1. **Write a program to find the cube of given number using function interface.**

```
import java.util.function.Function;

public class CubeCalculator {
   public static void main(String[] args) {
      Function<Integer, Integer> cube = x -> x * x * x;
      int number = 3; // Example number
      System.out.println("Cube: " + cube.apply(number));
   }
}
```

2. **Define class EmailId with members ,username and password. Define default and parameterized constructors. Accept values from the command line Throw user defined exceptions – "InvalidUsernameException" or "InvalidPasswordException" if the username and password are invalid**

```
class InvalidUsernameException extends Exception {
   public InvalidUsernameException(String message) {
      super(message);
   }
}

class InvalidPasswordException extends Exception {
   public InvalidPasswordException(String message) {
      super(message);
   }
}

class EmailId {
   String username;
```

```
    String password;

    public EmailId(String username, String password) throws
InvalidUsernameException, InvalidPasswordException {
        if (!username.matches("[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}")) {
            throw new InvalidUsernameException("Invalid Username");
        }
        if (password.length() < 6) {
            throw new InvalidPasswordException("Invalid Password");
        }
        this.username = username;
        this.password = password;
    }

    public static void main(String[] args) {
        try {
            EmailId email = new EmailId(args[0], args[1]);
            System.out.println("Email created: " + email.username);
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

3. **Write a program to calculate perimeter and area of rectangle. (hint : area = length * breadth , perimeter=2*(length+breadth))**

```
class Rectangle{
    public static void main(String[]  args){
        int length = 10, breadth = 5;
```

```
        int area = length * breadth;
        int perimetre = 2*(length*breadth);
        System.out.println("Area: " + area);
        System.out.println("Perimetre: " + perimetre);
    }
}
```

4. **Design a screen to handle the Mouse Events such as MOUSE_MOVED and MOUSE_CLICK and display the position of the Mouse_Click in a TextField.**

```
import java.awt.*;
import java.awt.event.*;

public class MouseEventExample {
    public static void main(String[] args) {
        Frame frame = new Frame("Mouse Event Example");
        TextField textField = new TextField();
        textField.setBounds(20, 50, 200, 30);
        frame.add(textField);

        frame.addMouseListener(new MouseAdapter() {
            public void mouseClicked(MouseEvent e) {
                textField.setText("Mouse Clicked at: " + e.getX() + ", " + e.getY());
            }
        });

        frame.setSize(300, 300);
        frame.setLayout(null);
        frame.setVisible(true);
    }
}
```

5. **Write a java program to accept 5 numbers using command line arguments sort and display them.**

```
import java.util.Arrays;


class Code{
   public static void main(String[] args){
      int[] numbers = new int[args.length];
      for(int i = 0; i<args.length; i++){
         numbers[i] = Integer.parseInt(args[i]);
      }
      Arrays.sort(numbers);
      System.out.println("Sorted NUmbers: " + Arrays.toString(numbers));
   }
}
```

6. **Write a menu driven program to perform the following operations on multidimensional array ie matrix : [20] i. Addition ii. Multiplication iii. Transpose of any matrix. iv. Exit**

```
import java.util.Scanner;

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

```
        Scanner sc = new Scanner(System.in);
        int[][] matrixA = {{1, 2}, {3, 4}}; // Example matrix
        int[][] matrixB = {{5, 6}, {7, 8}}; // Example matrix
        int choice;

        do {
            System.out.println("1. Addition\n2. Multiplication\n3. Transpose\n4. Exit");
            choice = sc.nextInt();

            switch (choice) {
                case 1:
                    // Addition Logic
                    break;
                case 2:
                    // Multiplication Logic
                    break;
                case 3:
                    // Transpose Logic
                    break;
            }
        } while (choice != 4);
    }
}
```

7. **1. Write a program to accept the array element and display in reverse order.**

```
import java.util.Scanner;

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

```
        Scanner sc = new Scanner(System.in);
        int[] arr = new int[5];
        System.out.println("Enter 5 numbers:");
        for (int i = 0; i < 5; i++) {
            arr[i] = sc.nextInt();
        }

