PRACTICAL LAB FILE

SEMESTER – 3rd

Lab: CABSMJ3P03

Bachelor of Science
(Computer Application)

Aligarh Muslim University
(DEPARTMENT OF COMPUTER SCIENCE)

SUBMITTED BY:
NAME: Mohd Shaheer Faisal
FACULTY NO: 24CABSA508
ENROLLMENT NO: GQ2030

SUBMITTED TO:
Dr. Asif Irshad Khan
Dr. Mohammad Nadeem
Dr. Ashraf Siddiqui
Dr. Mohammad Luqman
Mr. Jamaluddin
Mr. Md. Ozaif

GitHub Repository : https://github.com/Stephen047/CABSMJ3PO3

INDEX

Week No.		Problems with descriptions	Page No.	Signature of the teacher with date
1		Write a short report (1–2 pages) on “Resources for Learning and Practicing Java Programming”.		
2	1#	What are the softwares that helps to run java programs?		
	2#	What is JDK and JRE?		
	3#	What is eclipse IDE?		
	4#	How to run the java program in eclipse/NetBeans IDE?		
3	1#	Write a java program to add the two numbers.		
	2#	Write a java program to multiply two floating numbers.		
	3#	Write a java program to display a cube of a number.		
	4#	Write a Java program that takes three numbers as input to calculate and print the average of the numbers.		
	5#	Write a Java program to compute the distance between two points.		
	6#	Write a Java program to swap two numbers using a temporary variable.		
	7#	Write a Java program to calculate the area of a rectangle given its length and breadth.		
	8#	Write a Java program to convert temperature from Celsius to Fahrenheit.		
	9#	Write a Java program that takes two integer inputs and computes their remainder and quotient.		
	10#	Write a Java program to find the circumference of a circle given its radius.		
4	1#	Write a java program to check whether the given number is odd or even.		
	2#	Write a java program to find the largest number among the three numbers.		
	3#	Write a Java program that takes a number as input and prints its multiplication table upto 10.		
	4#	Write a Java program to calculate the sum of following series: 1 + 2 + 3 + 4 + ... + N		
	5#	Write a Java program to take a number, divide it by 2 and print the result until the number becomes less than 10.		
	6#	Write a Java program to check whether a given character is a vowel or consonant.		

INDEX

	7#	Write a Java program to find the smallest number among four given numbers.		
	8#	Write a Java program to calculate the sum of all even numbers from 1 up to a given number N.		
	9#	Write a Java program to check whether a given year is a leap year or not.		
	10#	Write a Java program that takes a number as input and prints all its factors.		
5	1#	Write a Java program to insert 10, 20, 30 ....in an array and display them.		
	2#	Write a Java program to calculate the sum of all the array elements.		
	3#	Write a java program to print the following pattern: 1 12 123 1234 12345		
	4#	Write a java program to find the sum of following series where n is input by the user: 1 + 1/2 + 1/3 + 1/4 + ... + 1/n		
	5#	Write a Java program and compute the sum of the digits of an integer.		
	6#	Write a Java program to calculate the factorial of a number.		
	7#	Write a Java program to find the largest element in a given integer array.		
	8#	Write a Java program to reverse the digits of a given integer.		
	9#	Write a Java program to check if a given number is a palindrome or not.		
	10#	Write a Java program to convert a decimal number into Hexadecimal number and vice-versa.		
	11#	Write a Java program to print the following pattern: * ** *** ** *		
6	1#	Write a Java program to print the odd numbers from 1 to 99.		
	2#	Write a Java program to check whether a number is prime or not.		
	3#	Write a Java program to swap the first and last elements of an array.		
	4#	Write a Java program to find the maximum and minimum among array elements.		

INDEX

	5#	Write a Java program to print all prime numbers between 0 to 100.		
	6#	Write a Java program to implement linear search.		
	7#	Write a Java program to print all prime numbers between 0 to 100.		
	8#	Write a Java program to find the second largest element in an array.		
	9#	Write a program to implement Fibonacci series up to N terms (0,1,1,2,3,5....).		
	10#	Write a Java program to reverse all elements of an array.		
	11#	Write a Java program to find the frequency of each character in a given string.		
7	1#	Write a Java function to implement binary search.		
	2#	Write a Java function to arrange the elements of an array in ascending order (Sorting).		
	3#	Write a program to reverse a given string.		
	4#	Write a program to check whether a given string is palindrome or not.		
	5#	Write a program to implement factorial of a number through recursion.		
	6#	Write a program to implement Fibonacci series of a number with and without recursion.		
	7#	Write a Java function to find the greatest common divisor (GCD) of two numbers with and without using recursion.		
	8#	Write a program to check whether two strings are anagrams of each other (“listen” and “silent” are anagrams).		
	9#	Implement quick sort using recursion.		
8	1#	Create a class FRUIT which has data members colour, taste and price. Also create a method display() which will print values of FRUIT object. Create three objects of FRUIT class and call their display() methods.		
	2#	Create a class FRUIT which has data members colour, taste and price. It has a method setDetails() which will set the values of colour, taste and price. Also create a method display() which will print values of FRUIT object.		
	3#	In previous question, set the values of using colour, taste and price using Constructor.		
	4#	Add one-argument constructor and two-argument constructor in addition to default constructor in FRUIT class.		
	5#	Use the concept of constructor-chaining in the previous question using this().		

INDEX

	6#	Create a class CAR with the following details: Data members: model, color, price. Member methods: setDetails() – to set values of all data members using setters. getDetails() – to get values of all data members using getters. display() – to print all details of the car. Requirements: 1. Implement default constructor to initialize default values. 2. Implement a parameterized constructor (with one argument) to set only model. 3. Implement another parameterized constructor (with two arguments) to set model and colour. 4. Use constructor chaining to reduce code redundancy. 5. Create three objects of CAR class using: a. Default constructor b. One-argument constructor c. Two-argument constructor 6. Set price for each object using the setDetails() method. 7. Call the display() method for each object.		
9	1#	Create a class Vehicle, write a method cost() in this class. Create two classes Bus and Train which have their own display() methods and inherit from Vehicle class. Create objects of Bus and Train class and call cost() and display() methods.		
	2#	Create class University which has data members- name and ranking. Create class Faculty that extends University class has data member- name and method- Details(). Create a new class Department which is derived from Faculty and has data member- name, chairman and method- Details() and Display() where Display() method calls Details() methods of both Faculty and Department class in its body. Create an object of Department class to Display() method and University ranking.		
	3#	Create class Account (Data members- Id, Account_holder_name, Address; Methodsdeposit(), withdraw()). Create two static methods in Account calculateSimpleInterest() and		

INDEX

		calculateCompoundInterest() and implement them.		
	4#	Create class Account (Data members- Id, Account_holder_name, Address; Methodsdeposit(), withdraw()). Declare deposit() and withdraw() as abstract methods. Declare Account class as abstract. (Create constructor in Account as well)		
	5#	Create two children of Account- Saving (Data Members- Min_balance; Methodsdisplay(), deposit(), withdraw()) and Current (Data Members- Max_withdrawl_limit; Methods- display(),deposit(), withdraw()) . Create constructors for both classes. Implementation of deposit() and withdraw() should be specific to Saving and Current class. Create objects of Saving and Current class and display them.		
	6#	Create a class Shape with a method area(). Create two derived classes Rectangle and Circle that extend Shape. Each class should override the area() method to calculate the area of the respective shape. Create objects of Rectangle and Circle and call their area() methods.		
	7#	Create a class Employee with data members: name, salary, and a method showDetails(). Create a class Manager that extends Employee with an additional data member department. Override the showDetails() method in Manager to display all details, including department. Create an object of Manager and call showDetails().		
	8#	Create an abstract class Appliance with data members brand, power and abstract methods turnOn() and turnOff(). Create two derived classes WashingMachine and Refrigerator that provide their own implementations of turnOn() and turnOff(). Create objects of WashingMachine and Refrigerator and call their methods.		
	9#	Create a class MathOperations with two static methods: findGCD(int a, int b) to calculate the greatest common divisor and findLCM(int a, int b) to calculate the least common multiple. Call these methods without creating an object of MathOperations.		

INDEX

	10#	Create a class Student with data members rollNo, name, marks. Add a static variable schoolName. Create static method changeSchoolName() to update schoolName. Demonstrate how the static variable is shared among all objects.		
10	1#	Create class Person (Data Member- name, phone). Create two member inner classes Address (Data Member- House_No, Street, City, State; Method- displayAddr()) and DateOfBirth (Data Member- Day, Month,Year; Method- displayDOB()). Display() is the method of Person class which will display name, address and date of birth of a Person object.		
	2#	Create class Edible. Within that define two static classes Fruit and Vegetable. Fruit class will have two methods- fruitDetails() is a static method and fruitPackaging() is a non-static method. Vegetable class also has similar methods - vegetableDetails() and vegetablePackaging(). Call all the four methods from main method.		
	3#	Create three different minMaxAdd() methods to calculate minimum, maximum and addition of integers, real numbers and characters.		
	4#	Create a class ObjectOriented which has methods- abstraction(), polymorphism() and inheritance(). Create a class JavaLanguage which inherits from ObjectOriented class and has its own methods- persistence() and interfaces(). Create an object of JavaLanguage class to access all of its own and parent's methods.		
	5#	In previous question, create a new class C++ which also inherits from ObjectOriented class and has its own methods- template() and friendFunction().Create an object of C++ class to access all of its own and parent's methods.		
	6#	Create class University which has data members- name and ranking. Create class Faculty that extends University class has data member- name and method- Details(). Create a new class Department which is derived from Faculty and has data member- name, chairman and method- Details() and Display() where Display() method calls Details() methods of both Faculty and Department class in its body. Create an object of Department class to Display() method and University ranking.		

INDEX

	7#	Create a class Employee (Data Members – empName, empId). Create two member inner classes: • Salary (Data Members – basic, hra, pf; Method – displaySalary() to print salary details). • JoiningDate (Data Members – day, month, year; Method – displayJoiningDate() to print joining date). In the Employee class, create a method displayEmployee() that prints the employee's name, ID, salary details, and joining date.		
	8#	Create a class Shape with overloaded methods area(): • area(int side) – calculates area of a square. • area(int length, int breadth) – calculates area of a rectangle. • area(double radius) – calculates area of a circle.		
	9#	Create a class Vehicle with a method run(). Create subclasses Bike and Car that override the run() method. In the main() method, use a reference of Vehicle to call run() for objects of Bike and Car.		
11	1#	Create an interface Account having methods- deposit(), withdraw() and aboutBank() (abountBank() is a static method). Create two classes Saving and Current which implement the Account interface. Call the methods of Saving and Current classes in main method.		
	2#	In the previous question, create a new method in Account interface- takeLoan() (takeLoan() is a default method). takeLoan() method would be implemented by Saving class only. Call the methods of Saving and Current classes in main method.		
	3#	Create interfaces Bike and Scooty, both of which have two methods- offer() and details() (details() is default method). Create a new class BuySomething which implements both interfaces. To remove ambiguity, create a method details() in BuySomething class as well in which call the details() method of both interfaces. Call the methods of BuySomething class in main method.		

INDEX

	4#	Create two interfaces Printer and Scanner, both having methods connect() and details() (details() is a default method). Create a class MultiFunctionMachine that implements both interfaces. In MultiFunctionMachine, override the details() method to resolve ambiguity and call the details() methods of both interfaces. Call all methods of MultiFunctionMachine in the main() method.		
	5#	Create an interface Device with a method powerOn(). Create another interface SmartDevice that extends Device and adds a method connectWiFi(). Create a class SmartPhone that implements SmartDevice. Demonstrate calling both powerOn() and connectWiFi() using a SmartPhone object in the main() method.		
12	1#	Write a program that calls a method that throws an exception of type ArithmeticException in a for loop at an undesirable situation (such as divide by zero or taking square root of negative number). Catch the exception and display appropriate message. (Example of Unchecked Exception).		
	2#	Write a program of your choice where a Checked Exception occurs at third function but handled at the first calling function. Use both ways of managing Checked Exception i.e. using try-catch block and throws keyword.		
	3#	You are developing an online banking system where users can transfer money between accounts. If a user tries to withdraw more money than is available in their account, an InsufficientFundsException should be thrown.		
	4#	Create a user-defined exception InvalidAgeException when the age of a person is below 18 years. Use this exception at appropriate place.		
13	1#	Write a Java Program to create a new file.		
	2#	Write a Java Program to write into a file.		
	3#	Write a Java Program to copy one file into another file.		
	4#	Write a java program to find total no. of characters in a file.		
	5#	Write a java program to find total no. of lines in a file.		

