

ASSIGNMENT – 1

1. Write a Java program to print your name.

```
public class Name {  
    public static void main(String[] args) { System.out.println("Rittika Paul");  
    }  
}
```

Output

```
java -cp /tmp/cZRviLRkwr Name  
  
Rittika Paul
```

2. Write a Java program to add two numbers.

```
import java.util.Scanner;public class Add {  
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);  
        System.out.print("Enter Two Numbers: ");int a = sc.nextInt();  
        int b = sc.nextInt(); System.out.println("Sum: " + (a+b));  
    }  
}
```

Output

```
java -cp /tmp/esnhBpZPzs Add  
Enter Two Numbers: 45  
4  
Sum: 49
```

3. Write a Java program to change temperature from Celsius to Fahrenheit.

```
import java.util.Scanner;public class CF {  
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);  
        System.out.print("Enter Temperature in Celsius: "); float c = sc.nextFloat();  
        float f = (float)1.8 * c + 32; System.out.println("Temperature in Fahrenheit: " + f);  
    }  
}
```

Output

```
java -cp /tmp/cZRviLRkwr CF  
Enter Temperature in Celsius: 100  
Temperature in Fahrenheit: 212.0
```

4. Write a Java program to change temperature from Fahrenheit to Celsius.

```
import java.util.Scanner;public class FC {  
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);  
        System.out.print("Enter Temperature in Fahrenheit: "); float f = sc.nextFloat(); float  
        c = (f - 32) / (float)1.8; System.out.println("Temperature in Celsius: " + c);  
    }  
}
```

Output

```
java -cp /tmp/cZRviLRkwr FC  
Enter Temperature in Fahrenheit: 44  
Temperature in Celsius: 6.666667
```

5. Write a Java program to find the area and perimeter of a rectangle.

```
import java.util.Scanner;public class APR {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Length and Breadth:
        ");int l = sc.nextInt(); int b = sc.nextInt();
        int area = l * b;
        int perimeter = 2 * (l + b);
        System.out.println("Area: " + area + "\nPerimeter: " + perimeter);
    }
}
```

Output

```
java -cp /tmp/cZRviLRkwr APR
Enter Length and Breadth: 5
7
Area: 35
Perimeter: 24
```

6. Write a Java program to find the area and perimeter of a circle.

```
import java.util.Scanner;public class APC {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Radius: ");
        int r = sc.nextInt();
        float area = (float)3.14 * r * r;
        float perimeter = 2 * (float)3.14 * r;
        System.out.println("Area: " + area + "\nPerimeter: " + perimeter);
    }
}
```

Output

```
java -cp /tmp/L5jfJb6Poo APC
Enter Radius: 6
Area: 113.04
Perimeter: 37.68
```

7. Write a Java program to display whether a number is odd or even.

```
import java.util.Scanner;public class
OddEven {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt();if(n % 2 == 0)
        {
            System.out.println(n + " is Even");
        } else {
            System.out.println(n + " is Odd");
        }
    }
}
```

Output

```
java -cp /tmp/L5jfJb6Poo OddEven
Enter Number: 7
7 is Odd
```

8. Write a Java program to check if a number is Positive or Negative.

```
import java.util.Scanner; public class
PositiveNegative {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt(); if(n >= 0) {
            System.out.println(n + " is Positive");
        } else {
            System.out.println(n + " is Negative");
        }
    }
}
```

Output

```
java -cp /tmp/dFLp2Jc1d9 PositiveNegative
Enter Number: 45
45 is Positive
```

9. Write a Java program to find maximum of three numbers.

```
import java.util.Scanner; public class
MaxOfThree {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter 3 Numbers: ");
        int a = sc.nextInt(); int b =
        sc.nextInt(); int c = sc.nextInt(); if(a
        >=b && b >= c) {
            System.out.println(a + " is Maximum");
        } else if(b >= c && b >= a) { System.out.println(b + " is
        Maximum"); } else {
            System.out.println(c + " is Maximum");
        }
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg MaxOfThree
Enter 3 Numbers: 44 655 88
655 is Maximum
```

10. Write a Java program to swap two numbers.

```
import java.util.Scanner;public class Swap
{
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter 2 Numbers: "); int a = sc.nextInt();
        int b = sc.nextInt();
        System.out.println("Before Swap: A: " + a + " and B: " + b); int t = a;
        a = b; b = t;
        System.out.println("After Swap: A: " + a + " and B: " + b);
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg Swap
Enter 2 Numbers: 2
4
Before Swap: A: 2 and B: 4
After Swap: A: 4 and B: 2
```

11. Write a Java program to convert miles to kilometers.

```
import java.util.Scanner;public class MK {
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter Distance in Miles: "); float m = sc.nextFloat();
        float km = (float)1.609 * m; System.out.println("Distance in Kilometers: " + km);
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg MK
Enter Distance in Miles: 100
Distance in Kilometers: 160.9
```

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

```
import java.util.Scanner;public class
LeapYear {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Year: ");
        int y = sc.nextInt(); boolean flag =
        false;
        if(y % 400 == 0) flag = true;
        else if(y % 100 == 0) flag = false; else if(y % 4 == 0) flag =
        true; else flag = false;
        if(flag) System.out.println("Leap Year"); else System.out.println("Not a
        Leap Year");
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg LeapYear
Enter Year: 2044
Leap Year
```

13. Write a Java program for following grading system.

Note:
 Percentage >= 90% : Grade A, Percentage >= 80% : Grade B, Percentage >= 70% : Grade C, Percentage >= 60% : Grade D,
 Percentage >= 40% : Grade E, Percentage < 40% : Grade F

```
import java.util.Scanner;public class Grade
{
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter Percentage: "); int p = sc.nextInt();
        char grade = 'F';
        if(p >= 90) grade = 'A'; else if(p >= 80) grade =
        'B';
        else if(p >= 70) grade = 'C'; else if(p >= 60) grade
        = 'D'; else if(p >= 40) grade = 'E';
        System.out.println("GRADE: " + grade);
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg Grade
Enter Percentage: 46
GRADE: E
```

14. Write a Java program to check whether a number is divisible by 5 or not.

```
import java.util.Scanner;public class
DivBy5 {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt();
        if(n % 5 == 0) System.out.println(n + " is Divisible by 5."); else System.out.println(n + " is not Divisible
        by 5.");
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg DivBy5
Enter Number: 49
49 is not Divisible by 5.
```

ASSIGNMENT - 2**1. Write a Java program to check whether a number is Buzz or not.**

```
import java.util.Scanner;public class Buzz
{
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt();
        if(n % 7 == 0 || n % 10 == 7) System.out.println(n + " is Buzz Number");
        else System.out.println(n + " is not Buzz Number");
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg Buzz
Enter Number: 70
70 is Buzz Number
```

2. Write a Java program to calculate factorial of 12.

```
public class Factorial12 {
    public static void main(String[] args) {int n = 12
        int fact = 1; while(n > 0) {
            fact *= n; n--;
        }
        System.out.println("Factorial of 12: " + fact);
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg Factorial12
Factorial of 12: 479001600
```

3. Write a Java program for Fibonacci series.

```
import java.util.Scanner;public class
Fibonacci {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number of Terms:
        "); int n = sc.nextInt(); int a = 0;int b =
        1;
        System.out.println("FIBONACCI SERIES :-"); while(n > 0) {
            int c = a + b; System.out.print(a + " ");
            a = b;
            b = c;
            n--;
        }
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg Fibonacci
Enter Number of Terms: 6
FIBONACCI SERIES :-
0 1 1 2 3 5 |
```

4. Write a Java program to reverse a number.

```
import java.util.Scanner;public class
Reverse {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt();int rev = 0;
        while(n > 0) {
            int rem = n % 10; rev = rev * 10 +
            rem; n /= 10;
        }
        System.out.println("Reverse: " + rev);
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg Reverse
Enter Number: 568
Reverse: 865
|
```

5. Admission to a professional course is subject to the following conditions:

- | | |
|--------------------------------|------------------------------------|
| (a) marks in Mathematics >= 60 | (b) marks in Physics >= 50 |
| (c) marks in Chemistry >= 40 | (d) Total in all 3 subjects >= 200 |
- (Or)

Total in Maths & Physics >= 150 Given the marks in the 3 subjects of n (user input) students, write a program to process the applications to list the eligible candidates.

```

import java.util.Scanner; public class
Admission {
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter Number of Students: ");int n =
        sc.nextInt(); int[] p = new int[n]; int[] m = new
        int[n]; int[] c = new int[n];
        boolean[] allowed = new boolean[n]; for(int i = 0; i < n; i++) {
            System.out.print("Enter Marks of Student " + (i+1) + " in Physics: ");

            p[i] = sc.nextInt();
            System.out.print("Enter Marks of Student " + (i+1) + " in Chemistry: ");
            c[i] = sc.nextInt();

            System.out.print("Enter Marks of Student " + (i+1) + " in Mathematics: ")

            m[i] = sc.nextInt();

            if(m[i] >= 60 && p[i] >= 50 && c[i] >= 40 && ((m[i] + p[i] +
            c[i] >= 200) || (m[i] + p[i] >= 150))) allowed[i] = true;
        }
        for(int i = 0; i < n; i++) {
            if(allowed[i]) System.out.println("Student " + (i+1) + " can take
                Admission");
            else System.out.println("Student " + (i+1) + " cannot take
                Admission");
        }
    }
}

```

Output

```

java -cp /tmp/cjOLBwE8Eg AdmissionProcessor
Enter the number of students: 3

Enter details for Student 1:
Marks in Mathematics: 34
Marks in Physics: 55
Marks in Chemistry: 77
Student 1 is eligible for admission based on total in Maths & Physics.

Enter details for Student 2:
Marks in Mathematics: 996
Marks in Physics: 5
Marks in Chemistry: 66
Student 2 is eligible for admission based on total in Maths & Physics.

Enter details for Student 3:
Marks in Mathematics: 65
Marks in Physics: 66
Marks in Chemistry: 77
Student 3 is eligible for admission.

```

6. Write a Java program to find all roots of a quadratic equation.

```

import java.util.Scanner;public class Roots
{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the Coefficient of x^2: ");

```

```

double a = sc.nextDouble();
System.out.print("Enter the Coefficient of x: "); double b = sc.nextDouble();
System.out.print("Enter the Constant Term: "); double c = sc.nextDouble();
    double d = b * b - 4 * a * c;if(d > 0) {
        double root1 = (-b + Math.sqrt(d)) / (2 * a); double root2 = (-b - Math.sqrt(d)) / (2 *
        a); System.out.format("Root 1: %.2f\nRoot 2: %.2f", root1, root2);
    } else if(d == 0) {
        double root = -b / (2 * a);
        System.out.format("Root 1: %.2f\nRoot 2: %.2f", root, root);
    } else {
        double real = (-b / (2 * a));
        double img = (Math.sqrt(-d) / (2 * a)); System.out.format("Root 1: %.2f + %.2fi\n", real, img);
        System.out.format("Root 2: %.2f - %.2fi", real, img);
    }
}
}

```

Output

```

java -cp /tmp/cjOLBwE8Eg QuadraticRoots
Enter the coefficients of the quadratic equation (ax^2 + bx + c):
Enter a: 45
Enter b: 42
Enter c: 66
The quadratic equation has no real roots.
|
```

7. Write a Java program to calculate the sum of natural numbers up to a certain range.

```

import java.util.Scanner;
public class SumNaturalNumbers {
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter the range (up to which natural numbers you want to
        calculate the sum): ");int n = sc.nextInt();
        int sum = 0;
        for(int i = 1; i <= n; i++) {sum += i;}
        System.out.println("Sum of Natural Numbers up to " + n + " : " + sum);
    }
}
```

Output

```

java -cp /tmp/cjOLBwE8Eg SumOfNaturalNumbers
Enter the range (up to which natural numbers you want to calculate the sum): 19
Sum of natural numbers up to 19 is: 190
```

8. Write a Java program to print all multiples of 10 between a given interval.

```

import java.util.Scanner; public class Multiples10
{
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter Starting point of the interval: ");
        int l = sc.nextInt(); System.out.print("Enter ending point of the
        interval: ");int u = sc.nextInt();
        for(int i = l; i <= u; i++) {
            System.out.println("10 X " + i + " : " + (10*i));
        }
}}
```

Output

```
java -cp /tmp/cjOLBwE8Eg MultiplesOfTen
Enter the starting point of the interval: 12
Enter the ending point of the interval: 44
Multiples of 10 between 12 and 44 are:
20
30
40
```