        System.out.print("Reversed: ");
        for (int i = 4; i >= 0; i--) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

8. **Write a Java program to design a screen using Awt that will take a user name and password. If the user name and password are not same, raise an Exception with appropriate message. User can have 3 login chances only. Use clear button to clear the TextFields.**

```
import java.awt.*;
import java.awt.event.*;

public class LoginScreen {
    public static void main(String[] args) {
        Frame frame = new Frame("Login");
```

```
        TextField username = new TextField();
        TextField password = new TextField();
        password.setEchoChar('*');

        Button login = new Button("Login");
        Button clear = new Button("Clear");

        login.addActionListener(e -> {
            if (!username.getText().equals(password.getText())) {
                System.out.println("Exception: Username and password do not match.");
            }
        });

        clear.addActionListener(e -> {
            username.setText("");
            password.setText("");
        });

        frame.setLayout(new FlowLayout());
        frame.add(new Label("Username:"));
        frame.add(username);
        frame.add(new Label("Password:"));
        frame.add(password);
        frame.add(login);
        frame.add(clear);

        frame.setSize(300, 200);
        frame.setVisible(true);
    }
}
```

9. **Define a class MyDate (day, month, year) with methods to accept and display a MyDate object. Accept date as dd, mm, yyyy. Throw user defined exception "InvalidDateException" if the date is invalid. Examples of invalid dates : 03 15 2019, 31 6 2000, 29 2 2021**

```
class InvalidDateException extends Exception {
   public InvalidDateException(String message) {
      super(message);
   }
}

class MyDate {
   int day, month, year;

   public MyDate(int day, int month, int year) throws InvalidDateException {
      if (month < 1 || month > 12 || day < 1 || day > 31) {
         throw new InvalidDateException("Invalid Date");
      }
      if (month == 2 && day > 29) {
         throw new InvalidDateException("Invalid Date");
      }
      this.day = day;
      this.month = month;
      this.year = year;
```

```
    }

    public static void main(String[] args) {
        try {
            MyDate date = new MyDate(29, 2, 2021);
            System.out.println("Date: " + date.day + "/" + date.month + "/" + date.year);
        } catch (InvalidDateException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

10. Write a program to read book information (bookid, bookname, bookprice, bookqty) in file "book.dat". Write a menu driven program to perform the following operationsusing Random access file: [20] i. Search for a specific book by name. ii. Display all book and total cost

```
import java.io.RandomAccessFile;
import java.io.IOException;

public class BookInfo {
    public static void main(String[] args) throws IOException {
        RandomAccessFile raf = new RandomAccessFile("book.dat", "rw");
```

```
        // Add book info logic here
        raf.writeUTF("Book1");
        raf.writeDouble(250.0);
        raf.writeInt(5);
        raf.close();
    }
}
```

11. Accept the names of two files and copy the contents of the first to the second. First file having Book name and Author name in file. Second file having the contents of First file and also add the comment ‘end of file’ at the end

```
import java.io.*;

public class FileCopy {
    public static void main(String[] args) throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader("source.txt"));
        BufferedWriter writer = new BufferedWriter(new FileWriter("destination.txt"));

        String line;
        while ((line = reader.readLine()) != null) {
            writer.write(line);
            writer.newLine();
        }
```

```
        writer.write("end of file");


        reader.close();
        writer.close();
    }
}
```

12. Write a program to read a text file “sample.txt” and display the contents of a file in reverse order and also original contents change the case (display in upper case).

```
import java.io.*;

public class FileReadReverse {
    public static void main(String[] args) throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader("sample.txt"));
        StringBuilder content = new StringBuilder();

        String line;
        while ((line = reader.readLine()) != null) {
            content.append(line.toUpperCase()).append("\n");
        }
        reader.close();

        System.out.println(content.reverse().toString());
```

```
    }
}
```

13. Define a class patient (patient_name, patient_age, patient_oxy_level,patient_HRCT_report). Create an object of patient. Handle appropriate exception while patient oxygen level less than 95% and HRCT scan report greater than 10, then throw user defined Exception "Patient is Covid Positive(+) and Need to Hospitalized" otherwise display its information.