INDEX

14		Write a Java program to: a. Connect with a database of your choice using JDBC API. b. Create an Employee table having employee id, age, name and salary. c. Insert five records into Employee table. d. d. Delete any two records.		

GitHub: https://github.com/Stephen047

LinkedIn: https://www.linkedin.com/in/shaheerfaisal/

WEEK 1

Resources for Learning and Practicing Java Programming

Java is a powerful, widely used programming language that is essential for software development, Android apps, web applications, and more. For learners at all levels, there is a wide range of high-quality resources available to learn and practice Java programming. This report outlines the most effective sources under six key categories.

1. Official Documentation and Websites

- Oracle Java Documentation (https://docs.oracle.com/en/java/)
  - This is the official documentation from Oracle, the main maintainer of Java. It includes comprehensive API references, detailed tutorials, technical guides, and Java SE downloads.

- OpenJDK (https://openjdk.org/)
  - OpenJDK is the open-source implementation of the Java Platform. It provides access to the source code of Java, reference implementations, and builds for developers interested in contributing to or learning from the core of Java.

2. Books and E-Books

- "Head First Java" by Kathy Sierra and Bert Bates
  - This beginner-friendly book uses a visually rich format and real-world analogies to explain Java concepts. It is excellent for building a strong foundation in object-oriented programming.

- "Java: The Complete Reference" by Herbert Schildt
  - A comprehensive guide that covers everything from basic syntax to advanced features. It is useful for both beginners and intermediate learners looking for a deep understanding of Java.

3. Online Learning Platforms

- Codecademy (https://www.codecademy.com/)
  - Offers interactive Java courses with hands-on exercises. Great for beginners who prefer a guided, step-by-step learning experience.

- Coursera (https://www.coursera.org/)
  - Features Java courses from top universities like Duke and the

University of Helsinki. Includes video lectures, assignments, and quizzes.

- Udemy (https://www.udemy.com/)
  - Provides a variety of Java courses tailored to different experience levels. Some popular courses include real-world projects and interview preparation.

4. Coding Practice Websites

- HackerRank (https://www.hackerrank.com/domains/tutorials/10-days-of-java)
  - Offers Java challenges and tutorials to help build problem-solving skills.

- LeetCode (https://leetcode.com/)
  - Focuses on algorithmic problems in Java, often used to prepare for technical job interviews.

- Codeforces (https://codeforces.com/)
  - Hosts regular contests and competitive programming challenges that can be solved using Java.

5. Community & Discussion Forums

- Stack Overflow (https://stackoverflow.com/questions/tagged/java)
  - A vast Q&A platform where developers ask and answer Java-related questions. Great for debugging and finding solutions to specific problems.

- Reddit r/java (https://www.reddit.com/r/java/)
  - A community for Java enthusiasts to discuss news, libraries, trends, and share learning resources.

6. My Preferred Resources

Personally, I find “Head First Java” and HackerRank to be the most helpful. The book makes learning Java engaging and beginner-friendly, especially with its analogies and visual approach. On the other hand, HackerRank offers practical coding problems that reinforce concepts and improve logical thinking. Additionally, I often refer to Oracle’s official documentation for clarity and accuracy when working with Java’s standard libraries.

WEEK 2

1. What are the softwares that help to run Java programs?

To run Java programs, you need software such as the **Java Development Kit** (JDK) and the **Java Runtime Environment** (JRE). In addition, many developers prefer using **Integrated Development Environments** (IDEs) such as *Eclipse, NetBeans, IntelliJ IDEA,* or *BlueJ*. These tools provide features like code editors, compilers, and debuggers, making it easier to write, compile, and execute Java code efficiently.

2. What is JDK and JRE?

The **Java Development Kit** (JDK) is a complete package required for developing Java applications. It contains essential tools such as the compiler (**javac**), a debugger, and various utilities needed to write and test programs.

The **Java Runtime Environment** (JRE) is included within the JDK but can also be installed separately. It contains only the components needed to run Java programs—such as the **Java Virtual Machine** (JVM) and core libraries—without the development tools.

3. What is Eclipse IDE?

Eclipse IDE is a widely used, open-source integrated development environment designed primarily for Java programming, though it can support other languages through plugins. It offers a range of features including syntax highlighting, code completion, project organization, and integrated debugging. By combining a built-in compiler with powerful development tools, Eclipse streamlines the process of creating, testing, and running Java applications.

4. How to run a Java program in Eclipse/NetBeans IDE?

Open the IDE and create a new **Java Project.**

Add a Java class file with a **main()** method.

Right-click the file or project and select **Run** (or click the Run button). The output will appear in the console.

WEEK 3

1. Write a java program to add the two numbers.

week3_1.java

```java
public class week3_1 {
    public static void main(String[] args) {
        int a = 10, b = 20;
        int c = a+b;
        System.out.println("Sum : " +c);
    }
}

```

```
Sum : 30

Process finished with exit code 0
```

2. Write a java program to multiply two floating numbers.

week3_1.java week3_2.java

```java
public class week3_2 {
    public static void main(String[] args) {
        float a = 1.5F, b = 2.5F;
        float c = a*b;
        System.out.println("Product : " +c);
    }
}

```

```
Product : 3.75

Process finished with exit code 0
```

3. Write a java program to display a cube of a number.

WEEK 3

week3_1.java week3_2.java week3_3.java

```java
public class week3_3 {
    public static void main(String[] args) {
        int a = 5;
        System.out.println("Cube of " +a+ " = " + a*a*a);
    }
}
```

```
Cube of 5 = 125

Process finished with exit code 0
```

4. Write a Java program that takes three numbers as input to calculate and print the average of the numbers.

week3_1.java week3_2.java week3_3.java week3_4.java

```java
import java.util.Scanner;

public class week3_4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a, b, c;
        System.out.println("Enter three numbers :");
        a = sc.nextInt();
        b = sc.nextInt();
        c = sc.nextInt();
        double avg = (a+b+c)/3.0;
        System.out.println("Average = "+avg);
    }
}
```

```
Enter three numbers :
5
10
15
Average = 10.0
```

5. Write a Java program to compute the distance between two points.

WEEK 3

week3_1.java week3_2.java week3_3.java week3_4.java week3_5.java

```java
import java.util.Scanner;

public class week3_5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double x1, y1;
        double x2, y2;
        System.out.println("Enter x1 and y1 :");
        x1 = sc.nextDouble();
        y1 = sc.nextDouble();
        System.out.println("Enter x2 and y2 :");
        x2 = sc.nextDouble();
        y2 = sc.nextDouble();
        double dis = Math.sqrt(Math.pow(x2-x1, 2) + Math.pow(y2-y1, 2));
        System.out.printf("Distance between %.2f, %.2f and %.2f, %.2f is %.2f",x1,y1,x2,y2,dis);
    }
}
```

```
Enter x1 and y1 :
-3
2
Enter x2 and y2 :
3
5
Distance between -3.00, 2.00 and 3.00, 5.00 is 6.71
```

6. Write a Java program to swap two numbers using a temporary variable.

week3_6.java

```java
public class week3_6 {
    public static void main(String[] args) {
        int a = 5, b = 10;
        System.out.printf("a = %d and b = %d\n", a, b);
        int temp = a;
        a = b; b = temp;
        System.out.printf("a = %d and b = %d", a, b);
    }
}
```

```
a = 5 and b = 10
a = 10 and b = 5
Process finished with exit code 0
```

WEEK 3

7. Write a Java program to calculate the area of a rectangle given its length and breadth.

week3_6.java week3_7.java

```java
import java.util.Scanner;

public class week3_7 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double l,b;
        System.out.println("Enter length and breadth :");
        l = sc.nextDouble();
        b = sc.nextDouble();
        System.out.println("Area = " + l*b);
    }
}
```

```
Enter length and breadth :
5
10
Area = 50.0
```

8. Write a Java program to convert temperature from Celsius to Fahrenheit.

week3_6.java week3_7.java week3_8.java

```java
import java.util.Scanner;

public class week3_8 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter Temp in Celsius : ");
        double temp = sc.nextDouble();
        System.out.println("Fahrenheit = " + (temp*9/5.0+32));
    }
}
```

```
Enter Temp in Celsius :
100
Fahrenheit = 212.0
```

WEEK 3

9. Write a Java program that takes two integer inputs and computes their remainder and quotient.

week3_6.java week3_7.java week3_8.java week3_9.java

```java
import java.util.Scanner;

public class week3_9 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a,b;
        System.out.println("Enter a and b:");
        a = sc.nextInt();
        b = sc.nextInt();
        System.out.println("a/b = "+ a/b);
        System.out.println("a%b = "+ a%b);
    }
}
```

```
Enter a and b:
16
3
a/b = 5
a%b = 1
```

10. Write a Java program to find the circumference of a circle given its radius.

week3_6.java week3_7.java week3_8.java wee

```java
import java.util.Scanner;

public class week3_10 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the radius :");
        double r = sc.nextDouble();
        System.out.println("Area = "+ 3.14*r*r);
    }
}
```

```
Enter the radius :
7
Area = 153.86
```

WEEK 4

1. Write a java program to check whether the given number is odd or even.

week4_1.java

```java
public class week4_1 {
    public static void main(String[] args) {
        int a = 10;
        if(a%2 == 0) System.out.println("Even");
        else System.out.println("Odd");
    }
}
```

```
Even

Process finished with exit code 0
```

2. Write a java program to find the largest number among the three numbers.

week4_1.java week4_2.java

```java
public class week4_2 {
    public static void main(String[] args) {
        int a=5, b=10, c=15;
        if (a>b) {
            if (a>c) System.out.println("Greatest = " + a);
            else System.out.println("Greatest = " + c);
        }
        else{
            if (b>c) System.out.println("Greatest = "+b);
            else System.out.println("Greatest = "+c);
        }
    }
}
```

```
Greatest = 15

Process finished with exit code 0
```

3. Write a Java program that takes a number as input and prints its multiplication table upto 10.

WEEK 4

week4_1.java week4_2.java week4_3.java

```java
import java.util.Scanner;

public class week4_3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number :");
        int n = sc.nextInt();
        for (int i = 1; i <= 10; i++)
            System.out.printf("%d x %d = %d\n", n, i, n*i);
    }
}

```

```
Enter the number :
5
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
```

4. Write a Java program to calculate the sum of following series:
1 + 2 + 3 + 4 + ... + N.

WEEK 4

week4_1.java week4_2.java week4_3.java

```java
import java.util.Scanner;

public class week4_4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter N :");
        int n = sc.nextInt();
        int sum = 0;
        for (int i = 1; i<=n; i++) sum += i;
        System.out.println("Result : "+ sum);
    }
}

```

```
Enter N :
10
Result : 55
```

5. Write a Java program to take a number, divide it by 2 and print the result until the number becomes less than 10.

week4_1.java week4_2.java week4_3.java

```java
import java.util.Scanner;

public class week4_5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number :");
        int n = sc.nextInt();
        while(n>10){
            n/=2;
            System.out.println(n);
        }
    }
}

```

```
Enter number :
50
25
12
6
```

WEEK 4

6. Write a Java program to check whether a given character is a vowel or consonant.

week4_6.java

```java
import java.util.Scanner;

public class week4_6 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter char :");
        char c = sc.nextLine().charAt(0);
        c = Character.toUpperCase(c);
        if (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U')
            System.out.println("Vowel");
        else System.out.println("Consonant");
    }
}

```

```
Enter char :
a
Vowel
```

7. Write a Java program to find the smallest number among four given numbers.

week4_6.java week4_7.java

```java
public class week4_7 {
    public static void main(String[] args) {
        int a=5,b=100,c=15,d=20;
        int max = a;
        if(max<b) max=b;
        if(max<c) max=c;
        if(max<d) max=d;
        System.out.println("Greatest = "+max);
    }
}