9. Write a Java program to generate multiplication table.

```
import java.util.Scanner;public class
MultiTable {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt();
        for(int i = 1; i <= 10; i++) {
            System.out.println(n + " X " + i + " : " + (n*i));
        }
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg MultiTable
Enter Number: 7
7 X 1 : 7
7 X 2 : 14
7 X 3 : 21
7 X 4 : 28
7 X 5 : 35
7 X 6 : 42
7 X 7 : 49
7 X 8 : 56
7 X 9 : 63
7 X 10 : 70
```

10. Write a Java program to find HCF of two numbers.

```
import java.util.Scanner;public class HCF {
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter Numbers: "); int a = sc.nextInt();
        int b = sc.nextInt(); while(b > 0) {
            int temp = b; b = a % b;
            a = temp;
        }
        System.out.println("HCF : " + a);
    }
}
```

Output

```
java -cp /tmp/cjOLBwE8Eg HCF
Enter Numbers: 723
567
HCF : 9
```

11. Write a Java program to find LCM of two numbers.

```
import java.util.Scanner;public class LCM
{
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter Numbers: "); int a = sc.nextInt();
        int b = sc.nextInt();int max = a;
        int min = b;if(a < b)
        {
            max = b; min =
            a;
        }
        for(int i = max; ; i+= max) { if(i % min == 0)
            System.out.println("LCM : " + i); break;
        }
    }
}
```

| Output |
|---|
| <pre>java -cp /tmp/cjOLBwE8Eg LCM Enter Numbers: 882 2 LCM : 88</pre> |

12. Write a Java program to count the number of digits of an integer.

```
import java.util.Scanner; public class CountDigits
{
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt();int c = 0;
        while(n > 0) {
            c++;
            n/=10;
        }
        System.out.println("Number of Digits: " + c);
    }
}
```

| Output |
|---|
| <pre>java -cp /tmp/6u5PfAVslw CountDigits Enter Number: 236 Number of Digits: 3</pre> |

13. Write a Java program to calculate the exponential of a number.

```
import java.util.Scanner; public class Exponential
{
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt(); System.out.print("\nEnter Power: ");int p =
        sc.nextInt();
        int ans = 1; while(p >
        0) {
            ans = ans * n;p--;
        }
        System.out.println("Exponential: " + ans);}}
```

```
Output
java -cp /tmp/6u5PfAVs1w Exponential
Enter Number: 463

Enter Power: 2
Exponential: 214369
```

14. Write a Java program to check whether a number is palindrome or not.

```
import java.util.Scanner; public class Palindrome
{
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt(); int on = n;
        int rev = 0; while(n >
        0) {
            int rem = n % 10; rev = rev * 10 +
            rem; n /= 10;
        }
        if(on == rev) System.out.println("Number is Palindrome"); else System.out.println("Number is
        not Palindrome");
    }
}
```

```
Output
java -cp /tmp/6u5PfAVs1w Palindrome
Enter Number: 232
Number is Palindrome
```

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

```
import java.util.Scanner;public class Prime
{
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt(); for(int i = 2; i < n;
        i++) {
            if(n % i == 0) {
                System.out.println("Number is not Prime");return;
            }
        }
        System.out.println("Number is Prime");
    }
}
```

```
Output
java -cp /tmp/6u5PfAVs1w Prime
Enter Number: 19
Number is Prime
```

16. Write a Java program to convert a Binary number to Decimal and Decimal toBinary.

```
import java.util.Scanner; public class DecimalBinary
{
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter Number in Decimal: ");int n = sc.nextInt();
        long binaryNumber = 0, remainder, i = 1, step = 1; while (n > 0) {
            remainder = n % 2;n /= 2;
            binaryNumber += remainder * i;i *= 10;
        }
        System.out.println("Binary: " + binaryNumber);System.out.print("Enter Number in
        Binary: "); n = sc.nextInt();
        int decimalNumber = 0;i = 0;
        while (n > 0) {
            remainder = n % 10;n /= 10;
            decimalNumber += remainder * Math.pow(2, i);i++;
        }
        System.out.println("Decimal: " + decimalNumber);
    }
}
```

Output

```
java -cp /tmp/6u5PfAVslw DecimalBinary
Enter Number in Decimal: 154
Binary: 10011010
Enter Number in Binary: 100010
Decimal: 34
```

17. Write a Java program to find median of a set of numbers.

```
import java.util.Scanner;public class
Median {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number of Terms:
        "); int n = sc.nextInt(); int[] arr = new int[n];int sum = 0;
        for(int i = 0; i < n; i++) { System.out.print("Enter Number: ");arr[i] =
            sc.nextInt();
            sum += arr[i];
        }
        if(arr.length % 2 == 0) {
            System.out.println("Median: " + ((arr[arr.length / 2] +arr[arr.length/2 - 1])/2.0));
        } else {
            System.out.println("Median: " + arr[arr.length / 2]);
        }
    }
}
```

Output

```
java -cp /tmp/6u5PfAVslw Median
Enter Number of Terms: 4
Enter Number: 23
Enter Number: 55
Enter Number: 23
Enter Number: 54
Median: 39.0
```

18. Write a Java program to compute the value of Euler's number that is used as the base of natural logarithms. Use the following formula: $e=1 + 1/1! + 1/2! + 1/3! + \dots + 1/n!$

```
import java.util.Scanner; public class EulerNumber {
    public static int fact(int n) { int fact = 1;
        while(n > 0) {
            fact *= n; n--;
        }
        return fact;
    }
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter N: ");
        int n = sc.nextInt(); double e = 1;
        for(int i = 1; i <= n; i++) {
            e += (double)(1/(double)fact(i));
        }
        System.out.println("e : " + e);
    }
}
```

Output

```
java -cp /tmp/QtDE1GXAGm EulerNumber
Enter N: 666
e : Infinity
```

19. Write a Java program to generate all combinations of 1, 2 or 3 using a loop.

```
public class Combinations123 {
    public static void main(String[] args) { for(int i = 1; i <= 3; i++) {
        for(int j = 1; j <= 3; j++) {
            for(int k = 1; k <= 3; k++) {
                if(i != j && j != k && i != k) System.out.println(i + " "
+ j + " " + k);
            }
        }
    }
}
```

Output

```
java -cp /tmp/QtDE1GXAGm Combinations123
1 2 3
1 3 2
2 1 3
2 3 1
3 1 2
3 2 1
```

20. Write a Java program to read two integer values m and n and to decide and print whether m is a multiple of n

```
import java.util.Scanner;public class MN {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter M and N: ");
        int m = sc.nextInt();int n =
        sc.nextInt();
        if(m % n == 0) System.out.println("M is a multiple of N");else System.out.println("M is not a multiple
        of N");
    }
}
```

| Output |
|--|
| <pre>java -cp /tmp/QtDE1GXAGm MN Enter M and N: 86 77 M is not a multiple of N</pre> |

21. Write a Java program to display prime numbers between a given interval.

```
import java.util.Scanner; public class
RangePrime {

    public static boolean prime(int n) { for(int i = 2; i < n; i++) {
        if(n%i==0) return false;
    }
    return true;
}

    public static void main(String[] args){ Scanner sc = new Scanner(System.in);
        System.out.print("Enter Lower Limit: "); int l = sc.nextInt();
        System.out.print("Enter Upper Limit: ");int u = sc.nextInt();
        System.out.println("PRIME NUMBERS :-"); for(int i = l; i <= u; i++) {
            if(prime(i))
                System.out.println(i);
        }
    }
}
```

| Output |
|---|
| <pre>java -cp /tmp/z3LqSozvpp RangePrime Enter Lower Limit: 20 Enter Upper Limit: 58 PRIME NUMBERS :- 23 29 31 37 41 43 47 53</pre> |

22. Write a Java program to check whether a given number is Armstrong or not.

```
import java.util.Scanner;public class
Armstrong {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number: ");
        int n = sc.nextInt();int on = n;
        int ans = 0; while(n >
        0) {
            int rem = n % 10;
            ans = ans + rem * rem * rem; n /= 10;
        }
        if(on == ans) System.out.println("Armstrong Number"); else System.out.println("Not
        an Armstrong Number");
    }
}
```

Output

```
java -cp /tmp/z3LqSozvpp Armstrong
Enter Number: 353
Not an Armstrong Number
```

23. Pattern:

```
1
2 3 4
5 6 7 8 9
```

```
import java.util.Scanner;public class
Pattern1 {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number of Rows:
        ");int n = sc.nextInt(); int c = 1;
        for(int i = 1; i <=n; i++) {
            for(int j = 1; j <= 2 * i - 1; j++) {System.out.print(c +
                " ");
                c++;
            }
            System.out.println();
        }
    }
}
```

Output

```
java -cp /tmp/z3LqSozvpp Pattern1
Enter Number of Rows: 3
1
2 3 4
5 6 7 8 9
```

24. Pattern 2:

```

1
212
32123
4321234
import java.util.Scanner;public class
Pattern2 {
    public static void main(String[] args) { Scanner sc = new
        Scanner(System.in); System.out.print("Enter Number of Rows:
");int n = sc.nextInt(); for(int i = 1; i <= n; i++) {
            for(int sp = 1; sp <= n - i; sp++) {System.out.print(" ");}
            for(int j = i; j >= 1; j--) { System.out.print(j);}
            for(int j = 2; j <= i; j++) { System.out.print(j);}
            System.out.println();
        }
    }
}

```

Output

```

java -cp /tmp/z3LqSozvpp Pattern2
Enter Number of Rows: 4
1
212
32123
4321234
|
```

25. Pattern:

```

1   1
2   2
3   3
4

```

```

import java.util.Scanner;public class
Pattern3 {
    public static void main(String[] args) { Scanner sc = new Scanner(System.in);
        System.out.print("Enter Number of Rows: ");int n = sc.nextInt(); int
        space = 2 * n - 2; for(int i = 1; i <= n; i++)
        {
            for(int sp = 1; sp < i; sp++) {System.out.print(" ");}
            System.out.print(i);
            for(int sp = 1; sp < space; sp++) {System.out.print(" ");}
            space -= 2;
            if(i != n) System.out.print(i);
            System.out.println();
        }
    }
}
```

Output

```
java -cp /tmp/z3LqSozvpp Pattern3
Enter Number of Rows: 4
1   1
2   2
3  3
4
```

ASSIGNMENT 3**1. Java program to calculate Sum & Average of an integer array.**

```
class ArraySumAndAverage {
public static void main(String[] args) {
    int[] arr = {1, 2, 3, 4, 5};
    int sum = 0;

    for (int num : arr) {
        sum += num;
    }
    double average = (double) sum / arr.length;
    System.out.println("Sum: " + sum);
    System.out.println("Average: " + average);
}
```

Output

```
java -cp /tmp/z3LqSozvpp ArraySumAndAverage
Sum: 15
Average: 3.0
```

2. Java program to implement stack using array

```
public class StackArray {
    private static final int MAX_SIZE = 100;
    private int[] stackArray;
    private int top;

    public StackArray() {
        stackArray = new int[MAX_SIZE];
        top = -1;
    }

    public void push(int element) {
        if (top < MAX_SIZE - 1) {
            stackArray[++top] = element;
            System.out.println("Pushed element: " + element);
        } else {
            System.out.println("Stack overflow. Cannot push element.");
        }
    }

    public int pop() {
        if (top >= 0) {
            int poppedElement = stackArray[top--];
            System.out.println("Popped element: " + poppedElement);
            return poppedElement;
        } else {
            System.out.println("Stack underflow. Cannot pop element.");
            return -1; // Return a default value indicating failure
        }
    }
}
```

```

    }
    public boolean isEmpty() {
        return top == -1;
    }
    public int peek() {
        if (!isEmpty()) {
            return stackArray[top];
        } else {
            System.out.println("Stack is empty. Cannot peek.");
            return -1; // Return a default value indicating failure
        }
    }
    public void display() {
        if (!isEmpty()) {
            System.out.print("Stack elements: ");
            for (int i = 0; i <= top; i++) {
                System.out.print(stackArray[i] + " ");
            }
            System.out.println();
        } else {
            System.out.println("Stack is empty.");
        }
    }
}

public static void main(String[] args) {
    StackArray stack = new StackArray();
    stack.push(10);
    stack.push(20);
    stack.push(30);
    stack.display();
    stack.pop();
    stack.display();
    System.out.println("Top element: " + stack.peek());
}
}