```
class Patient {
    String name;
    int age;
    double oxyLevel;
    double hrctReport;

    public Patient(String name, int age, double oxyLevel, double hrctReport) {
        this.name = name;
        this.age = age;
        this.oxyLevel = oxyLevel;
        this.hrctReport = hrctReport;
    }

    public void checkPatient() throws Exception {
        if (oxyLevel < 95 || hrctReport > 10) {
            throw new Exception("Patient is Covid Positive(+) and Need to Hospitalized");
```

```
        } else {
            System.out.println("Patient is fine.");
        }
    }

    public static void main(String[] args) {
        try {
            Patient patient = new Patient("John", 30, 94, 11);
            patient.checkPatient();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

14. Create an employee class(id,name,deptname,salary). Define a default and parameterized constructor. Use 'this' keyword to initialize instance variables. Keep a count of objects created. Create objects using parameterized constructor and display the object count after each object is created.(Use static member and method). Also display the contents of each object

```
class Employee {
    static int count = 0;
    int id;
    String name;
    String dept;
```

```
    double salary;

    public Employee(int id, String name, String dept, double salary) {
        this.id = id;
        this.name = name;
        this.dept = dept;
        this.salary = salary;
        count++;
        System.out.println("Employee created: " + this.name + ", Total
Employees: " + count);
    }

    public static void main(String[] args) {
        new Employee(1, "Alice", "HR", 50000);
        new Employee(2, "Bob", "IT", 60000);
    }
}
```

15. Define Student class(roll_no, name, percentage) to create n objects of the student class. Accept details from the user for each object. Define a static method "sortStudent" which sorts the array on the basis of percentage.

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

class Student {
    int rollNo;
    String name;
```

```
double percentage;

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

public static void sortStudents(Student[] students) {
   Arrays.sort(students, (s1, s2) -> Double.compare(s2.percentage, s1.percentage));
}

public static void main(String[] args) {
   Scanner sc = new Scanner(System.in);
   Student[] students = new Student[3];

   for (int i = 0; i < 3; i++) {
      System.out.print("Enter Roll No, Name, Percentage: ");
      students[i] = new Student(sc.nextInt(), sc.next(), sc.nextDouble());
   }

   sortStudents(students);
   for (Student s : students) {
```

```
            System.out.println(s.rollNo + " " + s.name + " " + s.percentage);
        }
    }
}
```

16. **Write a program to find the cube of given number using function interface. Define a class CricketPlayer (name,no_of_innings,no_of_times_notout, totatruns, bat_avg). Create an array of n player objects .Calculate the batting average for each player using static method avg(). Define a static sort method which sorts the array on the basis of average. Display the player details in sorted order**

```
class CricketPlayer {
    String name;
    int innings, notOuts, totalRuns;

    public CricketPlayer(String name, int innings, int notOuts, int totalRuns) {
        this.name = name;
        this.innings = innings;
        this.notOuts = notOuts;
        this.totalRuns = totalRuns;
    }

    public double avg() {
        return (double) totalRuns / innings;
```

```
    }

    public static void main(String[] args) {
        CricketPlayer[] players = {
            new CricketPlayer("Player1", 10, 3, 300),
            new CricketPlayer("Player2", 12, 2, 240)
        };

        Arrays.sort(players, (p1, p2) -> Double.compare(p2.avg(), p1.avg()));
        for (CricketPlayer player : players) {
            System.out.println(player.name + ": " + player.avg());
        }
    }
}
```

17. Write a menu driven program to perform the following operations [10] i. Calculate the volume of cylinder. (hint : Volume: π × r2 × h) ii. Find the factorial of given number. iii. Check the number is Armstrong or not. iv. Exit

```
import java.util.Scanner;