```

```
Greatest = 100

Process finished with exit code 0
```

WEEK 4

8. Write a Java program to calculate the sum of all even numbers from 1 up to a given number N.

week4_6.java | week4_7.java | week4_8.java

```java
import java.util.Scanner;

public class week4_8 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter N :");
        int n = sc.nextInt();
        int sum = 0;
        for (int i = 2; i<=n; i+=2) sum += i;
        System.out.println("Result : "+ sum);
    }
}

```

```
Enter N :
50
Result : 650
```

9. Write a Java program to check whether a given year is a leap year or not.

week4_6.java | week4_7.java | week4_8.java | week

```java
public class week4_9 {
    public static void main(String[] args) {
        int y=2024;
        if(y%400 == 0)
            System.out.println("Leap year");
        else if (y%4 == 0 && y%100 != 0)
            System.out.println("Leap year");
        else
            System.out.println("Not Leap Year");
    }
}

```

```
Leap year

Process finished with exit code 0
```

WEEK 4

10. Write a Java program that takes a number as input and prints all its factors.

week4_6.java | week4_7.java | week4_8.java | week4_9.java

```java
import java.util.Scanner;

public class week4_10 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter N :");
        int n = sc.nextInt();
        System.out.println("Factors :");
        for(int i=1; i<=n; i++) if(n%i == 0) System.out.println(i);
    }
}
```

```
Enter N :
20
Factors :
1
2
4
5
10
20
```

WEEK 5

1. Write a Java program to insert 10, 20, 30 ....in an array and display them.

week5_1.java

```java
import java.util.Scanner;

public class week5_1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length of Array :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter the elements :");
        for(int i=0;i<l;i++) arr[i] = sc.nextInt();
        System.out.print("Array : ");
        for(int i=0;i<l;i++) System.out.print(arr[i]+" ");
    }
}

```

```
Enter length of Array :
5
Enter the elements :
10 20 30 40 50
Array : 10 20 30 40 50
```

2. Write a Java program to calculate the sum of all the array elements.

week5_1.java week5_2.java

```java
import java.util.Scanner;

public class week5_2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length of Array :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        int sum=0;
        System.out.println("Enter the elements :");
        for(int i=0;i<l;i++) {
            arr[i] = sc.nextInt();
            sum+=arr[i];
        }
        System.out.println("Sum of all elements = "+sum);
    }
}

```

WEEK 5

```
Enter length of Array :
5
Enter the elements :
1 2 3 4 5
Sum of all elements = 15
```

3. Write a java program to print the following pattern:

```
    1
   12
  123
 1234
12345
```

week5_1.java | week5_2.java | week5_3.java

```java
public class week5_3 {
    public static void main(String[] args) {
        for (int i = 1; i <= 5; i++){
            for (int j = 1; j <= 5-i; j++)
                System.out.print(" ");
            for (int k = 1; k <= i; k++)
                System.out.print(k);
            System.out.println();
        }
    }
}
```

```
    1
   12
  123
 1234
12345
```

4. Write a java program to find the sum of following series where n is input by the user:
1+1/2+1/3+1/4+.....+1/n

WEEK 5

week5_1.java week5_2.java week5_3.java

```java
import java.util.Scanner;

public class week5_4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter N :");
        int n = sc.nextInt();
        double sum=0;
        for(int i=1; i<=n; i++) sum+=1.0/i;
        System.out.println("Result = "+sum);
    }
}

```

```
Enter N :
5
Result = 2.283333333333333
```

5. Write a Java program and compute the sum of the digits of an integer.

week5_1.java week5_2.java week5_3.java wee

```java
import java.util.Scanner;

public class week5_5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter an integer :");
        int n = sc.nextInt();
        int sum=0;
        while(n!=0){
            sum+=n%10;
            n/=10;
        }
        System.out.println("Sum of digits = "+sum);
    }
}

```

```
Enter an integer :
123
Sum of digits = 6
```

WEEK 5

6. Write a Java program to calculate the factorial of a number.

week5_6.java

```java
import java.util.Scanner;

public class week5_6 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number :");
        int n = sc.nextInt();
        int fact = 1;
        for(int i=2; i<=n; i++) fact*=i;
        System.out.println("Factorial = "+fact);
    }
}

```

```
Enter number :
6
Factorial = 720
```

7. Write a Java program to find the largest element in a given integer array.

week5_6.java week5_7.java

```java
import java.util.Scanner;

public class week5_7 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length of Array :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter the elements :");
        int max=0;
        for(int i=0;i<l;i++) {
            arr[i] = sc.nextInt();
            if(i==0) max = arr[i];
            if(arr[i] > max) max = arr[i];
        }
        System.out.println("Biggest element = "+max);
    }
}

```

WEEK 5

```
Enter length of Array
5
Enter the elements :
6 4 8 3 2
Biggest element = 8
```

8. Write a Java program to reverse the digits of a given integer.

week5_6.java | week5_7.java | week5_8.java

```java
import java.util.Scanner;

public class week5_8 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter an integer :");
        int n = sc.nextInt();
        int rev = 0;
        while(n!=0){
            rev = rev*10 + n%10;
            n/=10;
        }
        System.out.println("Reverse = "+ rev);
    }
}
```

```
Enter an integer
12345
Reverse = 54321
```

WEEK 5

9. Write a Java program to check if a given number is a palindrome or not.

week5_6.java | week5_7.java | week5_8.java | week5_9.

```java
import java.util.Scanner;

public class week5_9 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter an integer :");
        int n = sc.nextInt();
        int copy = n;
        int rev = 0;
        while(copy !=0){
            rev = rev*10 + copy %10;
            copy /=10;
        }
        if(rev == n) System.out.println("Palindrome");
        else System.out.println("Not Palindrome");
    }
}

```

```
Enter an integer :
12321
Palindrome
```

WEEK 5

10. Write a Java program to convert a decimal number into Hexadecimal number and vice-versa.

week5_6.java week5_7.java week5_8.java week5_9.java

```java
import java.util.Scanner;

public class week5_10 {

    static String decimalToHex(int num) {  1 usage
        if (num == 0) return "0";
        String hex = "";
        char[] hexDigits = "0123456789ABCDEF".toCharArray();
        while (num > 0) {
            int rem = num % 16;
            hex = hexDigits[rem] + hex;
            num = num / 16;
        }
        return hex;
    }

    static int hexToDecimal(String hex) {  1 usage
        int dec = 0;
        hex = hex.toUpperCase();
        for (int i = 0; i < hex.length(); i++) {
            char ch = hex.charAt(i);
            int value;
            if (ch >= '0' && ch <= '9') {
                value = ch - '0';
            } else {
                value = ch - 'A' + 10;
            }
            dec = dec * 16 + value;
        }
        return dec;
    }
```

WEEK 5

```java

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("1. Decimal to Hex");
        System.out.println("2. Hex to Decimal");
        System.out.print("Choose option: ");
        int choice = sc.nextInt();
        sc.nextLine(); //clears input buffer

        switch (choice) {
            case 1:
                System.out.print("Enter decimal number: ");
                int dec = sc.nextInt();
                String hex = decimalToHex(dec);
                System.out.println("Hexadecimal: " + hex);
                break;

            case 2:
                System.out.print("Enter hexadecimal number: ");
                String hexInput = sc.nextLine();
                int decimal = hexToDecimal(hexInput);
                System.out.println("Decimal: " + decimal);
                break;

            default:
                System.out.println("Invalid option");
        }
    }
}

```

```
1. Decimal to Hex
2. Hex to Decimal
Choose option: 2
Enter hexadecimal number: 4f
Decimal: 79
```

WEEK 5

11. Write a Java program to print the following pattern:

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

week5_6.java week5_7.java week5_8.java

```java
public class week5_11 {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= 3 - i; j++)
                System.out.print(" ");
            for (int k = 1; k <= i; k++)
                System.out.print("*");
            System.out.println();
        }
        for (int i = 2; i >= 1; i--){
            for (int j = 1; j <= 3 - i; j++)
                System.out.print(" ");
            for (int k = 3 - i; k < 3; k++)
                System.out.print("*");
            System.out.println();
        }
    }
}

```

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

WEEK 6

1. Write a Java program to print the odd numbers from 1 to 99.

```java
public class week6_1 {
    public static void main(String[] args) {
        for (int i = 1; i < 100; i+=2)
            System.out.println(i);
    }
}
```

```
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
Process finished with exit code 0
```

2. Write a Java program to check whether a number is prime or not.

```java
import java.util.Scanner;

public class week6_2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter a number :");
        int n = sc.nextInt();
        boolean flag = true;
        for (int i = 2; i < n; i++)
            if (n % i == 0) {
                flag = false;
                break;
            }
        if (flag) System.out.println("Prime");
        else System.out.println("Not Prime");
    }
}
```

```
"C:\Program Files\Java\
Enter a number :
41
Prime
```

3. Write a Java program to swap the first and last elements of an array.

WEEK 6

week6_1.java | week6_2.java | week6_3.java | week6_4.java

```java
import java.util.Scanner;

public class week6_3 {  Stephen047
    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length of Array :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter the elements :");
        for(int i=0;i<l;i++) arr[i] = sc.nextInt();
        System.out.println("Current Array :");
        for(int i=0;i<l;i++) System.out.print(arr[i]+", ");
        System.out.println();
        int temp = arr[l - 1];
        arr[l - 1] = arr[0];
        arr[0] = temp;
        System.out.println("After swap :");
        for(int i=0;i<l;i++) System.out.print(arr[i]+", ");
    }
}
```

```
Enter length of Array :
5
Enter the elements :
1 2 3 4 5
Current Array :
1, 2, 3, 4, 5,
After swap :
5, 2, 3, 4, 1,
Process finished with exit code 0
```

4. Write a Java program to find the maximum and minimum among array elements.

week6_1.java | week6_2.java | week6_3.java | week6_4.java

```java
import java.util.Scanner;

public class week6_4 {  Stephen047
    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length of Array :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter the elements :");
        int max=0, min=0;
        for(int i=0;i<l;i++) {
            arr[i] = sc.nextInt();
            if(i==0) {
                max = arr[i];
                min = arr[i];
            }
            if(arr[i] > max) max = arr[i];
            if(arr[i] < min) min = arr[i];
        }
        System.out.println("Biggest element = "+max);
        System.out.println("Smallest element = "+min);
    }
}
```

WEEK 6

```
Enter length of Array :
5
Enter the elements :
21 54 67 98 21
Biggest element = 98
Smallest element = 21
```

5. Write a Java program to print all prime numbers between 0 to 100.

week6_1.java | week6_2.java | week6_3.java | week6_4.java | week

```java
public class week6_5 {  Stephen047 *
    public static void main(String[] args) {  Stephen047 *
        Outer : for (int i=0; i<=100; i++){
            if (i == 0 || i == 1) continue;
            for (int j=2; j<i; j++) if (i%j == 0) continue Outer;
            System.out.print(i+" ");
        }
    }
}
```

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
Process finished with exit code 0
```

6. Write a Java program to implement linear search.

WEEK 6

week6_1.java | week6_2.java | week6_3.java | week6_4.java

```java
import java.util.Scanner;

public class week6_6 {  Stephen047
    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length of Array :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter the elements :");
        for(int i=0;i<l;i++) arr[i] = sc.nextInt();
        System.out.println("Enter target :");
        int x = sc.nextInt();
        boolean flag = false;
        int i;
        for(i=0; i<l; i++)
            if (x == arr[i]){
                flag = true;
                break;
            }
        if (flag) System.out.println("Found at index "+i);
        else System.out.println("Not found");
    }
}
```

```
Enter length of Array :
5
Enter the elements :
5 6 4 9 10
Enter target :
9
Found at index 3
```

7. Write a Java program to print all prime numbers between 0 to 100.

week6_6.java | week6_7.java × | week6_8.java | week6_9.java | week6_

```java
public class week6_7 {  Stephen047 *
    public static void main(String[] args) {  Stephen047 *
        Outer : for (int i=0; i<=100; i++){
            if (i == 0 || i == 1) continue;
            for (int j=2; j<i; j++) if (i%j == 0) continue Outer;
            System.out.print(i+" ");
        }
    }
}
```

```
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
Process finished with exit code 0
```

WEEK 6

8. Write a Java program to find the second largest element in an array.

week6_6.java | week6_7.java | week6_8.java | week6_9.ja

```java
import java.util.Scanner;

public class week6_8 {  Stephen047
    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length of Array :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter the elements :");
        int maxi =0;
        for(int i=0;i<l;i++) {
            arr[i] = sc.nextInt();
            if(arr[i] > arr[maxi]) maxi = i;
        }
        int m2 =arr[0];
        for(int i=0;i<l;i++) {
            if(i == maxi) continue;
            if(arr[i] > m2) m2 = arr[i];
        }
        System.out.println("2nd Biggest : "+m2);
    }
}
```

```
"C:\Program Files\Java\jdk-17\bin\java.exe" "-j
Enter length of Array :
5
Enter the elements :
21 48 56 19 18
2nd Biggest : 48
```

9. Write a program to implement Fibonacci series up to N terms.

WEEK 6

week6_6.java | week6_7.java | week6_8.java | week6_9.java | week

```java
import java.util.Scanner;

public class week6_9 {  Stephen047
    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter N :");
        int n = sc.nextInt();
        if (n >= 1) System.out.print("0 ");
        if (n >= 2) System.out.print("1 ");
        if (n >= 3){
            int a = 0, b = 1;
            int c;
            for (int i = 2; i <= n; i++){
                c = a+b;
                a = b;
                b = c;
                System.out.print(c+" ");
            }
        }
    }
}
```

```
Enter N :
10
0 1 1 2 3 5 8 13 21 34 55
Process finished with exit code 0
```

10. Write a Java program to reverse all elements of an array.