```

Output

```

java -cp /tmp/rzXlqFDbIe StackArray
Pushed element: 10
Pushed element: 20
Pushed element: 30
Stack elements: 10 20 30
Popped element: 30
Stack elements: 10 20
Top element: 20
|

```

3. Java program to implement Queue using array

```

public class QueueArray {
    private static final int MAX_SIZE = 100;
    private int[] queueArray;
    private int front, rear, size;

    public QueueArray() {
        queueArray = new int[MAX_SIZE];
        front = rear = -1;
        size = 0;
    }
    public void enqueue(int element) {
        if (size < MAX_SIZE) {
            if (isEmpty()) {
                front = rear = 0;
            } else {
                rear = (rear + 1) % MAX_SIZE;
            }
            queueArray[rear] = element;
            size++;
        }
    }
}

```

```

        System.out.println("Enqueued element: " + element);
    } else {
        System.out.println("Queue overflow. Cannot enqueue element.");
    }
}
public int dequeue() {
    if (!isEmpty()) {
        int dequeuedElement = queueArray[front];
        if (front == rear) {
            front = rear = -1;
        } else {
            front = (front + 1) % MAX_SIZE;
        }
        size--;
        System.out.println("Dequeued element: " + dequeuedElement);
        return dequeuedElement;
    } else {
        System.out.println("Queue underflow. Cannot dequeue element.");
        return -1;
    }
}
public boolean isEmpty() {
    return size == 0;
}
public void display() {
    if (!isEmpty()) {
        System.out.print("Queue elements: ");
        int count = 0;
        int index = front;
        while (count < size) {
            System.out.print(queueArray[index] + " ");
            index = (index + 1) % MAX_SIZE;
            count++;
        }
        System.out.println();
    } else {
        System.out.println("Queue is empty.");
    }
}
public static void main(String[] args) {
    QueueArray queue = new QueueArray();
    queue.enqueue(10);
    queue.enqueue(20);
    queue.enqueue(30);
    queue.display();
    queue.dequeue();
    queue.display();
}
}

```

Output

```

java -cp /tmp/rzXlqFDbIe QueueArray
Enqueued element: 10
Enqueued element: 20
Enqueued element: 30
Queue elements: 10 20 30
Dequeued element: 10
Queue elements: 20 30

```

4. Java program to calculate Sum of two 2-dimensional arrays

```

public class SumOf2DArrays {
    public static void main(String[] args) {
        int[][] matrix1 = {{1, 2, 3}, {4, 5, 6}};
        int[][] matrix2 = {{7, 8, 9}, {10, 11, 12}};
        int rows = matrix1.length;
        int cols = matrix1[0].length;
    }
}

```

```

int[][] sumMatrix = new int[rows][cols];
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        sumMatrix[i][j] = matrix1[i][j] + matrix2[i][j];
    }
}
System.out.println("Sum of matrices:");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        System.out.print(sumMatrix[i][j] + " ");
    }
    System.out.println();
}
}
}

```

Output

```

java -cp /tmp/G0q8iGhWYW SumOf2DArrays
Sum of matrices:
8 10 12
14 16 18

```

5. Java program to find the range of a 1D array:

```

class ArrayRange {
    public static void main(String[] args) {
        int[] arr = {5, 2, 9, 1, 7};
        int min = arr[0];
        int max = arr[0];
        for (int num : arr) {
            if (num < min) {
                min = num;
            }
            if (num > max) {
                max = num;
            }
        }
        int range = max - min;
        System.out.println("Range: " + range);
    }
}

```

Output

```

java -cp /tmp/G0q8iGhWYW ArrayRange
Range: 8

```

6. Java program to search an element in an array

```

class ArraySearch {
    public static void main(String[] args) {
        int[] arr = {4, 2, 7, 1, 9};
        int searchElement = 7;
        boolean found = false;
        for (int num : arr) {
            if (num == searchElement) {
                found = true;
                break;
            }
        }
        if (found) {
            System.out.println("Element found in the array.");
        } else {
            System.out.println("Element not found in the array.");
        }
    }
}

```

```
}
```

Output

```
java -cp /tmp/G0q8iGhWYW ArraySearch
Element found in the array.
```

7. Java program to find the sum of even numbers in an integer array:

```
class SumOfEvenNumbers {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9};
        int sum = 0;

        for (int num : arr) {
            if (num % 2 == 0) {
                sum += num;
            }
        }
        System.out.println("Sum of even numbers: " + sum);
    }
}
```

Output

```
java -cp /tmp/G0q8iGhWYW SumOfEvenNumbers
Sum of even numbers: 20
```

8. Java program to find the sum of diagonal elements in a 2D array:

```
class DiagonalSum {
    public static void main(String[] args) {
        int[][] matrix = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
        int sum = 0;

        for (int i = 0; i < matrix.length; i++) {
            sum += matrix[i][i];
        }
        System.out.println("Sum of diagonal elements: " + sum);
    }
}
```

Output

```
java -cp /tmp/G0q8iGhWYW DiagonalSum
Sum of diagonal elements: 15
```

9. Reverse the elements in an array of integers without using a second array:

```
public class ReverseArrayInPlace {
    public static void main(String[] args) {
        int[] array = {1, 2, 3, 4, 5};
        System.out.println("Original Array:");
        displayArray(array);
        reverseArrayInPlace(array);
        System.out.println("Reversed Array:");
        displayArray(array);
    }

    private static void reverseArrayInPlace(int[] arr) {
        int start = 0;
        int end = arr.length - 1;
```

```

        while (start < end) {
            int temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;
            start++;
            end--;
        }
    }
    private static void displayArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}

```

Output

```

java -cp /tmp/G0q8iGhWYW ReverseArrayInPlace
Original Array:
1 2 3 4 5
Reversed Array:
5 4 3 2 1
|

```

10. Java program to enter n elements in an array and find the smallest number among them

```

import java.util.Scanner;
class SmallestNumberInArray {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the size of the array: ");
        int size = scanner.nextInt();
        int[] arr = new int[size];
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < size; i++) {
            arr[i] = scanner.nextInt();
        }
        int smallest = arr[0];
        for (int num : arr) {
            if (num < smallest) {
                smallest = num;
            }
        }
        System.out.println("Smallest number in the array: " + smallest);
    }
}

```

Output

```

java -cp /tmp/G0q8iGhWYW SmallestNumberInArray
Enter the size of the array: 4
Enter the elements of the array:
4 8 5 7
Smallest number in the array: 4
|

```

11. Java program to find the sum of all odd numbers in a 2D array:

```

class SumOfOddNumbers {
    public static void main(String[] args) {
        int[][] matrix = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
        int sum = 0;

        for (int[] row : matrix) {
            for (int num : row) {
                if (num % 2 != 0) {
                    sum += num;
                }
            }
        }
    }
}

```

```

        }
    }
    System.out.println("Sum of odd numbers: " + sum);
}
}

```

Output

```
java -cp /tmp/G0q8iGhWYW SumOfOddNumbers
Sum of odd numbers: 25
```

12. Java program to print transpose of matrix:

```

public class TransposeMatrix {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };
        System.out.println("Original Matrix:");
        displayMatrix(matrix);
        int[][] transpose = calculateTranspose(matrix);
        System.out.println("Transpose Matrix:");
        displayMatrix(transpose);
    }
    private static int[][] calculateTranspose(int[][] mat) {
        int rows = mat.length;
        int cols = mat[0].length;
        int[][] transpose = new int[cols][rows];
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                transpose[j][i] = mat[i][j];
            }
        }
        return transpose;
    }
    private static void displayMatrix(int[][] mat) {
        for (int i = 0; i < mat.length; i++) {
            for (int j = 0; j < mat[0].length; j++) {
                System.out.print(mat[i][j] + " ");
            }
            System.out.println();
        }
        System.out.println();
    }
}

```

Output

```
java -cp /tmp/G0q8iGhWYW TransposeMatrix
Original Matrix:
1 2 3
4 5 6
7 8 9

Transpose Matrix:
1 4 7
2 5 8
3 6 9
```

13. Java program to check whether a given matrix is sparse or not:

```

class SparseMatrixCheck {
    public static void main(String[] args) {
        int[][] matrix = {{1, 0, 0}, {0, 0, 0}, {0, 0, 3}};
        int rows = matrix.length;

```

```

int cols = matrix[0].length;
int zeroCount = 0;
for (int[] row : matrix) {
    for (int num : row) {
        if (num == 0) {
            zeroCount++;
        }
    }
}
if (zeroCount > (rows * cols) / 2) {
    System.out.println("The matrix is sparse.");
} else {
    System.out.println("The matrix is not sparse.");
}
}
}

```

Output

```

java -cp /tmp/G0q8iGhWYW SparseMatrixCheck
The matrix is sparse.

```

14. Java program to count the prime numbers in an array:

```

class CountPrimeNumbers {
    public static void main(String[] args) {
        int[] arr = {2, 3, 5, 7, 8, 11, 13, 16};
        int count = 0;
        for (int num : arr) {
            if (isPrime(num)) {
                count++;
            }
        }
        System.out.println("Number of prime numbers in the array: " + count);
    }

    private static boolean isPrime(int num) {
        if (num <= 1) {
            return false;
        }
        for (int i = 2; i <= Math.sqrt(num); i++) {
            if (num % i == 0) {
                return false;
            }
        }
        return true;
    }
}

```

Output

```

java -cp /tmp/G0q8iGhWYW CountPrimeNumbers
Number of prime numbers in the array: 6

```

15. Java program to find the second highest element of an array:

```

class SecondHighestElement {
    public static void main(String[] args) {
        int[] arr = {5, 2, 9, 1, 7};
        int highest = Integer.MIN_VALUE;
        int secondHighest = Integer.MIN_VALUE;
        for (int num : arr) {
            if (num > highest) {
                secondHighest = highest;
                highest = num;
            } else if (num > secondHighest && num != highest) {
                secondHighest = num;
            }
        }
        System.out.println("Second highest element: " + secondHighest);
    }
}

```

```

        highest = num;
    } else if (num > secondHighest && num != highest) {
        secondHighest = num;
    }
    System.out.println("Second highest element: " + secondHighest);
}
}

```

Output

```
java -cp /tmp/G0q8iGhWYW SecondHighestElement
Second highest element: 7
```

16. Java program which counts the non-zero elements in an integer array

```

class CountNonZeroElements {
public static void main(String[] args) {
    int[] arr = {0, 5, 0, 3, 8, 0, 2};
    int count = 0;
    for (int num : arr) {
        if (num != 0) {
            count++;
        }
    }
    System.out.println("Number of non-zero elements: " + count);
}
}

```

Output

```
java -cp /tmp/G0q8iGhWYW CountNonZeroElements
Number of non-zero elements: 4
```

17. Java program to merge two float arrays

```

public class MergeFloatArrays {
public static void main(String[] args) {
    float[] array1 = {1.5f, 2.5f, 3.5f};
    float[] array2 = {4.5f, 5.5f, 6.5f};
    float[] mergedArray = mergeArrays(array1, array2);
    System.out.println("Merged Array:");
    displayArray(mergedArray);
}
private static float[] mergeArrays(float[] arr1, float[] arr2) {
    int length1 = arr1.length;
    int length2 = arr2.length;
    float[] mergedArray = new float[length1 + length2];
    System.arraycopy(arr1, 0, mergedArray, 0, length1);
    System.arraycopy(arr2, 0, mergedArray, length1, length2);
    return mergedArray;
}
private static void displayArray(float[] arr) {
    for (float num : arr) {
        System.out.print(num + " ");
    }
    System.out.println();
}
}

```

Output

```
java -cp /tmp/G0q8iGhWYW MergeFloatArrays
Merged Array:
1.5 2.5 3.5 4.5 5.5 6.5
```

18. Java program where elements of two integer arrays get added index-wise and get stored into a third array.

```
public class AddArraysIndexWise {
    public static void main(String[] args) {
        int[] array1 = {1, 2, 3};
        int[] array2 = {4, 5, 6};

        int[] resultArray = addArraysIndexWise(array1, array2);

        System.out.println("Resultant Array:");
        displayArray(resultArray);
    }

    // Method to add elements of two arrays index-wise
    private static int[] addArraysIndexWise(int[] arr1, int[] arr2) {
        int length = Math.min(arr1.length, arr2.length);
        int[] resultArray = new int[length];

        for (int i = 0; i < length; i++) {
            resultArray[i] = arr1[i] + arr2[i];
        }

        return resultArray;
    }

    // Method to display the elements of an array
    private static void displayArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}
```

