enter student's Name, Roll Number and Marks in three subjects, find the percentage, grade, and handle NumberFormatException. Validate marks to be in the range 0-100 by creating an exception called MarkOutofBoundsException and throw it.

2. Write a program to read two 1-D arrays of numbers, and perform element-wise division on these arrays. Handle arithmetic exception & array index out of bound exception.

3. Input an integer *n*. Calculate and display the squares and cubes of all integers from 1 to n using two different threads.

4. Given a 2-D array of size 3×3, write a multithreaded program to compute the sum of all the elements. Use three separate threads for computing the sum of each row of the matrix.

Additional Exercise:

1. Write a multithreaded java program to do the following.
   a. Using first thread generate Fibonacci numbers within 500 and store.
   b. Using second thread generate Prime numbers within 500 and store.
   c. The main thread should read and display the numbers, which are common.

2. Write a program to explain the multithreading with the use of multiplication tables. Three threads must be defined. Each one must create one multiplication table; they are 5 table, 7 table and 13 table.

Week 9: Collections

1. Create and initialize a vector with first five odd integers. Using a menu, perform the following operations on this vector:
    - Insert an element at a specified position
    - Insert an element at the end
    - Delete an element
    - Display the contents

2. Given a class called Employee, with fields for storing emp no, name, designation, age, salary. Create an arraylist to perform the following operations (use menu-driven approach):
    - Insert an object into the arraylist
    - Delete an object from the arraylist
    - Display the contents of the arraylist.

3. Input five strings and store them in an ArrayList. Sort it in alphabetic order. Display the list in the forward direction (using an iterator) and in both directions (using a listIterator).

Week 10: File handling

1. Create a file containing a set of strings. Read the file, sort the strings and store them in another file.
2. Write a program to copy the contents of one file to another file using FileReader and FileWriter classes.
3. Create a class called *Bank* with instance fields for representing each customer's name, age, account number and balance amount. Store the account details of 'n' customers in a file *account_info.txt* using serialization technique. Display the details of all the customers by reading the information from the file.

Additional Exercise:

1. Write a program to copy all files from source directory onto destination directory. Read source and destination directory names from keyboard.

Week 11: Swings, event handling & applications

1. Write a java application using swings to display a window for accepting two numbers and perform addition, subtraction, multiplication and division. Create necessary controls such as textboxes, buttons.
2. Implement the previous question, using a combo box to list out the options: addition, subtraction, multiplication and division for selecting the operation.
3. Create a java application to display a login window and navigate to a new window with welcome message on successful authentication.

4. Develop a swing application to perform currency conversion from Rupee to Dollar.

Additional Exercise:

1. Design a simple calculator using swings to show the working for simple arithmetic operations.
2. Write java application to implement insert, display and search operations on a STUDENT table.

Week 12: End Semester Examination

****