```
Enter length of Array :
5
Enter the elements :
1 2 3 4 5
Current Array :
1, 2, 3, 4, 5,
Current Array :
5, 4, 3, 2, 1,
Process finished with exit code 0
```

WEEK 6

week6_6.java week6_7.java week6_8.java week6_9.java

```java
import java.util.Scanner;

public class week6_10 {  Stephen047
    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length of Array :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter the elements :");
        for(int i=0;i<l;i++) arr[i] = sc.nextInt();
        System.out.println("Current Array :");
        for(int i=0;i<l;i++) System.out.print(arr[i]+", ");
        System.out.println();
        for (int i=0;i<l/2;i++){
            int temp = arr[i];
            arr[i] = arr[l-1-i];
            arr[l-1-i] = temp;
        }
        System.out.println("Current Array :");
        for(int i=0;i<l;i++) System.out.print(arr[i]+", ");
    }
}
```

11. Write a Java program to find the frequency of each character in a given string.

week6_6.java week6_7.java week6_8.java week6_9.java week6_10.java

```java
import java.util.Scanner;

public class week6_11 {  Stephen047
    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter String :");
        String str = sc.nextLine();
        for (int i = 0; i < str.length(); i++){
            char c = str.charAt(i);
            int count = 0;
            for (int j = 0; j < str.length(); j++) if (str.charAt(j) == c) count++;
            System.out.println("Freq of "+c+" = "+count);
        }
    }
}
```

```
Enter String :
Shafin
Freq of S = 1
Freq of h = 1
Freq of a = 1
Freq of f = 1
Freq of i = 1
Freq of n = 1
```

WEEK 7

1. Write a Java function to implement binary search.

week7_1.java × week7_2.java week7_3.java week7_4.java week7_5.java week7_6.java

```java
import java.util.Scanner;

public class week7_1 {  Stephen047

    static int search(int[] arr, int x){  1 usage  Stephen047
        int low = 0;
        int high = arr.length - 1;
        while(low <= high){
            int mid = (low + high)/2;
            if(arr[mid] == x) return mid;
            else if(x < arr[mid]) high = mid-1;
            else low = mid + 1;
        }
        return -1;
    }

    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the length :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter elements : ");
        for (int i = 0; i < l; i++) {
            arr[i] = sc.nextInt();
        }
        System.out.print("Enter the target : ");
        int x = sc.nextInt();

        int index = search(arr, x);
        if(index != -1)
            System.out.println("Found at index "+index);
        else
            System.out.println("Not Found");
    }
}
```

```
Enter the length :
5
Enter elements :
1 2 3 4 5
Enter the target : 4
Found at index 3
```

2. Write a Java function to arrange the elements of an array in ascending order (Sorting).

WEEK 7

week7_1.java | week7_2.java | week7_3.java | week7_4.java | week7_5.java

```java
import java.util.Scanner;

public class week7_2 {  Stephen047

    static void sort(int[] arr){  1 usage  Stephen047
        for (int i = 0; i < arr.length - 1; i++) {
            boolean swapped = false;
            for (int j = 0; j < arr.length - 1 - i; j++)
                if (arr[j] > arr[j+1]) {
                    int temp = arr[j];
                    arr[j] = arr[j+1];
                    arr[j+1] = temp;
                    swapped = true;
                }
            if(!swapped) return;
        }
    }

    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the length :");
        int l = sc.nextInt();
        int[] arr = new int[l];
        System.out.println("Enter elements : ");
        for (int i = 0; i < l; i++) {
            arr[i] = sc.nextInt();
        }
        sort(arr);
        System.out.println("Sorted Array :");
        for (int j : arr) {
            System.out.print(j + " ");
        }
    }
}
```

```
Enter the length :
5
Enter elements :
1 3 4 2 5
Sorted Array :
1 2 3 4 5
```

3. Write a program to reverse a given string.

WEEK 7

week7_1.java week7_2.java week7_3.java × week7_4.java

```java
import java.util.Scanner;

public class week7_3 {  Stephen047
    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a String : ");
        String str = sc.next();
        String rev = "";
        for (int i = 0; i < str.length(); i++) {
            rev = str.charAt(i) + rev;
        }
        System.out.println("Reversed string : "+rev);
    }
}

```

```
Enter a String : SHaheer
Reversed string : reehaHS
```

4. Write a program to check whether a given string is palindrome or not.

week7_1.java week7_2.java week7_3.java week7_4.java ×

```java
import java.util.Scanner;

public class week7_4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a String : ");
        String str = sc.next().toLowerCase();
        String rev = "";
        for (int i = 0; i < str.length(); i++) {
            rev = str.charAt(i) + rev;
        }
        if (str.equals(rev)) System.out.println("Palindrome");
        else System.out.println("Not Palindrome");
    }
}

```

```
Enter a String : Dad
Palindrome
```

5. Write a program to implement factorial of a number through recursion.

WEEK 7

week7_1.java week7_2.java week7_3.java

```java
import java.util.Scanner;

public class week7_5 {

    static int fact(int n){  2 usages
        if (n <= 1) return 1;
        return n * fact( n: n-1);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int n = sc.nextInt();
        System.out.println(fact(n));
    }
}
```

```
Enter a number : 6
720
```

6. Write a program to implement Fibonacci series of a number with and without recursion.

week7_1.java week7_2.java week7_3.java week7_4.java wee

```java
public class week7_6 {  Stephen047

    static void fib(int n){  1 usage  Stephen047
        if (n >= 1) System.out.print("0 ");
        if (n >= 2) System.out.print("1 ");
        if (n > 2) {
            int a = 0;
            int b = 1;
            for (int i = 3; i <= n; i++) {
                int c = a + b;
                a = b;
                b = c;
                System.out.print(c + " ");
            }
        }
    }
    static void fib_rec(int n, int a, int b){  2 usages  Stephen047
        if(n > 0){
            System.out.print(a + " ");
            fib_rec( n: n-1, b,  b: a+b);
        }
    }

    public static void main(String[] args) {  Stephen047
        fib( n: 10);
        System.out.println("\nWith recursion : ");
        fib_rec( n: 10,  a: 0,  b: 1);
    }
}
```

```
0 1 1 2 3 5 8 13 21 34
With recursion :
0 1 1 2 3 5 8 13 21 34
```

WEEK 7

7. Write a Java function to find the greatest common divisor (GCD) of two numbers with and without using recursion.

week7_4.java | week7_5.java | week7_6.java | week7_7.java

```java
public class week7_7 {  Stephen047
    static int gcd(int a, int b){  1 usage  Stephen047
        while(a%b != 0){
            int r = a%b;
            a = b;
            b = r;
        }
        return b;
    }
    static int gcd_rec(int a, int b){  2 usages  Stephen047
        if (a%b == 0) return b;
        return gcd_rec(b,  b: a%b);
    }

    public static void main(String[] args) {  Stephen047
        System.out.println(gcd( a: 15,  b: 100));
        System.out.println("With Recursion : ");
        System.out.println(gcd_rec( a: 15,  b: 100));
    }
}
```

```
5
With Recursion :
5
```

8. Write a program to check whether two strings are anagrams of each other (“listen” and “silent” are anagrams).

```java
import java.util.Arrays;
import java.util.Scanner;

public class week7_8 {  Stephen047

    static String sort(String str){  2 usages  Stephen047
        char[] temp = str.toLowerCase().toCharArray();
        Arrays.sort(temp);
        return new String(temp);
    }

    public static void main(String[] args) {  Stephen047
        Scanner sc = new Scanner(System.in);
        System.out.print("String 1 : ");
        String s1 = sc.nextLine();
        System.out.print("String 2 : ");
        String s2 = sc.nextLine();
        if ((sort(s1).trim()).equals(sort(s2).trim()))
            System.out.println("Anagrams");
        else System.out.println("Not Anagrams");
    }
}
// Examples - Listen, Silent; Heart, Earth; The Eyes, They see; Debit Card, Bad Credit;
```

WEEK 7

```
String 1 : Debit Card
String 2 : Bad Credit
Anagrams
```

9. Implement quick sort using recursion.

week7_4.java week7_5.java week7_6.java week7_7.java week7_8.java

```java
public class week7_9 {  Stephen047
    static int partition(int[] arr, int low, int high) {  1 usage  Stephen047

        int pivot = arr[high];
        int i = low - 1;
        for (int j = low; j < high; j++) {
            if (arr[j] < pivot) {
                i++;
                swap(arr, i, j);
            }
        }
        //putting pivot (high) in front of last exchange
        swap(arr,  i: i + 1, high);
        return i + 1;
    }
    static void swap(int[] arr, int i, int j) {  2 usages  Stephen047
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
    }

    static void quickSort(int[] arr, int low, int high) {  3 usages  Stephen047
        if (low < high) {
            //pi is pivot index
            int pi = partition(arr, low, high);
            quickSort(arr, low,  high: pi - 1);
            quickSort(arr,  low: pi + 1, high);
        }
    }
```

```
C:\Program Files\Ja
1 5 7 8 9 10
Process finished wit
```

WEEK 8

1. Create a class FRUIT which has data members color, taste and price. Also create a method display() which will print values of FRUIT object. Create three objects of FRUIT class and call their display() methods.
2. Create a class FRUIT which has data members color, taste and price. It has a method setDetails() which will set the values of color, taste and price. Also create a method display() which will print values of FRUIT object.
3. In previous question, set the values of using color, taste and price using Constructor.
4. Add one-argument constructor and two-argument constructor in addition to default constructor in FRUIT class.
5. Use the concept of constructor-chaining in the previous question using this().

Main.java FRUIT.java

```java
package week8_1;

public class FRUIT {  6 usages
    private String color;  4 usages
    private String taste;  4 usages
    private double price;  4 usages

    FRUIT(String color, String taste){  2 usages
        this.color = color;
        this.taste = taste;
        this.price = 0;
    }
    FRUIT(String color) { this(color,  taste: "Unknown"); }
    FRUIT(){  1 usage
        color = "Unknown";
        taste = "Unknown";
        price = 0;
    }

    public void setDetails(String color, String taste, double price){  3 usages
        this.color = color;
        this.taste = taste;
        this.price = price;
    }

    public void display(){
        System.out.println("Color : "+color);
        System.out.println("Taste : "+taste);
        System.out.println("Price : $"+price);
    }
}
```

WEEK 8

Main.java × FRUIT.java

```java
package week8_1;

public class Main {
    public static void main(String[] args) {
        FRUIT apple = new FRUIT( color: "Red",  taste: "Sweet");
        FRUIT banana = new FRUIT( color: "Yellow");
        FRUIT grape = new FRUIT();