Output

```
java -cp /tmp/G0q8iGhWYW AddArraysIndexWise
Resultant Array:
5 7 9
```

19. Java program to multiply two matrices:

```
public class MultiplyMatrices {
    public static void main(String[] args) {
        int[][] matrix1 = {
            {1, 2, 3},
            {4, 5, 6}
        };
        int[][] matrix2 = {
            {7, 8},
            {9, 10},
            {11, 12}
        };
        int[][] resultMatrix = multiplyMatrices(matrix1, matrix2);

        System.out.println("Resultant Matrix:");
        displayMatrix(resultMatrix);
```

```

}
private static int[][] multiplyMatrices(int[][] mat1, int[][] mat2) {
    int rows1 = mat1.length;
    int cols1 = mat1[0].length;
    int rows2 = mat2.length;
    int cols2 = mat2[0].length;
    if (cols1 != rows2) {
        System.out.println("Matrices cannot be multiplied.");
        return null;
    }
    int[][] resultMatrix = new int[rows1][cols2];

    for (int i = 0; i < rows1; i++) {
        for (int j = 0; j < cols2; j++) {
            for (int k = 0; k < cols1; k++) {
                resultMatrix[i][j] += mat1[i][k] * mat2[k][j];
            }
        }
    }
    return resultMatrix;
}
private static void displayMatrix(int[][] mat) {
    for (int i = 0; i < mat.length; i++) {
        for (int j = 0; j < mat[0].length; j++) {
            System.out.print(mat[i][j] + " ");
        }
        System.out.println();
    }
}
}

```

Output

```

java -cp /tmp/G0q8iGhWYW MultiplyMatrices
Resultant Matrix:
58 64
139 154
|

```

20. Java program to subtract two matrices

```

public class SubtractMatrices {
    public static void main(String[] args) {
        int[][] matrix1 = {
            {1, 2, 3},
            {4, 5, 6}
        };
        int[][] matrix2 = {
            {7, 8, 9},
            {10, 11, 12}
        };
        int[][] resultMatrix = subtractMatrices(matrix1, matrix2);

        System.out.println("Resultant Matrix:");
        displayMatrix(resultMatrix);
    }
    private static int[][] subtractMatrices(int[][] mat1, int[][] mat2) {
        int rows1 = mat1.length;
        int cols1 = mat1[0].length;
        int rows2 = mat2.length;
        int cols2 = mat2[0].length;
        if (rows1 != rows2 || cols1 != cols2) {
            System.out.println("Matrices cannot be subtracted.");
            return null;
        }

        int[][] resultMatrix = new int[rows1][cols1];
        for (int i = 0; i < rows1; i++) {
            for (int j = 0; j < cols1; j++) {

```

```

        resultMatrix[i][j] = mat1[i][j] - mat2[i][j];
    }
}

return resultMatrix;
}
private static void displayMatrix(int[][] mat) {
    for (int i = 0; i < mat.length; i++) {
        for (int j = 0; j < mat[0].length; j++) {
            System.out.print(mat[i][j] + " ");
        }
        System.out.println();
    }
}
}

```

Output

```

java -cp /tmp/G0q8iGhWYw SubtractMatrices
Resultant Matrix:
-6 -6 -6
-6 -6 -6

```

21. Write a Java program to find duplicate elements in a 1D array and find their frequency of occurrence.

```

public class FindDuplicateElements {
    public static void main(String[] args) {
        int[] array = {4, 2, 8, 3, 4, 2, 7, 8, 1, 9, 2};
        System.out.println("Original Array:");
        displayArray(array);

        findAndDisplayDuplicates(array);
    }
    private static void findAndDisplayDuplicates(int[] arr) {
        int length = arr.length;
        boolean[] visited = new boolean[length];
        System.out.println("Duplicate Elements and Their Frequencies:");
        for (int i = 0; i < length - 1; i++) {
            if (!visited[i]) {
                int count = 1;
                for (int j = i + 1; j < length; j++) {
                    if (arr[i] == arr[j]) {
                        visited[j] = true;
                        count++;
                    }
                }
                if (count > 1) {
                    System.out.println(arr[i] + " occurs " + count + " times");
                }
            }
        }
    }
    private static void displayArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}

```

```

Output
java -cp /tmp/G0q8iGhWYW FindDuplicateElements
Original Array:
4 2 8 3 4 2 7 8 1 9 2
Duplicate Elements and Their Frequencies:
4 occurs 2 times
2 occurs 3 times
8 occurs 2 times
|

```

22. Write a Java program to print every alternate number of a given array.

```

public class PrintAlternateNumbers {
    public static void main(String[] args) {
        int[] array = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        System.out.println("Original Array:");
        displayArray(array);
        System.out.println("Alternate Numbers:");
        printAlternateNumbers(array);
    }
    private static void printAlternateNumbers(int[] arr) {
        for (int i = 0; i < arr.length; i += 2) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }
    private static void displayArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}

```

```

Output
java -cp /tmp/G0q8iGhWYW PrintAlternateNumbers
Original Array:
1 2 3 4 5 6 7 8 9 10
Alternate Numbers:
1 3 5 7 9
|

```

23. Given are two one-dimensional arrays A & B, which are sorted in ascending order. Write a Java program to merge them into single sorted array C that contains every item from arrays A & B, in ascending order.

```

public class MergeSortedArrays {
    public static void main(String[] args) {
        int[] arrayA = {1, 3, 5, 7, 9};
        int[] arrayB = {2, 4, 6, 8, 10};
        int[] mergedArray = mergeSortedArrays(arrayA, arrayB);
        System.out.println("Merged Array:");
        displayArray(mergedArray);
    }
    private static int[] mergeSortedArrays(int[] arrA, int[] arrB) {
        int lengthA = arrA.length;
        int lengthB = arrB.length;
        int[] mergedArray = new int[lengthA + lengthB];

        int i = 0, j = 0, k = 0;
        while (i < lengthA && j < lengthB) {
            if (arrA[i] <= arrB[j]) {
                mergedArray[k++] = arrA[i++];
            } else {
|

```

```

        mergedArray[k++] = arrB[j++];
    }
}
, if any
while (i < lengthA) {
    mergedArray[k++] = arrA[i++];
}

if any
while (j < lengthB) {
    mergedArray[k++] = arrB[j++];
}

return mergedArray;
}

private static void displayArray(int[] arr) {
    for (int num : arr) {
        System.out.print(num + " ");
    }
    System.out.println();
}
}

```

Output

```

java -cp /tmp/G0q8iGhWYW MergeSortedArrays
Merged Array:
1 2 3 4 5 6 7 8 9 10
|
```

24. Write a Java program to show 0-arguments constructor.

```

public class ZeroArgumentsConstructorDemo {
    public static void main(String[] args) {
        MyClass myObject = new MyClass();
        System.out.println("Object created using 0-arguments constructor.");
    }
}
class MyClass {
    public MyClass() {
        System.out.println("Inside 0-arguments constructor.");
    }
}
```

Output

```

java -cp /tmp/G0q8iGhWYW ZeroArgumentsConstructorDemo
Inside 0-arguments constructor.
Object created using 0-arguments constructor.
```

26. Write a Java program to show parameterized constructor.

```

public class ParameterizedConstructorDemo {
    public static void main(String[] args) {
        MyClass myObject = new MyClass("Hello, Parameterized
                                         Constructor!"); myObject.displayMessage();
    }
}
class MyClass {
    private String message;
    public MyClass(String msg) {
        // Initializing the instance variable with the provided
        parameter message = msg;
```

```

    }
    public void displayMessage() {
        System.out.println("Message from the object: " + message);
    }
}

```

Output

```
java -cp /tmp/G0q8iGhWYw ParameterizedConstructorDemo
Message from the object: Hello, Parameterized Constructor!
```

27. Write a Java program to show constructor overloading.

```

public class ConstructorOverloadingDemo {
    public static void main(String[] args) {
        MyClass obj1 = new MyClass();
        MyClass obj2 = new MyClass("Hello, Constructor Overloading!");
        MyClass obj3 = new MyClass(42);
        obj1.displayMessage();
        obj2.displayMessage();
        obj3.displayMessage();
    }
}

class MyClass {
    private String message;
    private int number;

    public MyClass() {
        message = "Default Constructor Message";
        number = 0;
    }

    public MyClass(String msg) {
        message = msg;
        number = 0;
    }

    public MyClass(int num) {
        message = "Parameterized Constructor with Number";
        number = num;
    }

    public void displayMessage() {
        System.out.println("Message: " + message);
        System.out.println("Number: " + number);
        System.out.println();
    }
}

```

Output

```
java -cp /tmp/G0q8iGhWYw ConstructorOverloadingDemo
Message: Default Constructor Message
Number: 0

Message: Hello, Constructor Overloading!
Number: 0

Message: Parameterized Constructor with Number
Number: 42
```

28. Write a class, Grader, which has an instance variable, score, an appropriate constructor and appropriate methods. A method, letterGrade() that returns the letter grade as O/E/A/B/C/F. Now write a demo class to test the Grader class by reading a score from the user, using it to create a Grader object after validating that the value is not negative and is not greater than 100. Finally, call the letterGrade() method to

get and print the grade.

```
import java.util.Scanner;
class Grader {
    private int score;
    public Grader(int score) {
        this.score = score;
    }
    public String letterGrade() {
        if (score >= 90 && score <= 100) {
            return "O";
        } else if (score >= 80 && score < 90)
            { return "E";
        } else if (score >= 70 && score < 80)
            { return "A";
        } else if (score >= 60 && score < 70)
            { return "B";
        } else if (score >= 50 && score < 60)
            { return "C";
        } else {
            return "F";
        }
    }
}

public class GraderDemo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the score: ");
        int userScore = scanner.nextInt();
        if (userScore >= 0 && userScore <= 100) {
            Grader grader = new Grader(userScore);
            String grade = grader.letterGrade();
            System.out.println("Letter Grade: " + grade);
        } else {
            System.out.println("Invalid score. Score must be between 0 and 100.");
        }
        scanner.close();
    }
}
```

Output

```
java -cp /tmp/oRSC47w6CK graderDemo
Enter the score: 64
Letter Grade: B
|
```

28. Write a class, **Commission**, which has an instance variable, sales; an appropriate constructor; and a method, commission() that returns the commission. Now write a demo class to test the Commission class by reading a sale from the user, using it to create a Commission object after validating that the value is not negative. Finally, call the commission() method to get and print the commission. If the sales are negative, your demo should print the message “Invalid Input”.

```
import java.util.Scanner;
class Commission {
    private double sales;
    public Commission(double sales) {
        this.sales = sales;
    }
    public double calculateCommission() {
        if (sales < 0) {
            System.out.println("Invalid Input");
            return -1; // You can choose to return a specific value for invalid input
        } else if (sales <= 1000)
            { return 0.1 * sales;
        }
    }
}
```

```

        } else if (sales <= 5000)
            { return 0.15 * sales;
        } else {
            return 0.2 * sales;
        }
    }

public class CommissionDemo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            System.out.print("Enter the sales amount: ");
            double sales = scanner.nextDouble();
            Commission commissionObj = new Commission(sales);
            double result = commissionObj.calculateCommission();
            if (result != -1) {
                System.out.printf("Commission: $%.2f%n", result);
            }
        } catch (java.util.InputMismatchException e) { System.out.println("Invalid
            Input. Please enter a valid numeric value.");
        } finally {
            scanner.close();
        }
    }
}

```

| |
|--|
| Output |
| <pre> java -cp /tmp/oRSC47w6CK CommissionDemo Enter the sales amount: 486 Commission: \$48.60 </pre> |

ASSIGNMENT – 4

1. Write a Java program to implement the concept of inheritance

```

import java.util.*;
class Employee {
    float salary;
    public void setSalary(float salary) {
        this.salary = salary;
    }
    public float getSalary() {
        return salary;
    }
}
class Programmer extends Employee {
    int bonus;
    public void setBonus(int bonus) {
        this.bonus = bonus;
    }
    public int getBonus() {
        return bonus;
    }
}
public class q1_inheritance {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Programmer p = new Programmer();
        System.out.print("Enter salary of the programmer: ");
        float salary = scanner.nextFloat();
        p.setSalary(salary);
        System.out.print("Enter bonus of the programmer: ");
        int bonus = scanner.nextInt();
        p.setBonus(bonus);
    }
}

```

```
System.out.println("Programmer salary is: " + p.getSalary());
System.out.println("Bonus of Programmer is: " + p.getBonus());
scanner.close(); }}
```