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

```
        Scanner sc = new Scanner(System.in);
        int choice;

        do {
            System.out.println("1. Volume of Cylinder\n2. Factorial\n3. Armstrong
Check\n4. Exit");
            choice = sc.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Radius: ");
                    double r = sc.nextDouble();
                    System.out.print("Height: ");
                    double h = sc.nextDouble();
                    double volume = Math.PI * r * r * h;
                    System.out.println("Volume: " + volume);
                    break;
                case 2:
                    System.out.print("Number: ");
                    int num = sc.nextInt();
                    System.out.println("Factorial: " + factorial(num));
                    break;
                case 3:
                    System.out.print("Number: ");
```

```
                num = sc.nextInt();
                System.out.println(num + (isArmstrong(num) ? " is" : " is not") + " an
Armstrong number.");
                break;
            }
        } while (choice != 4);
    }

    static int factorial(int n) {
        return (n == 0) ? 1 : n * factorial(n - 1);
    }

    static boolean isArmstrong(int num) {
        int sum = 0, temp = num, digits = String.valueOf(num).length();
        while (temp != 0) {
            int digit = temp % 10;
            sum += Math.pow(digit, digits);
            temp /= 10;
        }
        return sum == num;
    }
}
```

18. Define an interface "Operation" which has methods area(),volume().Define a constant PI having a value 3.142.Create a class cylinder which implements this interface (members – radius, height) Create one object and calculate the area and volume.

```
interface Operation {
   double area();
   double volume();
}

class Cylinder implements Operation {
   double radius, height;
   static final double PI = 3.142;

   public Cylinder(double radius, double height) {
      this.radius = radius;
      this.height = height;
   }

   @Override
   public double area() {
      return 2 * PI * radius * (radius + height);
   }
```

```
    @Override
    public double volume() {
        return PI * radius * radius * height;
    }

    public static void main(String[] args) {
        Cylinder cylinder = new Cylinder(5, 10);
        System.out.println("Area: " + cylinder.area());
        System.out.println("Volume: " + cylinder.volume());
    }
}
```

19. **Write a program for multilevel inheritance such that country is inherited from continent. State is inherited from country. Display the place, state, country and continent**

```
class Continent {
    String name = "Asia";
}

class Country extends Continent {
    String countryName = "India";
}
```

```
class State extends Country {
   String stateName = "Maharashtra";

   public void display() {
      System.out.println("Place: Mumbai, State: " + stateName + ", Country: " +
countryName + ", Continent: " + name);
   }

   public static void main(String[] args) {
      new State().display();
   }
}
```

20. Write a java program that take input as a person name in the format of first, middle and last name and then print it in the form last, first and middle name, where in the middle name first character is capital letter

```
import java.util.Scanner;

public class NameFormatter {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter full name (first, middle, last): ");
      String[] names = sc.nextLine().split(" ");
```

```
        String formattedName = names[2] + ", " + names[0] + " " + (names.length > 2 ?
    names[1].substring(0, 1).toUpperCase() + names[1].substring(1) : "");
        System.out.println("Formatted Name: " + formattedName);
    }
}
```

21. Define Student class(roll_no, name, percentage) to create n objects of the Student class. Accept details from the user for each object. Define a static method "sortStudent" which sorts the array on the basis of percentage

SAME AS Q15

22. Create an abstract class "order" having members id,description.Create two subclasses "Purchase Order" and "Sales Order" having members customer name and Vendor name respectively.Define methods accept and display in all cases. Create 3 objects each of Purchase Order and Sales Order and accept and display details.

```
abstract class Order {
    int id;
    String description;

    abstract void accept();
    abstract void display();
}

class PurchaseOrder extends Order {
```

```
    String customerName;

    @Override
    void accept() {
        // Accept data logic
    }

    @Override
    void display() {
        System.out.println("Purchase Order: " + customerName);
    }
}

class SalesOrder extends Order {
    String vendorName;

    @Override
    void accept() {
        // Accept data logic
    }

    @Override
    void display() {
```

```
        System.out.println("Sales Order: " + vendorName);
    }
}

public class Main {
    public static void main(String[] args) {
        Order[] orders = {new PurchaseOrder(), new SalesOrder()};
        for (Order order : orders) {
            order.accept();
            order.display();
        }
    }
}
```