        System.out.println("Apple:");
        apple.display();
        System.out.println("Banana:");
        banana.display();
        System.out.println("Grape:");
        grape.display();

        apple.setDetails( color: "Red",  taste: "Sweet",  price: 6.99);
        banana.setDetails( color: "Yellow",  taste: "Sweet",  price: 3.99);
        grape.setDetails( color: "Green",  taste: "Sour",  price: 5.99);

        System.out.println("Apple:");
        apple.display();
        System.out.println("Banana:");
        banana.display();
        System.out.println("Grape:");
        grape.display();
    }
}
```

```
Apple:
Color : Red
Taste : Sweet
Price : $0.0
Banana:
Color : Yellow
Taste : Unknown
Price : $0.0
Grape:
Color : Unknown
Taste : Unknown
Price : $0.0
```

```
Apple:
Color : Red
Taste : Sweet
Price : $6.99
Banana:
Color : Yellow
Taste : Sweet
Price : $3.99
Grape:
Color : Green
Taste : Sour
Price : $5.99
```

6. Create a class CAR with the following details:
Data members: model, color, price.
Member methods:
setDetails() – to set values of all data members using setters.
getDetails() – to get values of all data members using getters.
display() – to print all details of the car.

WEEK 8

Requirements:
1. Implement default constructor to initialize default values.
2. Implement a parameterized constructor (with one argument) to set only model.
3. Implement another parameterized constructor (with two arguments) to set model and color.
4. Use constructor chaining to reduce code redundancy.
5. Create three objects of CAR class using:
- Default constructor
- One-argument constructor
- Two-argument constructor
6. Set price for each object using the setDetails() method.
7. Call the display() method for each object.

Main.java CAR.java

```java
package week8_2;

public class CAR {  6 usages
    private String color;  4 usages
    private String model;  4 usages
    private double price;  4 usages

    CAR(String color, String model){  2 usages
        this.color = color;
        this.model = model;
        this.price = 0;
    }
    CAR(String color) { this(color,  model: "Unknown"); }
    CAR(){  1 usage
        color = "Unknown";
        model = "Unknown";
        price = 0;
    }

    public void setDetails(String color, String model, double price){  3 usages
        this.color = color;
        this.model = model;
        this.price = price;
    }

    public void getDetails() { display(); }

    public void display(){
        System.out.println("Color : "+color);
        System.out.println("Model : "+ model);
        System.out.println("Price : $"+price);
    }
}

```

WEEK 8

Main.java × CAR.java

```java
package week8_2;

public class Main {
    public static void main(String[] args) {
        CAR porsche = new CAR( color: "White", model: "911 GT3 RS");
        CAR mercedes = new CAR( color: "Navy");
        CAR ferrari = new CAR();

        System.out.println("Porsche:");
        porsche.getDetails();
        System.out.println("Mercedes:");
        mercedes.getDetails();
        System.out.println("Ferrari:");
        ferrari.getDetails();

        porsche.setDetails( color: "White", model: "911 GT3 RS", price: 100000);
        mercedes.setDetails( color: "Navy", model: "1986 S Class", price: 150000);
        ferrari.setDetails( color: "Red", model: "F40", price: 200000);

        System.out.println("Porsche:");
        porsche.display();
        System.out.println("Mercedes:");
        mercedes.display();
        System.out.println("Ferrari:");
        ferrari.display();
    }
}

```

```
Porsche:
Color : White
Model : 911 GT3 RS
Price : $0.0
Mercedes:
Color : Navy
Model : Unknown
Price : $0.0
Ferrari:
Color : Unknown
Model : Unknown
Price : $0.0
```

```
Porsche:
Color : White
Model : 911 GT3 RS
Price : $100000.0
Mercedes:
Color : Navy
Model : 1986 S Class
Price : $150000.0
Ferrari:
Color : Red
Model : F40
Price : $200000.0
```

WEEK 9

1. Create a class Vehicle, write a method cost() in this class. Create two classes Bus and Train which have their own display() methods and inherit from Vehicle class. Create objects of Bus and Train class and call cost() and display() methods.

Main.java

```
package week9_1;

public class Main {
    public static void main(String[] args) {
        Bus myBus = new Bus();
        myBus.display();
        myBus.cost();

        System.out.println();

        Train myTrain = new Train();
        myTrain.display();
        myTrain.cost();
    }
}
```

Bus.java

```
package week9_1;

class Bus extends Vehicle {  2 usages
    public void display() {
        System.out.println("This is a Bus.");
    }
}
```

Train.java

```
package week9_1;

class Train extends Vehicle {  2 usages
    public void display() {
        System.out.println("This is a Train.");
    }
}
```

Vehicle.java

```
package week9_1;

class Vehicle {  2 usages  2 inheritors
    public void cost() {  2 usages
        System.out.println("The cost of the vehicle is...");
    }
}
```

```
This is a Bus.
The cost of the vehicle is...

This is a Train.
The cost of the vehicle is...
```

WEEK 9

2. Create class University which has data members- name and ranking. Create class Faculty that extends University class has data member- name and method- Details(). Create a new class Department which is derived from Faculty and has data member- name, chairman and method- Details() and Display() where Display() method calls Details() methods of both Faculty and Department class in its body. Create an object of Department class to Display() method and University ranking.

Main.java | Department.java | Faculty.java | University.java

```java
package week9_2;

class University {  1 usage  2 inheritors
    String name;  2 usages
    int ranking;  2 usages

    public University(String name, int ranking) {  1 usage
        this.name = name;
        this.ranking = ranking;
    }
}
```

Main.java | Department.java | Faculty.java | University.java

```java
package week9_2;

class Faculty extends University {  1 usage  1 inheritor
    String facultyName;  2 usages

    public Faculty(String uniName, int ranking, String facultyName) {  1 usage
        super(uniName, ranking);
        this.facultyName = facultyName;
    }

    public void Details() {  2 usages  1 override
        System.out.println("Faculty: " + this.facultyName);
    }
}
```

WEEK 9

Main.java × Department.java × Faculty.java University.java

```java
package week9_2;

class Department extends Faculty {  2 usages
    String deptName;  2 usages
    String chairman;  2 usages

    public Department(String uniName, int ranking, String facultyName, String deptName, String chairman) {
        super(uniName, ranking, facultyName);
        this.deptName = deptName;
        this.chairman = chairman;
    }
    public void Details() {  2 usages
        System.out.println("Department: " + this.deptName);
        System.out.println("Chairman: " + this.chairman);
    }
    public void Display() {  1 usage
        System.out.println("--- Complete Details ---");
        System.out.println("University: " + this.name);
        super.Details();
        this.Details();
    }
}
```

Main.java × Department.java Faculty.java University.java

```java
package week9_2;

public class Main {
    public static void main(String[] args) {
        Department csDept = new Department( uniName: "AMU", ranking: 8,
                facultyName: "F/O Science", deptName: "CS Dept.", chairman: "Mr. A.R. Faridi");

        csDept.Display();

        System.out.println("------------------------");
        System.out.println("University Ranking: " + csDept.ranking);
    }
}
```

```
--- Complete Details ---
University: AMU
Faculty: F/O Science
Department: CS Dept.
Chairman: Mr. A.R. Faridi
------------------------
University Ranking: 8
```

3. Create class Account (Data members- Id, Account_holder_name, Address; Methodsdeposit(), withdraw()). Create two static methods in Account calculateSimpleInterest() and calculateCompoundInterest() and implement them.

WEEK 9

Main.java | Account.java

```java
package week9_3;

class Account {  4 usages
    int id;  2 usages
    String Account_holder_name;  2 usages
    String Address;  2 usages
    private double balance;  7 usages

    public Account(int id, String name, String address, double initialBalance) {  1 usage
        this.id = id;
        this.Account_holder_name = name;
        this.Address = address;
        this.balance = initialBalance;
    }

    public void deposit(double amount) {  1 usage
        if (amount > 0) {
            this.balance += amount;
            System.out.printf("Successfully deposited %.2f. New balance is %.2f\n", amount, this.balance);
        } else {
            System.out.println("Deposit amount must be positive.");
        }
    }

    public void withdraw(double amount) {
        if (amount <= 0) {
            System.out.println("Withdrawal amount must be positive.");
        } else if (this.balance >= amount) {
            this.balance -= amount;
            System.out.printf("Successfully withdrew %.2f. New balance is %.2f\n", amount, this.balance);
        } else {
            System.out.println("Withdrawal failed. Insufficient funds.");
        }
    }

    public void displayDetails() {  2 usages
        System.out.println("--- Account " + this.id + " ---");
        System.out.println("Holder: " + this.Account_holder_name);
        System.out.println("Address: " + this.Address);
        System.out.printf("Current Balance: %.2f\n", this.balance);
        System.out.println("----------------------");
    }

    public static double calculateSimpleInterest(double p, double r, double t) {  1 usage
        return (p * r * t) / 100.0;
    }

    public static double calculateCompoundInterest(double p, double r, double t ) {  1 usage
        double rateAsDecimal = r / 100.0;
        double amount = p * Math.pow(1 + rateAsDecimal, t);
        return amount - p;
    }
}
```

WEEK 9

Main.java × Account.java

```java
package week9_3;

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

        Account account1 = new Account( id: 101, name: "Amit Sharma", address: "Agra, UP", initialBalance: 25000.0);

        System.out.println("Using non-static methods:");
        account1.displayDetails();
        account1.deposit( amount: 10000.0);
        account1.withdraw( amount: 5000.0);
        account1.displayDetails();

        System.out.println();

        System.out.println("Using static methods:");
        double p = 50000;
        double r = 7.5;
        double t = 5;

        double si = Account.calculateSimpleInterest(p, r, t);
        System.out.printf("Simple Interest for %.2f at %.2f%% for %.1f years is: %.2f\n", p, r, t, si);

        double ci = Account.calculateCompoundInterest(p, r, t);
        System.out.printf("Compound Interest for %.2f at %.2f%% for %.1f years (compounded annually) is: %.2f\
    }
}
```

```
Using non-static methods:
--- Account 101 ---
Holder: Amit Sharma
Address: Agra, UP
Current Balance: 25000.00
--------------------
Successfully deposited 10000.00. New balance is 35000.00
Successfully withdrew 5000.00. New balance is 30000.00
--- Account 101 ---
Holder: Amit Sharma
Address: Agra, UP
Current Balance: 30000.00
--------------------

Using static methods:
Simple Interest for 50000.00 at 7.50% for 5.0 years is: 18750.00
Compound Interest for 50000.00 at 7.50% for 5.0 years (compounded annually) is: 21781.47
```

4. Create class Account (Data members- Id, Account_holder_name, Address; Methodsdeposit(), withdraw()). Declare deposit() and withdraw() as abstract methods. Declare Account class as abstract. (Create constructor in Account as well).

5. Create two children of Account- Saving (Data Members- Min_balance; Methodsdisplay(), deposit(), withdraw()) and

WEEK 9

Current (Data Members- Max_withdrawl_limit; Methods- display(),deposit(), withdraw()) . Create constructors for both classes. Implementation of deposit() and withdraw() should be specific to Saving and Current class. Create objects of Saving and Current class and display them

Main.java | Account.java | Current.java | Saving.java

```java
package week9_4;

abstract class Account {  2 usages  2 inheritors
    int id;  7 usages
    String accountHolderName;  3 usages
    String address;  3 usages

    public Account(int id, String name, String address) {  2 usages
        this.id = id;
        this.accountHolderName = name;
        this.address = address;
    }

    public abstract void deposit(double amount);  2 usages  2 implementations
    public abstract void withdraw(double amount);  2 implementations
}
```

Main.java | Account.java | Current.java | Saving.java

```java
package week9_4;

class Current extends Account {  2 usages
    // Data member specific to Current account
    double maxWithdrawLimit;  2 usages

    /**
     * Constructor for the Current class.
     */
    public Current(int id, String name, String address, double maxWithdrawal) {  1 usage
        super(id, name, address); // Initialize parent class members
        this.maxWithdrawLimit = maxWithdrawal;
    }

    @Override  2 usages
    public void deposit(double amount) {
        System.out.printf("Depositing %.2f into Current Account #%d\n", amount, this.id);
    }