Output

```
java -cp /tmp/hdbuwew2OP Inheritance
Enter salary of the programmer: 45000
Enter bonus of the programmer: 10000
Programmer salary is: 45000.0
Bonus of Programmer is: 10000
```

2. Write a Java program to show method overloading.

```
import java.util.*;
class Adder {
    static int add(int a, int b)
    { return a + b; }
    static int add(int a, int b, int c) {
        return a + b + c; }
}
public class q2_method_overload { public static void
main(String[] args) { Scanner scanner = new
Scanner(System.in); System.out.println("Enter two
numbers for addition:"); int num1 = scanner.nextInt();

int num2 = scanner.nextInt();
System.out.println("Result of addition using 2 nos" + Adder.add(num1, num2));
System.out.println("Enter three numbers for addition:");
int num3 = scanner.nextInt();
int num4 = scanner.nextInt();
int num5 = scanner.nextInt();
System.out.println("Result of addition using 3 nos: " + Adder.add(num3, num4, num5));
scanner.close(); }}
```

Output

```
java -cp /tmp/bydgyuhd7 Overloading
Enter Two Numbers for Addition:
12 45
Result of Addition Using 2 Numbers: 57
Enter Three Numbers for Addition:
12 47 89
Result of Addition Using 3 Numbers: 148
```

3. Write a Java program to show method overriding.

```
import java.util.*;
class Vehicle {
    void run() {
        System.out.println("Vehicle is running");}
}
public class q3_method_overriding extends Vehicle {
    public static void main(String args[]) {
        q3_method_overriding obj = new q3_method_overriding();
        obj.run();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a message to override 'run' method: ");
        String message = scanner.nextLine();
        obj.run(message);
        scanner.close(); }}
```

```
void run(String message) {
    System.out.println("Overridden run method: " + message); }
```

Output

```
java -cp /tmp/fferesrst9V Overriding
Enter a message to override 'run' method: stop
Overridden run method: stop
```

4. Write a Java program to show method hiding.

```
import java.util.*;
class Complex {
    public static void f1() {
        System.out.println("f1 method of the Complex class is executed."); }
    class Sample extends Complex {
        public static void f1() {
            System.out.println("f1 of the Sample class is executed."); }
        public class q4_method_hiding {
            public static void main(String args[]) {
                Scanner scanner = new Scanner(System.in);
                System.out.println("Enter 1 to execute f1 from Complex class, 2 to execute f1 from Sample class:");
                int choice = scanner.nextInt();
                if (choice == 1) {
                    Complex.f1();
                } else if (choice == 2)
                    { Sample.f1(); }
                else {
                    System.out.println("Invalid choice!"); }
                scanner.close(); }
```

Output

```
java -cp /tmp/fejwbhwbo Hiding
1. Enter 1 to Execute f1 from Complex Class
2. Enter 2 to Execute f1 from Sample Class:
2
f1 of the Sample class is executed.
```

5. Create a general class ThreeDObject and derive the classes Box, Cube, Cylinder and Cone from it.

The class **ThreeDObject** has methods **wholeSurfaceArea()** and **volume()**. Override these two methods in each of the derived classes to calculate the volume and whole surface area of each type of three-dimensional objects. The dimensions of the objects are to be taken from the users and passed through the respective constructors of each derived class. Write a main method to test these classes.

```
import java.util.*;
class ThreeDObject {
    public double wholeSurfaceArea() {
        return 0.0; }
    public double volume() {
        return 0.0; }
    class Box extends ThreeDObject {
        private double length;
        private double width;
        private double height;
        public Box(double length, double width, double height) {
            this.length = length;
            this.width = width;
            this.height = height; }
        public double wholeSurfaceArea() {
```

```
return 2 * (length * width + length * height + width * height);
}
public double volume() {
    return length * width * height;
}
}
class Cube extends Box {
    public Cube(double side) {
        super(side, side, side);
    }
}
class Cylinder extends ThreeDObject {
    protected double radius;
    protected double height;
    public Cylinder(double radius, double height) {
        this.radius = radius;
        this.height = height;
    }
    public double wholeSurfaceArea() {
        return 2 * Math.PI * radius * (radius + height);
    }
    public double volume() {
        return Math.PI * radius * radius * height;
    }
}
class Cone extends Cylinder {
    public Cone(double radius, double height) {
        super(radius, height);
    }
    public double wholeSurfaceArea() {
        return Math.PI * radius * (radius + Math.sqrt(radius * radius + height * height));
    }
    public double volume() {
        return (1.0 / 3) * Math.PI * radius * radius * height;
    }
}
public class q5_sa_volume {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter dimensions for Box (length, width, height):");
        double boxLength = scanner.nextDouble();
        double boxWidth = scanner.nextDouble();
        double boxHeight = scanner.nextDouble();
        Box box = new Box(boxLength, boxWidth, boxHeight);
        System.out.println("Box Whole Surface Area: " + box.wholeSurfaceArea());
        System.out.println("Box Volume: " + box.volume());
        System.out.println("\nEnter side length for Cube:");
        double cubeSide = scanner.nextDouble();
        Cube cube = new Cube(cubeSide);
        System.out.println("Cube Whole Surface Area: " + cube.wholeSurfaceArea());
        System.out.println("Cube Volume: " + cube.volume());
        System.out.println("\nEnter dimensions for Cylinder (radius, height):");
        double cylinderRadius = scanner.nextDouble();
        double cylinderHeight = scanner.nextDouble();
        Cylinder cylinder = new Cylinder(cylinderRadius, cylinderHeight);
        System.out.println("Cylinder Whole Surface Area: " + cylinder.wholeSurfaceArea());
        System.out.println("Cylinder Volume: " + cylinder.volume());
        System.out.println("\nEnter dimensions for Cone (radius, height):");
        double coneRadius = scanner.nextDouble();
        double coneHeight = scanner.nextDouble();
        Cone cone = new Cone(coneRadius, coneHeight);
        System.out.println("Cone Whole Surface Area: " + cone.wholeSurfaceArea());
```

```

System.out.println("Cone Volume: " + cone.volume());
scanner.close();
}
}

Output
java -cp /tmp/uifhfwihru5 ThreedObject
Enter dimensions for Box (length, width, height):
12 5 18
Box Whole Surface Area: 732.0
Box Volume: 1080.0

```

6. Write a program to create a class named Vehicle having protected instance variables regnNumber, speed, color, ownerName and a method showData () to show “This is a vehicle class”. Inherit the Vehicle class into subclasses named Bus and Car having individual private instance variables routeNumber in Bus and manufacturerName in Car and both of them having showData () method showing all details of Bus and Car respectively with content of the super class’s showData () method.

```

import java.util.*;
class Vehicle {
protected String regnNumber;
protected int speed;
protected String color;
protected String ownerName;
public Vehicle(String regnNumber, int speed, String color, String ownerName) {
this.regnNumber = regnNumber;
this.speed = speed;
this.color = color;
this.ownerName = ownerName;
}
public void showData() {
System.out.println("This is a vehicle class");
}
}
class Bus extends Vehicle {
private String routeNumber;
public Bus(String regnNumber, int speed, String color, String ownerName, String routeNumber)
{ super(regnNumber, speed, color, ownerName);
this.routeNumber = routeNumber;
}
public void showData() {
super.showData();
System.out.println("Route Number: " + routeNumber);
System.out.println("Registration Number: " + regnNumber);
System.out.println("Speed: " + speed);
System.out.println("Color: " + color);
System.out.println("Owner Name: " + ownerName); }
}
class Car extends Vehicle {
private String manufacturerName;
public Car(String regnNumber, int speed, String color, String ownerName, String manufacturerName) {
super(regnNumber, speed, color, ownerName);
this.manufacturerName = manufacturerName; }
public void showData() {
super.showData();
System.out.println("Manufacturer Name: " + manufacturerName);
System.out.println("Registration Number: " + regnNumber);
System.out.println("Speed: " + speed);
System.out.println("Color: " + color);
System.out.println("Owner Name: " + ownerName); } }

public class q6_vehicle {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Enter details for Bus:");

```

```

System.out.print("Registration Number: ");
String busRegnNumber = scanner.nextLine();
System.out.print("Speed: ");
int busSpeed = scanner.nextInt();
scanner.nextLine();
System.out.print("Color: ");
String busColor = scanner.nextLine();
System.out.print("Owner Name: ");
String busOwnerName = scanner.nextLine();
System.out.print("Route Number: ");
String busRouteNumber = scanner.nextLine();
Bus myBus = new Bus(busRegnNumber, busSpeed, busColor, busOwnerName, busRouteNumber);
System.out.println("Details of Bus:");
myBus.showData();
scanner.close();
}
    
```

Output

```

java -cp /tmp/gdgetfees4 Vehicle
Enter details for Bus:
Registration Number: 1234
Speed: 300
Color: red
Owner Name: rahul
Route Number: 4
Details of Bus:
This is a vehicle class
Route Number: 4
Registration Number: 1234
Speed: 300
Color: red
Owner Name: rahul
    
```

7. An educational institution maintains a database of its employees. The database is divided into a number of classes whose hierarchical relationships are shown below. Write all the classes and define the methods to create the database and retrieve individual information as and when needed. Write a driver program to test the classes. Staff (code, name) Teacher (subject, publication) is a Staff Officer (grade) is a Staff Typist (speed) is a Staff RegularTypist (remuneration) is a Typist CasualTypist (daily wages) is a Typist.

```

import java.util.*;
class Staff {
protected String code;
protected String name;
public Staff(String code, String name) {
this.code = code;
this.name = name; }
public String toString() {
return "Code: " + code + ", Name: " + name; }
class Teacher extends Staff {
    
```

```
private String subject;
private String publication;
public Teacher(String code, String name, String subject, String publication)
{ super(code, name);
this.subject = subject;
this.publication = publication;}
public String toString()
{return super.toString() + ", Subject: " + subject + ", Publication: " + publication; }
class Officer extends Staff {
private String grade;
public Officer(String code, String name, String grade) {
super(code, name);
this.grade = grade; }
public String toString()
{return super.toString() + ", Grade: " + grade; }
class Typist extends Staff {
private int speed;
public Typist(String code, String name, int speed) {
super(code, name);
this.speed = speed; }
public String toString()
{return super.toString() + ", Speed: " + speed; }
class RegularTypist extends Typist {
private double remuneration;
public RegularTypist(String code, String name, int speed, double remuneration) {
super(code, name, speed);
this.remuneration = remuneration; }
public String toString()
{return super.toString() + ", Remuneration: " + remuneration; }
class CasualTypist extends Typist {
private double dailyWages;
public CasualTypist(String code, String name, int speed, double dailyWages) {
super(code, name, speed);
this.dailyWages = dailyWages; }
public String toString()
{return super.toString() + ", Daily Wages: " + dailyWages; }
public class q7_database_employees {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Enter details for Teacher:");
System.out.print("Code: ");
String teacherCode = scanner.nextLine();
System.out.print("Name: ");
String teacherName = scanner.nextLine();
System.out.print("Subject: ");
String subject = scanner.nextLine();
System.out.print("Publication: ");
String publication = scanner.nextLine();
Teacher teacher = new Teacher(teacherCode, teacherName, subject,
publication); System.out.println("Enter details for Officer:");
System.out.print("Code: ");
String officerCode = scanner.nextLine();
System.out.print("Name: ");
String officerName = scanner.nextLine();
System.out.print("Grade: ");
String grade = scanner.nextLine();
Officer officer = new Officer(officerCode, officerName, grade);
System.out.println("Enter details for Regular Typist:");
System.out.print("Code: ");
String regularTypistCode = scanner.nextLine();
System.out.print("Name: ");
String regularTypistName = scanner.nextLine();
```

```

System.out.print("Speed: ");
int regularTypistSpeed = scanner.nextInt();
System.out.print("Remuneration: ");
double remuneration = scanner.nextDouble();
RegularTypist regularTypist = new RegularTypist(regularTypistCode, regularTypistName,
regularTypistSpeed, remuneration);
System.out.println("Enter details for Casual Typist:");
System.out.print("Code: ");
String casualTypistCode = scanner.next();
System.out.print("Name: ");
String casualTypistName = scanner.next();
System.out.print("Speed: ");
int casualTypistSpeed = scanner.nextInt();
System.out.print("Daily Wages: ");
double dailyWages = scanner.nextDouble();
CasualTypist casualTypist = new CasualTypist(casualTypistCode, casualTypistName, casualTypistSpeed, dailyWages);
System.out.println("Teacher: " + teacher);
System.out.println("Officer: " + officer);
System.out.println("Regular Typist: " + regularTypist);
System.out.println("Casual Typist: " + casualTypist);
scanner.close(); }}
```