    @Override
    public void withdraw(double amount) {
        System.out.printf("Withdrawing %.2f from Current Account #%d\n", amount, this.id);
    }

    public void display() {
        System.out.println("--- Current Account Details ---");
        System.out.println("Account ID: " + this.id);
        System.out.println("Holder Name: " + this.accountHolderName);
        System.out.println("Address: " + this.address);
        System.out.println("Max Withdrawal Limit: " + this.maxWithdrawLimit);
        System.out.println("-----------------------------");
    }
}
```

WEEK 9

Main.java | Account.java | Current.java | Saving.java

```java
package week9_4;

class Saving extends Account {  2 usages
    // Data member specific to Saving account
    double minBalance;  2 usages

    public Saving(int id, String name, String address, double minBalance) {  1 usage
        super(id, name, address); // Initialize parent class members
        this.minBalance = minBalance;
    }

    @Override  2 usages
    public void deposit(double amount) {
        System.out.printf("Depositing %.2f into Saving Account #%d\n", amount, this.id);
    }

    @Override
    public void withdraw(double amount) {
        System.out.printf("Withdrawing %.2f from Saving Account #%d\n", amount, this.id);
    }

    public void display() {
        System.out.println("--- Saving Account Details ---");
        System.out.println("Account ID: " + this.id);
        System.out.println("Holder Name: " + this.accountHolderName);
        System.out.println("Address: " + this.address);
        System.out.println("Minimum Balance Rule: " + this.minBalance);
        System.out.println("----------------------------");
    }
}
```

Main.java | Account.java | Current.java | Saving.java

```java
package week9_4;

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

        Saving savingAccount = new Saving( id: 101,  name: "Shaheer",  address: "AMU",  minBalance: 1000.0);

        Current currentAccount = new Current( id: 202,  name: "Shafin",  address: "AMU",  maxWithdrawal: 100000.0);

        savingAccount.display();
        savingAccount.deposit( amount: 15000);
        savingAccount.withdraw( amount: 4500);

        System.out.println();

        currentAccount.display();
        currentAccount.deposit( amount: 50000);
        currentAccount.withdraw( amount: 25000);
    }
}
```

WEEK 9

```
--- Saving Account Details ---
Account ID: 101
Holder Name: Shaheer
Address: AMU
Minimum Balance Rule: 1000.0
----------------------------
Depositing 15000.00 into Saving Account #101
Withdrawing 4500.00 from Saving Account #101

--- Current Account Details ---
Account ID: 202
Holder Name: Shafin
Address: AMU
Max Withdrawal Limit: 100000.0
-----------------------------
Depositing 50000.00 into Current Account #202
Withdrawing 25000.00 from Current Account #202
```

6. Create a class Shape with a method area(). Create two derived classes Rectangle and Circle that extend Shape. Each class should override the area() method to calculate the area of the respective shape. Create objects of Rectangle and Circle and call their area() methods.

Main.java | Shape.java | Circle.java | Rectangle.java

```java
package week9_6;

public class Main {
    public static void main(String[] args) {
        Rectangle rect = new Rectangle( length: 12.5,  width: 4.0);
        Circle circle = new Circle( radius: 7.5);

        rect.area();
        circle.area();
    }
}
```

Main.java | Shape.java | Circle.java | Rectangle.java

```java
package week9_6;

class Shape {  2 usages  2 inheritors
    public void area() {  2 usages  2 overrides
        System.out.println("Area of Shape is...");
    }
}
```

WEEK 9

Main.java | Shape.java | Circle.java | Rectangle.java

```java
package week9_6;

class Circle extends Shape {  2 usages
    private double radius;  3 usages

    public Circle(double radius) {  1 usage
        this.radius = radius;
    }

    @Override  2 usages
    public void area() {
        double result = Math.PI * radius * radius;
        System.out.printf("Area of Circle is: %.2f\n", result);
    }
}
```

Main.java | Shape.java | Circle.java | Rectangle.java

```java
package week9_6;

class Rectangle extends Shape {  2 usages
    private double length;  2 usages
    private double width;  2 usages

    public Rectangle(double length, double width) {  1 usage
        this.length = length;
        this.width = width;
    }

    @Override  2 usages
    public void area() {
        double result = length * width;
        System.out.println("Area of Rectangle is: " + result);
    }
}
```

```
Area of Rectangle is: 50.0
Area of Circle is: 176.71
```

7. Create a class Employee with data members: name, salary, and a method showDetails(). Create a class Manager that extends Employee with an additional data member department. Override the showDetails() method in Manager to display all details, including department. Create an object of Manager and call showDetails().

WEEK 9

```java
package week9_7;

public class Main {
    public static void main(String[] args) {
        Manager m1 = new Manager( name: "Mohd Nadeem",
                salary: 120000.0, department: "CS");
        System.out.println("Manager Info:");
        m1.showDetails();
    }
}
```

```java
package week9_7;

class Employee {
    String name;
    double salary;

    public Employee(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }

    public void showDetails() {
        System.out.println("Name: " + this.name);
        System.out.println("Salary: " + this.salary);
    }
}
```

```java
package week9_7;

class Manager extends Employee {
    String department;

    public Manager(String name, double salary, String department) {
        super(name, salary);
        this.department = department;
    }

    @Override
    public void showDetails() {
        super.showDetails();
        System.out.println("Department: " + this.department);
    }
}
```

```
Manager Info:
Name: Mohd Nadeem
Salary: 120000.0
Department: CS
```

8. Create an abstract class Appliance with data members brand, power and abstract methods turnOn() and turnOff(). Create two derived classes WashingMachine and Refrigerator that provide their own implementations of turnOn() and turnOff(). Create objects of WashingMachine and Refrigerator and call their methods.

WEEK 9

Main.java | Appliance.java | Refrigerator.java | WashingMachine.java

```java
package week9_8;

public class Main {
    public static void main(String[] args) {
        WashingMachine w = new WashingMachine( brand: "Bosch", power: 2200);
        System.out.println("Washing Machine:");
        w.displayInfo();
        w.turnOn();
        w.turnOff();

        System.out.println();

        Refrigerator fridge = new Refrigerator( brand: "Whirlpool", power: 180);
        System.out.println("Refrigerator: ");
        fridge.displayInfo();
        fridge.turnOn();
        fridge.turnOff();
    }
}
```

Main.java | Appliance.java | Refrigerator.java | WashingMachine.java

```java
package week9_8;

abstract class Appliance {  2 usages  2 inheritors
    String brand;  2 usages
    int power;  2 usages

    public Appliance(String brand, int power) {  2 usages
        this.brand = brand;
        this.power = power;
    }

    public abstract void turnOn();  2 usages  2 implementations
    public abstract void turnOff();  2 usages  2 implementations

    public void displayInfo() {  2 usages
        System.out.println("Brand: " + this.brand);
        System.out.println("Power Consumption: " + this.power + "W");
    }
}
```

Main.java | Appliance.java | Refrigerator.java | Washir

```java
package week9_8;

class Refrigerator extends Appliance {  2 usages
    public Refrigerator(String brand, int power) {  1 usag
        super(brand, power);
    }

    @Override  2 usages
    public void turnOn() {
        System.out.println("The fridge is on.");
    }

    @Override  2 usages
    public void turnOff() {
        System.out.println("The fridge is off.");
    }
}
```

WEEK 9

Main.java | Appliance.java | Refrigerator.java | WashingMachin

```java
package week9_8;

class WashingMachine extends Appliance {  2 usages
    public WashingMachine(String brand, int power) {  1 usage
        super(brand, power);
    }

    @Override  2 usages
    public void turnOn() {
        System.out.println("The washing machine is on.");
    }

    @Override  2 usages
    public void turnOff() {
        System.out.println("The washing machine is off.");
    }
}
```

```
Washing Machine:
Brand: Bosch
Power Consumption: 2200W
The washing machine is on.
The washing machine is off.

Refrigerator:
Brand: Whirlpool
Power Consumption: 180W
The fridge is on.
The fridge is off.
```

9. Create a class MathOperations with two static methods: findGCD(int a, int b) to calculate the greatest common divisor and findLCM(int a, int b) to calculate the least common multiple. Call these methods without creating an object of MathOperations.

WEEK 9

Main.java × MathOperations.java

```java
package week9_9;

public class Main {
    public static void main(String[] args) {
        int number1 = 48;
        int number2 = 18;

        int gcd = MathOperations.findGCD(number1, number2);
        System.out.println("GCD: " + gcd);

        int lcm = MathOperations.findLCM(number1, number2);
        System.out.println("LCM: " + lcm);
    }
}
```

ain.java MathOperations.java ×

```java
package week9_9;

class MathOperations {  2 usages
    public static int findGCD(int a, int b) {  2 usage
        while (b != 0) {
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;
    }
    public static int findLCM(int a, int b) {  1 usage
        if (a == 0 || b == 0) {
            return 0;
        }
        return Math.abs(a * b) / findGCD(a, b);
    }
}
```

```
GCD: 6
LCM: 144
```

10. Create a class Student with data members rollNo, name, marks. Add a static variable schoolName. Create static method changeSchoolName() to update schoolName. Demonstrate how the static variable is shared among all objects.

Main.java × Student.java

```java
package week9_10;

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

        Student student1 = new Student( rollNo: 508,  name: "Shaheer",  marks: 95.5);
        Student student2 = new Student( rollNo: 548,  name: "Shafin",  marks: 89.0);

        System.out.println("Initial Details of All Students\n");
        student1.displayDetails();
        student2.displayDetails();

        System.out.println();

        Student.changeSchoolName( newName: "JMI");

        System.out.println("Details After Updating School Name\n");
        student1.displayDetails();
        student2.displayDetails();
    }
}
```

WEEK 9

Main.java Student.java

```java
package week9_10;

class Student {  5 usages
    int rollNo;  2 usages
    String name;  2 usages
    double marks;  2 usages
    static String schoolName = "AMU";  2 usages

    public Student(int rollNo, String name, double marks) {  2 usages
        this.rollNo = rollNo;
        this.name = name;
        this.marks = marks;
    }

    public static void changeSchoolName(String newName) { schoolName = newName; }

    public void displayDetails() {  4 usages
        System.out.println("Roll No: " + this.rollNo);
        System.out.println("Name: " + this.name);
        System.out.println("Marks: " + this.marks);
        System.out.println("School Name: " + schoolName);
        System.out.println("-------------------------");
    }
}

```

```
Initial Details of All Students

Roll No: 508
Name: Shaheer
Marks: 95.5
School Name: AMU
-------------------------
Roll No: 548
Name: Shafin
Marks: 89.0
School Name: AMU
-------------------------
```

```
Details After Updating School Name

Roll No: 508
Name: Shaheer
Marks: 95.5
School Name: JMI
-------------------------
Roll No: 548
Name: Shafin
Marks: 89.0
School Name: JMI
-------------------------
```

WEEK 10

1. Create class Person (Data Member- name, phone). Create two member inner classes Address (Data Member- House_No, Street, City, State; Method- displayAddr()) and DateOfBirth (Data Member- Day, Month,Year; Method- displayDOB()). Display() is the method of Person class which will display name, address and date of birth of a Person object.

Main.java Person.java

```java
package week10_1;

class Person {  2 usages
    String name;  2 usages
    String phone;  2 usages
    private Address address;  2 usages
    private DateOfBirth dob;  2 usages

    public Person(String name, String phone, String houseNo, String street,
                  String city, String state, int day, int month, int year) {
        this.name = name;
        this.phone = phone;
        this.address = new Address(houseNo, street, city, state);
        this.dob = new DateOfBirth(day, month, year);
    }
    public void Display() {  1 usage
        System.out.println("--- Person Details ---");
        System.out.println("Name: " + this.name);
        System.out.println("Phone: " + this.phone);

        this.address.displayAddr();
        this.dob.displayDOB();
        System.out.println("----------------------");
    }
    class Address {  2 usages
        String House_No;  2 usages
        String Street;  2 usages
        String City;  2 usages
        String State;  2 usages

        public Address(String houseNo, String street, String city, String state) {
            this.House_No = houseNo;
            this.Street = street;
            this.City = city;
            this.State = state;
        }

        public void displayAddr() {  1 usage
            System.out.println("Address: " + this.House_No + ", " +
                    this.Street + ", " + this.City + ", " + this.State);
        }
    }
    class DateOfBirth {  2 usages
        int Day;  2 usages
        int Month;  2 usages
        int Year;  2 usages

        public DateOfBirth(int day, int month, int year) {  1 usage
            this.Day = day;
            this.Month = month;
            this.Year = year;
        }

        public void displayDOB() {  1 usage
            System.out.println("Date of Birth: " +
                    this.Day + "/" + this.Month + "/" + this.Year);
        }
    }
}
```

WEEK 10

Main.java | Person.java