Output

```

java -cp /tmp/rterer4were1 Teacher
Teacher: Code: 12, Name: anjali, Subject: maths, Publication: chaya
Officer: Code: 255, Name: parul, Grade: A
Regular Typist: Code: nan, Name: renox, Speed: 150, Remuneration: 5000.0
Casual Typist: Code: 342, Name: anjali, Speed: 300, Daily Wages: 400.0

```

8. Create a base class **Building** that stores the number of floors of a building, number of rooms and it's total footage. Create a derived class **House** that inherits **Building** and also stores the number of bedrooms and bathrooms. Demonstrate the working of the classes.

```

import java.util.*;
class Building {
    protected int numFloors;
    protected int numRooms;
    protected double totalFootage;
    public Building(int numFloors, int numRooms, double totalFootage) {
        this.numFloors = numFloors;
        this.numRooms = numRooms;
        this.totalFootage = totalFootage; }
    public String toString() {
        return "Number of Floors: " + numFloors + ", Number of Rooms: " + numRooms + ", Total Footage: " + totalFootage; }}
```

```

class House extends Building {
    private int numBedrooms;
    private int numBathrooms;
    public House(int numFloors, int numRooms, double totalFootage, int numBedrooms, int numBathrooms) {
        super(numFloors, numRooms, totalFootage);
        this.numBedrooms = numBedrooms;
        this.numBathrooms = numBathrooms; }
    public String toString() {
        return super.toString() + ", Number of Bedrooms: " + numBedrooms + ", Number of Bathrooms: " + numBathrooms; }}
```

```

public class q8_building {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter details for Building:");
        System.out.print("Number of Floors: ");
        int buildingFloors = scanner.nextInt();
        System.out.print("Number of Rooms: ");
        int buildingRooms = scanner.nextInt();
        System.out.print("Total Footage: ");
        double buildingFootage = scanner.nextDouble();
```

```

Building building = new Building(buildingFloors, buildingRooms, buildingFootage);
System.out.println("Enter details for House:");
System.out.print("Number of Floors: ");
int houseFloors = scanner.nextInt();
System.out.print("Number of Rooms: ");
int houseRooms = scanner.nextInt();
System.out.print("Total Footage: ");
double houseFootage = scanner.nextDouble();
System.out.print("Number of Bedrooms: ");
int houseBedrooms = scanner.nextInt();
System.out.print("Number of Bathrooms: ");
int houseBathrooms = scanner.nextInt();
House house = new House(houseFloors, houseRooms, houseFootage, houseBedrooms, houseBathrooms);
System.out.println("\nBuilding Details:");
System.out.println(building);
System.out.println("\nHouse Details:");
System.out.println(house);
scanner.close();
}}

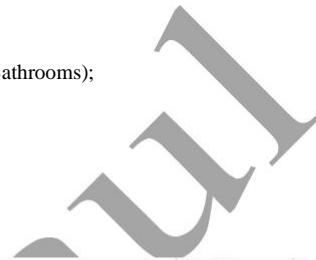
```

Output

```

java -cp /tmp/dtygq3fgs2 BuildingDetails
Building Details:
Number of Floors: 12, Number of Rooms: 45, Total Footage: 1200.0
House Details:
Number of Floors: 14, Number of Rooms: 8, Total Footage: 455.0, Number of Bedrooms: 3,
Number of Bathrooms: 1

```

**Clear**

- 9. In the earlier program, create a second derived class Office that inherits Building and stores the number of telephones and tables. Now demonstrate the working of all three classes.**

```

import java.util.*;
class Building {
    protected int numFloors;
    protected int numRooms;
    protected double totalFootage;
    public Building(int numFloors, int numRooms, double totalFootage) {
        this.numFloors = numFloors;
        this.numRooms = numRooms;
        this.totalFootage = totalFootage; }
    public String toString() {
        return "Number of Floors: " + numFloors + ", Number of Rooms: " + numRooms + ", Total Footage: " + totalFootage; }
}
class House extends Building {
    private int numBedrooms;
    private int numBathrooms;
    public House(int numFloors, int numRooms, double totalFootage, int numBedrooms, int numBathrooms) {
        super(numFloors, numRooms, totalFootage);
        this.numBedrooms = numBedrooms;
        this.numBathrooms = numBathrooms; }
    public String toString() {
        return super.toString() + ", Number of Bedrooms: " + numBedrooms + ", Number of Bathrooms: " + numBathrooms; }
}
class Office extends Building {
    private int numTelephones;
    private int numTables;
    public Office(int numFloors, int numRooms, double totalFootage, int numTelephones, int numTables) {
        super(numFloors, numRooms, totalFootage);
        this.numTelephones = numTelephones;
        this.numTables = numTables; }
    public String toString() {
        return super.toString() + ", Number of Telephones: " + numTelephones + ", Number of Tables: " +
        numTables; } }
public class q9_office {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter details for Building:");

```

```

System.out.print("Number of Floors: ");
int buildingFloors = scanner.nextInt();
System.out.print("Number of Rooms: ");
int buildingRooms = scanner.nextInt();
System.out.print("Total Footage: ");
double buildingFootage = scanner.nextDouble();
Building building = new Building(buildingFloors, buildingRooms, buildingFootage);
System.out.println("Enter details for House:"); System.out.print("Number of Floors: ");
");
int houseFloors = scanner.nextInt();
System.out.print("Number of Rooms: ");
int houseRooms = scanner.nextInt();
System.out.print("Total Footage: ");
double houseFootage = scanner.nextDouble();
System.out.print("Number of Bedrooms: ");
int houseBedrooms = scanner.nextInt();
System.out.print("Number of Bathrooms: ");
int houseBathrooms = scanner.nextInt();
House house = new House(houseFloors, houseRooms, houseFootage, houseBedrooms, houseBathrooms);
System.out.println("Enter details for Office:"); System.out.print("Number of Floors: ");
int officeFloors = scanner.nextInt();
System.out.print("Number of Rooms: ");
int officeRooms = scanner.nextInt();
System.out.print("Total Footage: ");
double officeFootage = scanner.nextDouble();
System.out.print("Number of Telephones: ");
int officeTelephones = scanner.nextInt();
System.out.print("Number of Tables: ");
int officeTables = scanner.nextInt();
Office office = new Office(officeFloors, officeRooms, officeFootage, officeTelephones, officeTables);
System.out.println("\nBuilding Details:");
System.out.println(building);
System.out.println("\nHouse Details:");
System.out.println(house);
System.out.println("\nOffice Details:");
System.out.println(office);
scanner.close();}}

```

Clear

```

java -cp /tmp/ygrywger0 BuildingDetails
Building Details:
Number of Floors: 2, Number of Rooms: 14, Total Footage: 1200.0
House Details:
Number of Floors: 2, Number of Rooms: 25, Total Footage: 522.0, Number of Bedrooms: 10
, Number of Bathrooms: 2
Office Details:
Number of Floors: 5, Number of Rooms: 48, Total Footage: 1400.0, Number of Telephones:
700, Number of Tables: 500

```

- 10. Write a Java program which creates a base class Num and contains an integer number along with a method shownum() which displays the number. Now create a derived class HexNum which inherits Num and overrides shownum() which displays the hexadecimal value of the number. Demonstrate the working of the classes.**

```

import java.util.*;
class Num {
protected int number;
public Num(int number) {
this.number = number;
}
public void showNum() {
System.out.println("Number: " + number); }
}
class HexNum extends Num {
public HexNum(int number) {
}
}

```

```

super(number); }
public void showNum() {
System.out.println("Hexadecimal Value: " + Integer.toHexString(number));
}} public class q10_Hex_Num {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Enter a number:");
int num = scanner.nextInt();
Num regularNum = new Num(num);
HexNum hexNum = new HexNum(num);
System.out.println("Using base class Num:");
regularNum.showNum();
System.out.println("\nUsing derived class HexNum:");
hexNum.showNum();
scanner.close(); }}
```

Output

```

java -cp /tmp/bcayugrbw6 Num
Enter a number:
145
Using base class Num:
Number: 145
Using derived class HexNum:
Hexadecimal Value: 91
```

11. Write a Java program which creates a base class Num and contains an integer number along with a method showNum() which displays the number. Now create a derived class OctNum which inherits Num and overrides showNum() which displays the octal value of the number. Demonstrate the working of the classes.

```

import java.util.*;
class Num {
protected int number;
public Num(int number) {
this.number = number; }
public void showNum() {
System.out.println("Number: " + number); }}
class OctNum extends Num {
public OctNum(int number) {
super(number); }
public void showNum() {
System.out.println("Octal Value: " + Integer.toOctalString(number)); }}
public class q11_Oct_Num {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Enter a number:");
int num = scanner.nextInt();
Num regularNum = new Num(num);
OctNum octNum = new OctNum(num);
System.out.println("Using base class Num:");
regularNum.showNum();
System.out.println("\nUsing derived class OctNum:");
octNum.showNum();
scanner.close(); }}
```

Output

```

java -cp /tmp/dyggqwjyw1 Num
Enter a number:
835
Using base class Num:
Number: 835
Using derived class OctNum:
Octal Value: 1503
```

- 12.** Create a base class Distance which stores the distance between two locations in miles and a method travelTime(). The method prints the time taken to cover the distance when the speed is 60 miles per hour. Now in a derived class DistanceMKS, override travelTime() so that it prints the time assuming the distance is in kilometers and the speed is 100 km per second. Demonstrate the working of the classes.

```
import java.util.*;
class Distance {
    protected double distanceInMiles;
    public Distance(double distanceInMiles) {
        this.distanceInMiles = distanceInMiles;
    }
    public void travelTime() {
        double time = distanceInMiles / 60.0;
        System.out.println("Time taken to cover the distance: " + time + " hours");
    }
} class DistanceMKS extends Distance {
    public DistanceMKS(double distanceInMiles) {
        super(distanceInMiles);
    }
    public void travelTime() {
        double distanceInKilometers = distanceInMiles * 1.60934;
        double time = distanceInKilometers / 100.0;
        System.out.println("Time taken to cover the distance: " + time + " hours");
    }
}
public class q13_distance {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter distance in miles:");
        double distanceInMiles = scanner.nextDouble();
        Distance distance1 = new Distance(distanceInMiles);
        DistanceMKS distance2 = new DistanceMKS(distanceInMiles);
        System.out.println("For Distance:");
        distance1.travelTime();
        System.out.println("\nFor DistanceMKS:");
        distance2.travelTime();
        scanner.close();
    }
}
```

| Output |
|--|
| <pre>java -cp /tmp/bfyrdgey5 distance Enter distance in miles: 45 For Distance: Time taken to cover the distance: 0.75 hours For DistanceMKS: Time taken to cover the distance: 0.7242029999999999 hours</pre> |

- 13.** Create a base class called “vehicle” that stores number of wheels and speed. Create the following derived classes – “car” that inherits “vehicle” and also stores number of passengers. “truck” that inherits “vehicle” and also stores the load limit. Write a main function to create objects of these two derived classes and display all the information about “car” and “truck”. Also compare the speed of these two vehicles - car and truck and display which one is faster.

```
import java.util.*;
class Vehicle {
    protected int numberOfWheels;
    protected double speed;
    public Vehicle(int numberOfWheels, double speed) {
        this.numberOfWheels = numberOfWheels;
        this.speed = speed;
    }
    public void displayInfo() {
        System.out.println("Number of Wheels: " + numberOfWheels);
        System.out.println("Speed: " + speed + " mph");
    }
}
class Car extends Vehicle {
    private int numberOfPassengers;
    public Car(int numberOfWheels, double speed, int numberOfPassengers) {
        super(numberOfWheels, speed);
        this.numberOfPassengers = numberOfPassengers;
    }
    public void displayInfo() {
```

```

super.displayInfo();
System.out.println("Number of Passengers: " + numberOfPassengers);}}
```

```

class Truck extends Vehicle {
private double loadLimit;
public Truck(int numberOfWheels, double speed, double loadLimit) {
super(numberOfWheels, speed);
this.loadLimit = loadLimit;}}
```

```

public void displayInfo() {
super.displayInfo();
System.out.println("Load Limit: " + loadLimit + " tons");}}
```

```

public class q14_vehicle_car_truck {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Enter information for Car:");
System.out.print("Number of Wheels: ");
int carWheels = scanner.nextInt();
System.out.print("Speed (mph): ");
double carSpeed = scanner.nextDouble();
System.out.print("Number of Passengers: ");
int carPassengers = scanner.nextInt();
Car car = new Car(carWheels, carSpeed, carPassengers);
System.out.println("\nEnter information for Truck:");
System.out.print("Number of Wheels: ");
int truckWheels = scanner.nextInt();
System.out.print("Speed (mph): ");
double truckSpeed = scanner.nextDouble();
System.out.print("Load Limit (tons): ");
double truckLoadLimit = scanner.nextDouble();
Truck truck = new Truck(truckWheels, truckSpeed, truckLoadLimit);
System.out.println("\nCar Information:");
car.displayInfo();
System.out.println("\nTruck Information:");
truck.displayInfo();
System.out.println("\nComparison:");
if (car.speed > truck.speed) {
System.out.println("The car is faster than the truck.");
} else if (car.speed < truck.speed) {
System.out.println("The truck is faster than the
car."); } else {
System.out.println("Both car and truck have the same speed.");
}
scanner.close();}}
```

Output

```

java -cp /tmp/byetywgr4 carInformation
Car Information:
Number of Wheels: 4
Speed: 300.0 mph
Number of Passengers: 4
Truck Information:
Number of Wheels: 6
Speed: 120.0 mph
Load Limit: 500.0 tons
Comparison:
The car is faster than the truck.
```

14. Write a Java program to explain “multilevel inheritance.”

```

import java.util.*;
class Animal {
public void eat() {
System.out.println("Animal is eating...");}
class Dog extends Animal {
public void bark() {
System.out.println("Dog is barking...");}
class Labrador extends Dog {
public void color() {
```

```

System.out.println("Labrador is brown in color."); }
public class q15_multilevel_inheritnce {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Which animal do you want to create?");
System.out.println("1. Animal");
System.out.println("2. Dog");
System.out.println("3. Labrador");
System.out.print("Enter your choice: ");
int choice = scanner.nextInt();
switch (choice) {
case 1:
Animal animal = new Animal();
animal.eat();
break;
case 2:
Dog dog = new Dog();
dog.eat();
dog.bark();
break;
case 3:
Labrador labrador = new Labrador();
labrador.eat();
labrador.bark();
labrador.color();
break;
default:
System.out.println("Invalid choice!");
scanner.close();}}
```

Output

```

java -cp /tmp/zbyuebyr2 multilevel
Which animal do you want to create?
1. Animal
2. Dog
3. Labrador
Enter your choice: 2
Animal is eating... Dog is barking...
```

ASSIGNMENT – 5

1. Create a “circle” class & a “point” class. The coordinates of the circle are given and used within the “circle” class as object of the “point” class. Display the area of circle.

```

import java.util.*;
class Point {
double x, y;
Point(double x, double y) { this.x = x;
this.y = y; }} class Circle {
Point centre; int r;
Circle(Point centre, int r) { this.centre = centre; this.r = r; }
void area() {
System.out.println("Area of Circle: " + Math.PI * r * r); }} class q1_area_of_circle
{ static Scanner sc = new Scanner(System.in); public static void main(String args[])
{
System.out.print("Enter X Co-ordinates: "); double x = sc.nextDouble(); System.out.print("Enter Y Co-ordinates: "); double y = sc.nextDouble();
System.out.print("Enter Radius: ");
int r = sc.nextInt();
```

```
Point centre = new Point(x, y); Circle cir = new Circle(centre, r); cir.area(): }}
```

Output

```
java -cp /bydegye04 Circle
Enter X Co-ordinates: 12
Enter Y Co-ordinates: 32
Enter Radius: 45
Area of Circle: 6361.725123519332
```

2. Create a class called Time, which has three private instance variables – hour, min and sec. It contains a method called add() which takes one Time object as parameter and print the added value of the calling Time object and passes Time object. In the main method, declare two Time objects and assign values using constructor and call the add() method.

```
public class Time {
    private int hour;
    private int min;
    private int sec;
    public Time(int hour, int min, int sec) {
        this.hour = hour;
        this.min = min;
        this.sec = sec;
    }
    public void add(Time t) {
        int totalSec1 = this.hour * 3600 + this.min * 60 + this.sec;
        int totalSec2 = t.hour * 3600 + t.min * 60 + t.sec;
        int total = totalSec1 + totalSec2;

        int addedHour = total / 3600;
        int addedMin = (total % 3600) / 60;
        int addedSec = total % 60;
        System.out.println("Added Time: " + addedHour + " hours " + addedMin + " minutes " + addedSec + " seconds");
    }
    public static void main(String[] args) {
        Time time1 = new Time(5, 30, 45);
        Time time2 = new Time(2, 15, 20);
        System.out.println("Time 1: " + time1.hour + " hours " + time1.min + " minutes " + time1.sec + " seconds");
        System.out.println("Time 2: " + time2.hour + " hours " + time2.min + " minutes " + time2.sec + " seconds");
        time1.add(time2);
    }
}
```

Output

```
java -cp /tmp/tNEE3bjq1j Time
Time 1: 5 hours 30 minutes 45 seconds
Time 2: 2 hours 15 minutes 20 seconds
Added Time: 7 hours 46 minutes 5 seconds
```

3. Create a class called Complex, which has three private instance variables –real and imaginary. It contains a method called add() which takes one Complex object as parameter and print the added value of the calling Complex object and passes Complex object. In the main method, declare two Complex objects and assign values using constructor and call the add() method.

```
public class Complex {
    private double real;
    private double imaginary;
```

```

public Complex(double real, double imaginary) {
    this.real = real;
    this.imaginary = imaginary;
}
public void add(Complex other) {
    double sumReal = this.real + other.real;
    double sumImaginary = this.imaginary + other.imaginary;
    System.out.println("Sum: " + sumReal + " + " + sumImaginary + "i");
}
public static void main(String[] args) {
    Complex complex1 = new Complex(2.5, 3.5);
    Complex complex2 = new Complex(1.5, 2.5);
    System.out.println("Complex 1:");
    complex1.add(complex2);
}
}

Output
java -cp /tmp/cKwiGNu1UQ Complex
Complex 1:
Sum: 4.0 + 6.0i

```

4. Write a program to define a class having one 3-digit number, num as data member. Initialize and display reverse of that number.

```

import java.util.Scanner;
public class ThreeDigitNumber {
    private int num;
    public ThreeDigitNumber(int num) {
        if (num >= 100 && num <= 999) {
            this.num = num;
        } else {
            System.out.println("Invalid input! Please enter a 3-digit number.");
        }
    }
    public int reverseNumber() {
        int reverse = 0;
        int temp = num;
        while (temp != 0) {
            int digit = temp % 10;
            reverse = reverse * 10 + digit;
            temp /= 10;
        }
        return reverse;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a 3-digit number: ");
        int num = scanner.nextInt();
        ThreeDigitNumber number = new ThreeDigitNumber(num);
        if (number.num != 0) {
            int reverse = number.reverseNumber();
            System.out.println("Reverse of " + num + " is: " + reverse);
        }
    }
}

```

}

Output

```
java -cp /tmp/cKwiGNu1UQ ThreeDigitNumber
Enter a 3-digit number: 639
Reverse of 639 is: 936
```

5. Write a program to define a class Student with four data members such as name, roll no., sub1, and sub2. Define appropriate methods to initialize and display the values of data members. Also calculate total marks and percentage scored by student.

```
import java.util.Scanner;
public class Student {
    private String name;
    private int rollNo;
    private int sub1;
    private int sub2;
    public Student(String name, int rollNo, int sub1, int sub2) {
        this.name = name;
        this.rollNo = rollNo;
        this.sub1 = sub1;
        this.sub2 = sub2;
    }
    public void displayStudentDetails() {
        System.out.println("Student Name: " + name);
        System.out.println("Roll No.: " + rollNo);
        System.out.println("Subject 1 Marks: " + sub1);
        System.out.println("Subject 2 Marks: " + sub2);
    }
    public int calculateTotalMarks() {
        return sub1 + sub2;
    }
    public double calculatePercentage() {
        int totalMarks = calculateTotalMarks();
        return (totalMarks / 2.0);
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter Student Name: ");
        String name = scanner.nextLine();
        System.out.print("Enter Roll No.: ");
        int rollNo = scanner.nextInt();
        System.out.print("Enter Marks for Subject 1: ");
        int sub1 = scanner.nextInt();
        System.out.print("Enter Marks for Subject 2: ");
        int sub2 = scanner.nextInt();
        Student student = new Student(name, rollNo, sub1, sub2);
        System.out.println("\nStudent Details:");
        student.displayStudentDetails();
        int totalMarks = student.calculateTotalMarks();
        double percentage = student.calculatePercentage();
        System.out.println("\nTotal Marks: " + totalMarks);
        System.out.println("Percentage: " + percentage + "%");
    }
}
```

}

Output

```
java -cp /tmp/cKwiGNu1UQ Student
Enter Student Name: Anjali
Enter Roll No.: 57
Enter Marks for Subject 1: 88
Enter Marks for Subject 2: 92

Student Details:
Student Name: Anjali
Roll No.: 57
Subject 1 Marks: 88
Subject 2 Marks: 92

Total Marks: 180
Percentage: 90.0%
```

6. Write a program to define a class Employee to accept emp_id, emp_name, basic_salary from the user and display the gross_salary.

```
import java.util.Scanner;
public class Employee {
    private int empId;
    private String empName;
    private double basicSalary;
    public Employee(int empId, String empName, double basicSalary) {
        this.empId = empId;
        this.empName = empName;
        this.basicSalary = basicSalary;
    }
    public double calculateGrossSalary() {
        return basicSalary + (0.2 * basicSalary);
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter Employee ID: ");
        int empId = scanner.nextInt();
        scanner.nextLine(); // Consume newline character
        System.out.print("Enter Employee Name: ");
        String empName = scanner.nextLine();
        System.out.print("Enter Basic Salary: ");
        double basicSalary = scanner.nextDouble();

        Employee employee = new Employee(empId, empName, basicSalary);
        double grossSalary = employee.calculateGrossSalary();
        System.out.println("\nEmployee Details:");
        System.out.println("ID: " + empId);
        System.out.println("Name: " + empName);
        System.out.println("Basic Salary: " + basicSalary);
        System.out.println("Gross Salary: " + grossSalary);
    }
}
```

```

Output
java -cp /tmp/cKwiGNu1UQ Employee
Enter Employee ID: 23146
Enter Employee Name: Akash
Enter Basic Salary: 67000

Employee Details:
ID: 23146
Name: Akash
Basic Salary: 67000.0
Gross Salary: 80400.0

```

7. Write a program to define a class Fraction having data members numerator and denominator. Initialize three objects using different constructors and display its fractional value.

```

public class Fraction {
    private int numerator;
    private int denominator;
    public Fraction() {
        this(1, 1);
    }
    public Fraction(int numerator, int denominator) {
        this.numerator = numerator;
        if (denominator != 0) {
            this.denominator = denominator;
        } else {
            System.out.println("Denominator cannot be zero. Setting it to 1.");
            this.denominator = 1;
        }
    }
    (denominator defaults to 1)
    public Fraction(int numerator) {
        this(numerator, 1);
    }
    public void displayFraction() {
        System.out.println(numerator + "/" + denominator);
    }
}
public static void main(String[] args) {
    Fraction fraction1 = new Fraction();
    Fraction fraction2 = new Fraction(3, 4);
    Fraction fraction3 = new Fraction(5);
    System.out.println("Fraction 1: ");
    fraction1.displayFraction();
    System.out.println("Fraction 2: ");
    fraction2.displayFraction();
    System.out.println("Fraction 3: ");
    fraction3.displayFraction();
}
}

```

```

Output
java -cp /tmp/cKwiGNu1UQ Fraction
Fraction 1:
1/1
Fraction 2:
3/4
Fraction 3:
5/1