```java
package week10_1;

public class Main {
    public static void main(String[] args) {
        Person person1 = new Person(
                name: "Amit Kumar",
                phone: "+91-9876543210",
                houseNo: "123-B",
                street: "Kamla Nagar",
                city: "Agra",
                state: "Uttar Pradesh",
                day: 15,
                month: 8,
                year: 1990
        );
        person1.Display();
    }
}
```

```
--- Person Details ---
Name: Amit Kumar
Phone: +91-9876543210
Address: 123-B, Kamla Nagar, Agra, Uttar Pradesh
Date of Birth: 15/8/1990
----------------------
```

2. Create class Edible. Within that define two static classes Fruit and Vegetable. Fruit class will have two methods- fruitDetails() is a static method and fruitPackaging() is a non-static method. Vegetable class also has similar methods - vegetableDetails() and vegetablePackaging(). Call all the four methods from main method.

WEEK 10

Main.java × Edible.java

```java
package week10_2;

public class Main {

    public static void main(String[] args) {

        System.out.println("--- Demonstrating Fruit Class ---");
        Edible.Fruit.fruitDetails();

        Edible.Fruit myFruit = new Edible.Fruit();
        myFruit.fruitPackaging();

        System.out.println("\n--- Demonstrating Vegetable Class ---");
        Edible.Vegetable.vegetableDetails();

        Edible.Vegetable myVegetable = new Edible.Vegetable();
        myVegetable.vegetablePackaging();
    }
}
```

Main.java Edible.java ×

```java
package week10_2;

public class Edible {  6 usages

    public static class Fruit {  3 usages
        public static void fruitDetails() {  1 usage
            System.out.println("Static Method (Fruit)");
        }
        public void fruitPackaging() {  1 usage
            System.out.println("Non-Static Method (Fruit)");
        }
    }
    public static class Vegetable {  3 usages
        public static void vegetableDetails() {  1 usage
            System.out.println("Static Method (Vegetable)");
        }
        public void vegetablePackaging() {  1 usage
            System.out.println("Non-Static Method (Vegetable)");
        }
    }
}
```

```
--- Demonstrating Fruit Class ---
Static Method (Fruit)
Non-Static Method (Fruit)

--- Demonstrating Vegetable Class ---
Static Method (Vegetable)
Non-Static Method (Vegetable)
```

WEEK 10

3. Create three different minMaxAdd() methods to calculate minimum, maximum and addition of integers, real numbers and characters.

Main.java

```java
package week10_3;

public class Main {
    public static void minMaxAdd(int a, int b) {  1 usage
        System.out.println("Integer");
        System.out.println("Min: " + Math.min(a, b));
        System.out.println("Max: " + Math.max(a, b));
        System.out.println("Sum: " + (a + b));
    }
    public static void minMaxAdd(double a, double b) {  1 usage
        System.out.println("Double (Real)");
        System.out.println("Min: " + Math.min(a, b));
        System.out.println("Max: " + Math.max(a, b));
        System.out.println("Sum: " + (a + b));
    }
    public static void minMaxAdd(char a, char b) {  1 usage
        System.out.println("Character");
        System.out.println("Min: '" + (char)Math.min(a, b) + "'");
        System.out.println("Max: '" + (char)Math.max(a, b) + "'");
        System.out.println("Sum (as int): " + (a + b));
    }
    public static void main(String[] args) {
        minMaxAdd( a: 10,  b: 20);
        minMaxAdd( a: 7.5,  b: 3.2);
        minMaxAdd( a: 'B',  b: 'X');
    }
}
```

```
Integer
Min: 10
Max: 20
Sum: 30
```

```
Double (Real)
Min: 3.2
Max: 7.5
Sum: 10.7
```

```
Character
Min: 'B'
Max: 'X'
Sum (as int): 154
```

4. Create a class ObjectOriented which has methods- abstraction(), polymorphism() and inheritance(). Create a class JavaLanguage which inherits from ObjectOriented class and has its own methods- persistence() and interfaces(). Create an object of JavaLanguage class to access all of its own and parent's methods.

5. In previous question, create a new class C++ which also inherits from ObjectOriented class and has its own methods- template() and friendFunction().Create an object of C++ class to access all of its own and parent's methods.

WEEK 10

Main.java | JavaLanguage.java | ObjectOriented.java | Cpp.java

```java
package week10_4;

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

        JavaLanguage myJava = new JavaLanguage();
        System.out.println("Accessing Parent's Methods (from ObjectOriented)");
        myJava.abstraction();
        myJava.polymorphism();
        myJava.inheritance();

        System.out.println("\nAccessing Own Methods (from JavaLanguage)");
        myJava.persistence();
        myJava.interfaces();

        Cpp myCpp = new Cpp();
        System.out.println("\nAccessing Parent's Methods (from ObjectOriented)");
        myCpp.abstraction();
        myCpp.polymorphism();
        myCpp.inheritance();

        System.out.println("\nAccessing Own Methods (from CppLanguage)");
        myCpp.template();
        myCpp.friendFunction();
    }
}
```

.java | ObjectOriented.java | JavaLanguage.

```java
package week10_4;

class ObjectOriented {  2 usages  2 inheritors

    public void abstraction() {  2 usages
        System.out.println("Abstraction");
    }

    public void polymorphism() {  2 usages
        System.out.println("Polymorphism");
    }

    public void inheritance() {  2 usages
        System.out.println("Inheritance");
    }
}
```

.java | ObjectOriented.java | JavaLanguag

```java
package week10_4;

class JavaLanguage extends ObjectOriented {

    public void persistence() {  1 usage
        System.out.println("Persistence");
    }

    public void interfaces() {  1 usage
        System.out.println("Interfaces");
    }
}
```

WEEK 10

Main.java | ObjectOriented.java | JavaLanguage.java | Cpp.java

```
1   package week10_4;
2
3   class Cpp extends ObjectOriented {  2 usages
4       public void template() { System.out.println("Templates"); }
7       public void friendFunction() { System.out.println("Friend Functions"); }
10  }
11
```

```
Accessing Parent's Methods (from ObjectOriented)
Abstraction
Polymorphism
Inheritance

Accessing Own Methods (from JavaLanguage)
Persistence
Interfaces

Accessing Parent's Methods (from ObjectOriented)
Abstraction
Polymorphism
Inheritance

Accessing Own Methods (from CppLanguage)
Templates
Friend Functions
```

6. Create class University which has data members- name and ranking. Create class Faculty that extends University class has data member- name and method- Details(). Create a new class Department which is derived from Faculty and has data member- name, chairman and method- Details() and Display() where Display() method calls Details() methods of both Faculty and Department class in its body. Create an object of Department class to Display() method and University ranking.

WEEK 10

Main.java × University.java Faculty.java Department.java

```java
package week10_6;

public class Main {
    public static void main(String[] args) {
        Department csDept = new Department(
                uniName: "Stanford University",
                ranking: 1,
                facultyName: "School of Engineering",
                deptName: "Computer Science",
                chairman: "Dr. John Hennessy"
        );
        System.out.println("Calling Display():");
        csDept.Display();
        System.out.println("\nAccessing University Ranking directly:");
        System.out.println("Ranking: " + csDept.ranking);
    }
}
```

Main.java University.java × Faculty.java Dep

```java
package week10_6;

class University {  1 usage  2 inheritors
    String name;  2 usages
    int ranking;  2 usages

    public University(String name, int ranking) {
        this.name = name;
        this.ranking = ranking;
    }
}
```

Main.java × University.java Faculty.java × Department.java

```java
package week10_6;

class Faculty extends University {  1 usage  1 inheritor
    String name;  2 usages

    public Faculty(String uniName, int ranking, String facultyName) {
        super(uniName, ranking);
        this.name = facultyName;
    }
    public void Details() {  2 usages  1 override
        System.out.println("University: " + super.name);
        System.out.println("Faculty: " + this.name);
    }
}
```

WEEK 10

Main.java | University.java | Faculty.java | Department.java

```java
package week10_6;

class Department extends Faculty {  2 usages
    String name;  2 usages
    String chairman;  2 usages

    public Department(String uniName, int ranking, String facultyName,  1 usage
                      String deptName, String chairman) {

        super(uniName, ranking, facultyName);

        this.name = deptName;
        this.chairman = chairman;
    }
    @Override  2 usages
    public void Details() {
        System.out.println("Department: " + this.name);
        System.out.println("Chairman: " + this.chairman);
    }
    public void Display() {  1 usage
        super.Details();
        this.Details();
    }
}
```

```
Calling Display():
University: Stanford University
Faculty: School of Engineering
Department: Computer Science
Chairman: Dr. John Hennessy

Accessing University Ranking directly:
Ranking: 1
```

7. Create a class Employee (Data Members – empName, empId). Create two member inner classes: ● Salary (Data Members – basic, hra, pf; Method – displaySalary() to print salary details) ● JoiningDate (Data Members – day, month, year; Method – displayJoiningDate() to print joining date). In the Employee class, create a method displayEmployee() that prints the employee's name, ID, salary details, and joining date.

WEEK 10

Main.java × Employee.java

```java
package week10_7;

public class Main {
    public static void main(String[] args) {
        Employee emp1 = new Employee( empName: "Rajesh Kumar",
                empId: 101, basic: 80000, hra: 20000,
                pf: 8000, day: 14, month: 10, year: 2021);
        emp1.displayEmployee();
    }
}

```

Main.java Employee.java ×

```java
package week10_7;

class Employee {
    String empName;
    int empId;
    private Salary salary;
    private JoiningDate joiningDate;
    public Employee(String empName, int empId, double basic,
                    double hra, double pf, int day, int month, int year) {
        this.empName = empName;
        this.empId = empId;
        this.salary = new Salary(basic, hra, pf);
        this.joiningDate = new JoiningDate(day, month, year);
    }
    public void displayEmployee() {
        System.out.println("Employee Name: " + this.empName);
        System.out.println("Employee ID: " + this.empId);
        this.salary.displaySalary();
        this.joiningDate.displayJoiningDate();
    }
```

WEEK 10

```
    class Salary {
        double basic;
        double hra;
        double pf;

        public Salary(double basic, double hra, double pf) {
            this.basic = basic;
            this.hra = hra;
            this.pf = pf;
        }
        public void displaySalary() {
            double total = (this.basic + this.hra) - this.pf;
            System.out.println("Salary Details for: " + empName);
            System.out.println("Basic Pay: " + this.basic);
            System.out.println("HRA: " + this.hra);
            System.out.println("PF: " + this.pf);
            System.out.println("Total Salary: " + total);
        }
    }
    class JoiningDate {
        int day;
        int month;
        int year;
        public JoiningDate(int day, int month, int year) {
            this.day = day;
            this.month = month;
            this.year = year;
        }
        public void displayJoiningDate() {
            System.out.println("Joining Date: " + this.day + "/" + this.month + "/" + this.year);
        }
    }
}
```

```
Employee Name: Rajesh Kumar
Employee ID: 101
Salary Details for: Rajesh Kumar
Basic Pay: 80000.0
HRA: 20000.0
PF: 8000.0
Total Salary: 92000.0
Joining Date: 14/10/2021
```

8. Create a class Shape with overloaded methods area():

- area(int side)– calculates area of a square.
- area(int length, int breadth) – calculates area of a rectangle.
- area(double radius) – calculates area of a circle.

WEEK 10

```java
package week10_8;

public class Main {
    public static void main(String[] args) {
        Shape myShape = new Shape();
        myShape.area( side: 10);
        myShape.area( length: 20,  breadth: 10);
        myShape.area( radius: 7.5);
    }
}
```

```java
package week10_8;

class Shape {
    public void area(int side) {
        int result = side * side;
        System.out.println("Area of the Square: " + result);
    }
    public void area(int length, int breadth) {
        int result = length * breadth;
        System.out.println("Area of the Rectangle: " + result);
    }
    public void area(double radius) {
        double result = Math.PI * radius * radius;
        System.out.printf("Area of the Circle: %.2f\n", result);
    }
}
```

```
Area of the Square: 100
Area of the Rectangle: 200
Area of the Circle: 176.71
```

9. Create a class Vehicle with a method run(). Create subclasses Bike and Car that override the run() method. In the main() method, use a reference of Vehicle to call run() for objects of Bike and Car.

```java
package week10_9;

public class Main {
    public static void main(String[] args) {
        Vehicle v1 = new Bike();
        Vehicle v2 = new Car();
        v1.run();
        v2.run();
    }
}
```

```java
package week10_9;

class Vehicle {  4 usages  2 inheritors
    public void run() {  2 usages  2 overrides
        System.out.println("Vehicle is running.");
    }
}
```

WEEK 10

Main.java | Vehicle.java | Bike.java | Car.jav

```java
package week10_9;

class Bike extends Vehicle {  1 usage
    @Override  2 usages
    public void run() {
        System.out.println("Bike is running.");
    }
}
```

Main.java | Vehicle.java | Bike.java | Car.ja

```java
package week10_9;

class Car extends Vehicle {  1 usage
    @Override  2 usages
    public void run() {
        System.out.println("Car is running.");
    }
}

```

```
Bike is running.
Car is running.
```

WEEK 11

1. Create an interface Account having methods- deposit(), withdraw() and aboutBank() (abountBank() is a static method). Create two classes Saving and Current which implement the Account interface. Call the methods of Saving and Current classes in main method.

2. In the previous question, create a new method in Account interface- takeLoan() (takeLoan() is a default method). takeLoan() method would be implemented by Saving class only. Call the methods of Saving and Current classes in main method.

Main.java | Account.java | Current.java | Saving.java

```java
package week11_1;

public class Main {
    public static void main(String[] args) {
        Saving saving = new Saving();
        Current current = new Current();

        saving.deposit();
        saving.withdraw();
        saving.takeLoan();
        current.deposit();
        current.withdraw();
    }
}
```

Main.java | Account.java | Current.java | Saving.java

```java
package week11_1;

public interface Account {  3 usages  2 implementations
    void deposit();  2 usages  2 implementations
    void withdraw();  2 implementations
    static void aboutBank() { System.out.println("This is a bank."); }
    default void takeLoan() { System.out.println("You are eligible for a loan."); }
}
```

WEEK 11

Main.java | Account.java | Current.java | Saving.java

```java
package week11_1;

public class Current implements Account{ 2 usages

    @Override 2 usages
    public void deposit() { System.out.println("Depositing in Current Account."); }

    @Override
    public void withdraw() { System.out.println("Withdrawing from Current Account."); }
}
```

Main.java | Account.java | Current.java | Saving.java

```java
package week11_1;

public class Saving implements Account{ 2 usages

    @Override 2 usages
    public void deposit() { System.out.println("Depositing in Saving Account."); }

    @Override
    public void withdraw() { System.out.println("Withdrawing from Saving Account."); }

    @Override 2 usages
    public void takeLoan() { Account.super.takeLoan(); }
}
```

```
Depositing in Saving Account.
Withdrawing from Saving Account.
You are eligible for a loan.
Depositing in Current Account.
Withdrawing from Current Account.
```

3. Create interfaces Bike and Scooty, both of which have two methods- offer() and details() (details() is default method). Create a new class BuySomething which implements both interfaces. To remove ambiguity, create a method details() in BuySomething class as well in which call the details() method of both interfaces. Call the methods of BuySomething class in main method.

WEEK 11

Main.java | Bike.java | Scooty.java | BuySomething.java

```java
package week11_3;

public class Main {
    public static void main(String[] args) {
        BuySomething buySomething = new BuySomething();
        buySomething.details();
        buySomething.offer();
    }
}
```

Main.java | Bike.java | Scooty.java | BuySomething.java

```java
package week11_3;

public interface Bike {
    void offer();
    default void details() { System.out.println("This bike costs 50k"); }
}
```

Main.java | Bike.java | Scooty.java | BuySomething.java

```java
package week11_3;

public interface Scooty { 2 usages  1 implementation
    void offer();  1 usage  1 implementation
    default void details() { System.out.println("This scooty costs 45k"); }
}
```

Main.java | Bike.java | Scooty.java | BuySomething.java

```java
package week11_3;

public class BuySomething implements Bike,Scooty {
    @Override
    public void offer() { System.out.println("50 percent off"); }

    @Override
    public void details() {
        Bike.super.details();
        Scooty.super.details();
    }
}
```

```
This bike costs 50k
This scooty costs 45k
50 percent off
```

WEEK 11

4. Create two interfaces Printer and Scanner, both having methods connect() and details() (details() is a default method). Create a class MultiFunctionMachine that implements both interfaces. In MultiFunctionMachine, override the details() method to resolve ambiguity and call the details() methods of both interfaces. Call all methods of MultiFunctionMachine in the main() method.

Main.java

```java
package week11_4;

public class Main {
    public static void main(String[] args) {
        MultiFunctionMachine mf = new MultiFunctionMachine();
        mf.connect();
        mf.details();
    }
}
```

Printer.java

```java
package week11_4;

public interface Printer {  2 usages  1 implementation
    void connect();  1 usage  1 implementation
    default void details() { System.out.println("This machine can print"); }
}
```

Scanner.java

```java
package week11_4;

public interface Scanner {
    void connect();  1 usage  1 implementation
    default void details() { System.out.println("This machine can scan"); }
}
```

MultiFunctionMachine.java

```java
package week11_4;

public class MultiFunctionMachine implements Printer, Scanner{  2 usa
    @Override  2 usages
    public void connect() { System.out.println("Connecting..."); }

    @Override
    public void details() {
        Printer.super.details();
        Scanner.super.details();
    }
}
```

```
Connecting...
This machine can print
This machine can scan
```

WEEK 11

5. Create an interface Device with a method powerOn(). Create another interface SmartDevice that extends Device and adds a method connectWiFi(). Create a class SmartPhone that implements SmartDevice. Demonstrate calling both powerOn() and connectWiFi() using a SmartPhone object in the main() method.

Main.java | Device.java | SmartDevice.java | Sn

```
1   package week11_5;
2
3   public class Main {
4       public static void main(String[] args) {
5           SmartPhone smartPhone = new SmartPhone();
6           smartPhone.powerOn();
7           smartPhone.connectWifi();
8       }
9   }
```

Main.java | Device.java

```
1   package week11_5;
2
3   public interface Device {
4       void powerOn();  1 usage
5   }
```

Main.java | Device.java | SmartDevice.java

```
1   package week11_5;
2
3   public interface SmartDevice extends Device{
4       void connectWifi();  1 usage  1 implementation
5   }
```

Main.java | Device.java | SmartDevice.java | SmartPhone.java

```
1   package week11_5;
2
3   public class SmartPhone implements SmartDevice{  2 usages
4
5       @Override  1 usage
6       public void connectWifi() { System.out.println("Connecting..."); }
9
10      @Override  1 usage
11      public void powerOn() { System.out.println("Booting..."); }
14  }
```

```
Booting...
Connecting...
```

WEEK 12

1. Write a program that calls a method that throws an exception of type ArithmeticException in a for loop at an undesirable situation (such as divide by zero or taking square root of negative number). Catch the exception and display appropriate message. (Example of Unhecked Exception).

Main.java

```java
package week12_1;

public class Main {
    public static int divide(int dividend, int divisor) {  1 usage
        return dividend / divisor;
    }
    public static void main(String[] args) {
        for (int i = 3; i >= -3; i--) {

            try {
                int result = divide( dividend: 100, i);
                System.out.println("Result of 100 / " + i + " = " + result);
            } catch (ArithmeticException e) {
                System.out.println("Error: Cannot divide by zero. Skipping i = " + i);
            }
        }
        System.out.println("Loop finished.");
    }
}
```

```
Result of 100 / 3 = 33
Result of 100 / 2 = 50
Result of 100 / 1 = 100
Error: Cannot divide by zero. Skipping i = 0
Result of 100 / -1 = -100
Result of 100 / -2 = -50
Result of 100 / -3 = -33
Loop finished.
```

2. Write a program of your choice where a Checked Exception occurs at third function but handled at the first calling function. Use both ways of managing Checked Exception i.e. using try-catch block and throws keyword.

WEEK 12

Main.java

```java
package week12_2;

import java.io.FileInputStream;
import java.io.IOException;

public class Main {
    public void f1() {  1 usage
        System.out.println("f1: Calling f2() (Inside try-catch block)");
        try {
            f2();
        } catch (IOException e) {
            System.out.println("f1: Exception was caught!");
            System.out.println("f1: Error Details: " + e.getMessage());
        }
        System.out.println("f1: Complete Execution.");
    }
    public void f2() throws IOException {  1 usage
        System.out.println("  f2: Calling f3()");
        f3();
        System.out.println("  f2: This Line Not Executed.");
    }
    public void f3() throws IOException {  1 usage
        System.out.println("    f3: Trying to read a non-existent file");
        FileInputStream fis = new FileInputStream( name: "random.txt");
        System.out.println("    f3: This Line Not Executed.");
    }
    public static void main(String[] args) {
        Main demo = new Main();
        demo.f1();
    }
}
```

```
f1: Calling f2() (Inside try-catch block)
  f2: Calling f3()
    f3: Trying to read a non-existent file
f1: Exception was caught!
f1: Error Details: random.txt (The system cannot find the file specified)
f1: Complete Execution.
```

WEEK 12

3. You are developing an online banking system where users can transfer money between accounts. If a user tries to withdraw more money than is available in their account, an InsufficientFundsException should be thrown.

Main.java

```java
package week12_3;

public class Main {
    public static void main(String[] args) {
        BankAccount account = new BankAccount( initialBalance: 1000.0);
        try {
            account.withdraw( amount: 500.0);
            account.withdraw( amount: 800.0);
            System.out.println("This message will not be seen");

        } catch (InsufficientFundsException e) {
            System.out.println("Error Caught: " + e.getMessage());
        }
    }
}
```

BankAccount.java

```java
package week12_3;

class InsufficientFundsException extends Exception { 3 usages
    public InsufficientFundsException(String message) { 1 usage
        super(message);
    }
}
class BankAccount { 2 usages
    private double balance; 6 usages

    public BankAccount(double initialBalance) { 1 usage
        this.balance = initialBalance;
        System.out.println("Account created with balance: " + this.balance);
    }

    public void withdraw(double amount) throws InsufficientFundsException {
        System.out.println("Attempting to withdraw: " + amount);

        if (amount > this.balance) {
            throw new InsufficientFundsException("Withdrawal failed. Balance:" + this.balance);
        }
        this.balance -= amount;
        System.out.println("Success. New balance: " + this.balance);
    }
}
```

```
Account created with balance: 1000.0
Attempting to withdraw: 500.0
Success. New balance: 500.0
Attempting to withdraw: 800.0
Error Caught: Withdrawal failed. Balance:500.0
```

WEEK 12

4. Create a user-defined exception InvalidAgeException when the age of a person is below 18 years. Use this exception at appropriate place.

Main.java

```
1   package week12_4;
2
3   class InvalidAgeException extends Exception {  4 usages
4       public InvalidAgeException(String message) {  1 usage
5           super(message);
6       }
7   }
8   public class Main {
9       public static void validateVoter(int age) throws InvalidAgeException {  2 usag
10          if (age < 18) {
11              throw new InvalidAgeException("You must be 18 or older to vote.");
12          }
13          System.out.println("Your vote is registered.");
14      }
15      public static void main(String[] args) {
16          try {
17              validateVoter( age: 25);
18          } catch (InvalidAgeException e) {
19              System.out.println("Error Caught: " + e.getMessage());
20          }
21          System.out.println();
22          try {
23              validateVoter( age: 15);
24          } catch (InvalidAgeException e) {
25              System.out.println("Error Caught: " + e.getMessage());
26          }
27      }
28  }
```

```
Your vote is registered.

Error Caught: You must be 18 or older to vote.
```