```

8. Write a program to define a class Item containing code and price. Accept this data for five objects using array of objects. Display code, price in tabular form and also, display total price of all items.

```
import java.util.Scanner;
public class Item {
    private String code;
    private double price;
    public Item(String code, double price) {
        this.code = code;
        this.price = price;
    }
    public String getCode() {
        return code;
    }
    public double getPrice() {
        return price;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Item[] items = new Item[5];
        for (int i = 0; i < items.length; i++) {
            System.out.println("Enter details for item " + (i + 1) + ":");
            System.out.print("Code: ");
            String code = scanner.next();
            System.out.print("Price: ");
            double price = scanner.nextDouble();
            items[i] = new Item(code, price);
        }
        System.out.println("\nCode\tPrice");
        double totalPrice = 0;
        for (Item item : items) {
            System.out.println(item.getCode() + "\t" + item.getPrice());
            totalPrice += item.getPrice();
        }
        System.out.println("\nTotal Price of All Items: " + totalPrice);
    }
}
```

```
Output
java -cp /tmp/ckwiGNu1UQ Item
Enter details for item 1:
Code: 233
Price: 1500
Enter details for item 2:
Code: 234
Price: 800
Enter details for item 3:
Code: 235
Price: 3500
Enter details for item 4:
Code: 236
Price: 500
Enter details for item 5:
Code: 237
Price: 2700

Code      Price
233  1500.0
234  800.0
235  3500.0
236  500.0
237  2700.0

Total Price of All Items: 9000.0
```

9. Write a program to define a class **Tender** containing data members **cost** and **company name**. Accept data for five objects and display company name for which cost is minimum.

```
import java.util.Scanner;
public class Tender {
    private double cost;
    private String companyName;
    public Tender(double cost, String companyName) {
        this.cost = cost;
        this.companyName = companyName;
    }
    public double getCost() {
        return cost;
    }
    public String getCompanyName() {
        return companyName;
    }
}

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    Tender[] tenders = new Tender[5];
    for (int i = 0; i < tenders.length; i++) {
        System.out.println("Enter details for tender " + (i + 1) + ":");
        System.out.print("Cost: ");
        double cost = scanner.nextDouble();
        scanner.nextLine(); // Consume newline character
        System.out.print("Company Name: ");
        String companyName = scanner.nextLine();
        tenders[i] = new Tender(cost, companyName);
    }
    String minCostCompanyName = tenders[0].getCompanyName();
    double minCost = tenders[0].getCost();
    for (int i = 1; i < tenders.length; i++) {
        if (tenders[i].getCost() < minCost) {
            minCost = tenders[i].getCost();
            minCostCompanyName = tenders[i].getCompanyName();
        }
    }
}
```

```

        }
    }
    System.out.println("\nCompany Name with Minimum Cost: " + minCostCompanyName);
}}
```

Output

```

java -cp /tmp/cKwiGNu1UQ Tender
Enter details for tender 1:
Cost: 50000
Company Name: RM Brand
Enter details for tender 2:
Cost: 33000
Company Name: DVC Furnitures
Enter details for tender 3:
Cost: 45000
Company Name: BBC Pvt Ltd
Enter details for tender 4:
Cost: 88000
Company Name: UV Brand
Enter details for tender 5:
Cost: 62000
Company Name: DIY Pvt Ltd

Company Name with Minimum Cost: DVC Furnitures

```

10. Write a program to define a class 'employee' with data members as empid, name and salary. Accept data for 5 objects using Array of objects and print it.

```

import java.util.Scanner;
public class Employee {
    private int empId;
    private String name;
    private double salary;
    public Employee(int empId, String name, double salary) {
        this.empId = empId;
        this.name = name;
        this.salary = salary;
    }
    public int getEmpId() {
        return empId;
    }
    public String getName() {
        return name;
    }
    public double getSalary() {
        return salary;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Employee[] employees = new Employee[5];
        for (int i = 0; i < employees.length; i++) {
            System.out.println("Enter details for employee " + (i + 1) + ":");
            System.out.print("Employee ID: ");
            int empId = scanner.nextInt();
            scanner.nextLine();
            System.out.print("Name: ");
            String name = scanner.nextLine();
            System.out.print("Salary: ");
            double salary = scanner.nextDouble();
            employees[i] = new Employee(empId, name, salary);
        }
        System.out.println("\nEmployee Details:");
    }
}
```

```

for (Employee employee : employees) {
    System.out.println("Employee ID: " + employee.getEmpId());
    System.out.println("Name: " + employee.getName());
    System.out.println("Salary: " + employee.getSalary());
    System.out.println();
}
}
}

```

| Output |
|---|
| <pre> java -cp /tmp/cKwiGNu1UQ Employee Enter details for employee 1: Employee ID: 23145 Name: Rahul Salary: 50000 Enter details for employee 2: Employee ID: 23146 Name: Anjali Salary: 80000 Enter details for employee 3: Employee ID: 23147 Name: Kiran Salary: 42000 Enter details for employee 4: Employee ID: 23148 Name: Rosy Salary: 67000 Enter details for employee 5: Employee ID: 23149 Name: Akash Salary: 55000 Employee Details: Employee ID: 23145 Name: Rahul Salary: 50000.0 </pre> |

| |
|---|
| <pre> Employee ID: 23146 Name: Anjali Salary: 80000.0 Employee ID: 23147 Name: Kiran Salary: 42000.0 Employee ID: 23148 Name: Rosy Salary: 67000.0 Employee ID: 23149 Name: Akash Salary: 55000.0 </pre> |
|---|

11. Define a class called circle that contains:

- Two private instance variables: radius (of type double) and color (of type String),
- Initialize the variables radius and color with default value of 1.0 and "red", respectively using default constructor.
- Include a second constructor that will use the default value for color and sets the radius to the value passed as parameter.
- Two public methods: getRadius() and getArea() for returning the radius and area of the circle
- Invoke the above methods and constructors in the main.

```

public class Circle {
    private double radius;
    private String color;
    public Circle() {
        this.radius = 1.0;
        this.color = "red";
    }
    public Circle(double radius) {
        this.radius = radius;
        this.color = "red";
    }
    public double getRadius() {
        return radius;
    }
    public double getArea() {
        return Math.PI * radius * radius;
    }
}

```

```
public static void main(String[] args) {  
    Circle circle1 = new Circle();  
    System.out.println("Circle 1 - Radius: " + circle1.getRadius() + ", Area: " +  
        circle1.getArea()); Circle circle2 = new Circle(2.5);  
    System.out.println("Circle 2 - Radius: " + circle2.getRadius() + ", Area: " + circle2.getArea());  
}  
}
```

Output

```
java -cp /tmp/6GowVQdIG9 Circle  
Circle 1 - Radius: 1.0, Area: 3.141592653589793  
Circle 2 - Radius: 2.5, Area: 19.634954084936208
```

12. Write a program which will accept an integer from the user and pass the value to a method called `PrintNumberInWord` that will print "ONE", "TWO", ..., "NINE", "ZERO" if the integer variable "number" is 1, 2, ..., 9, or 0, respectively.

```
import java.util.Scanner;  
public class NumberWordPrinter {  
    public static void main(String[] args) {  
        Scanner scanner = new Scanner(System.in);  
        System.out.print("Enter a number (0-9): ");  
        int number = scanner.nextInt();  
        printNumberInWord(number);  
    }  
    public static void printNumberInWord(int number) {  
        switch (number) {  
            case 0:  
                System.out.println("ZERO");  
                break;  
            case 1:  
                System.out.println("ONE");  
                break;  
            case 2:  
                System.out.println("TWO");  
                break;  
            case 3:  
                System.out.println("THREE");  
                break;  
            case 4:  
                System.out.println("FOUR");  
                break;  
            case 5:  
                System.out.println("FIVE");  
                break;  
            case 6:  
                System.out.println("SIX");  
                break;  
            case 7:  
                System.out.println("SEVEN");  
                break;  
            case 8:  
                System.out.println("EIGHT");  
                break;  
            case 9:  
                System.out.println("NINE");  
                break;  
        }  
    }  
}
```

```

        System.out.println("NINE");
        break;
    default:
        System.out.println("Invalid number entered");
    }
}
}

Output

java -cp /tmp/6GowVQdIG9 NumberWordPrinter
Enter a number (0-9): 6
SIX

```

13. Design a class named Account that contains:

- I. A private int data field named id for the account (default 0).
- II. A private double data field named balance for the account (default 0).
- III. A private double data field named annualInterestRate that stores the cur-rent interest rate (default 0). Assume all accounts have the same interest rate.
- IV. A private Date data field named dateCreated that stores the date when the account was created.
- V. A no-arg constructor that creates a default account.
- VI. A constructor that creates an account with the specified id and initial balance.
- VII. The accessor and mutator methods for id,balance, and annualInterestRate.
- VIII. The accessor method for dateCreated.
- IX. A method named getMonthlyInterestRate() that returns the monthly interest rate.
- X. A method named getMonthlyInterest() that returns the monthly interest.
- XI. A method named withdraw that withdraws a specified amount from the account.
- XII. A method named deposit that deposits a specified amount to the account.

```

import java.util.; import java.text.*;
@SuppressWarnings("unused") class Account {
private int id;
private double balance, annualInterestRate; private String dataCreated;
static SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy"); static Date date = new Date();
Account() { id = 0;
balance = 0.0;
annualInterestRate = 0.0;
dataCreated = formatter.format(date); }
Account(int id, double balance, double annualInterestRate) { this.id = id;
this.balance = balance; this.annualInterestRate = annualInterestRate; dataCreated =
formatter.format(date); } void getMonthlyInterestRate() {
System.out.println("Monthly Interest Rate: " + annualInterestRate / 12); } void getMonthlyInterest() {
System.out.println("Monthly Interest: " + (balance * annualInterestRate / 12 / 100)); } void withdraw(int amount)
{ balance -= amount;
System.out.println("Amount After Withdraw: " + balance); } void deposit(int amount)
{ balance += amount;
System.out.println("Amount After Deposit: " + balance); } } class q13_Interest {
static Scanner sc = new Scanner(System.in); public static void main(String args[])
{ System.out.print("Enter Id: "); int id = sc.nextInt();
System.out.print("Enter Initial Balance: "); double bal = sc.nextDouble(); System.out.print("Enter Annual Interest Rate: "); double rate
= sc.nextDouble();
Account obj = new Account(id, bal, rate); int amount;
while (true) {

```

```

System.out.println("\n1. Show Interest Rate\n2. Show Monthly Interest\n3. Withdraw\n4. Deposit"); int ch = sc.nextInt();
switch (ch) { case 1:
obj.getMonthlyInterestRate();
break; case 2:
obj.getMonthlyInterest(); break;
case 3:
System.out.println("Enter Amount: "); amount = sc.nextInt();
obj.withdraw(amount); break; case 4:
System.out.println("Enter Amount: "); amount = sc.nextInt();
obj.deposit(amount); break; } int x;
System.out.println("Edit More?\n1. Yes\n2. No\n"); x = sc.nextInt();
if (x == 2) { System.exit(0); }}}
```

Output

```

java -cp /tmp/ftyftyftv1 Account
Enter Id: 123
Enter Initial Balance: 150000
Enter Annual Interest Rate: 10
1. Show Interest Rate
2. Show Monthly Interest
3. Withdraw
4. Deposit
2
Monthly Interest: 1250.0
```

14. Write a test program that prompts the user to enter the investment amount (e.g., 1000) and the interest rate (e.g., 9%), and print a table that displays future value for the years from 1 to 30, as shown below: The amount invested: 1000 Annual interest rate: 9%

| Years | Future Value |
|-------|-----------------|
| 1 | 1093.8 |
| 2 | 1196.41 |
| ... | |
| 29 | 13467.25 |
| 30 | 14730.57 |

```

import java.util.Scanner;
public class InvestmentCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the investment amount: ");
        double investmentAmount = scanner.nextDouble();
        System.out.print("Enter the annual interest rate (in percentage): ");
        double annualInterestRate = scanner.nextDouble();
        double monthlyInterestRate = annualInterestRate / 100 / 12;
        System.out.printf("%-5s %-20s%n", "Years", "Future Value");
        for (int years = 1; years <= 30; years++) {
            double futureValue = calculateFutureValue(investmentAmount, monthlyInterestRate, years *
                12); System.out.printf("%-5d %-20.2f%n", years, futureValue);
        }
    }
    public static double calculateFutureValue(double investmentAmount, double monthlyInterestRate, int numberOfMonths)
    { return investmentAmount * Math.pow(1 + monthlyInterestRate, numberOfMonths);
    }
}
```

```
Output
java -cp /tmp/TECBzS37ji InvestmentCalculator
Enter the investment amount: 50000
Enter the annual interest rate (in percentage): 7.3
Years Future Value
1      53774.63
2      57834.22
3      62200.28
4      66895.95
5      71946.10
6      77377.51
7      83218.94
8      89501.36
9      96258.06
10     103524.84
11     111340.20
12     119745.57
13     128785.49
14     138507.85
15     148964.17
16     160209.88
17     172304.55
18     185312.28
19     199302.00
20     214347.84
21     230529.53
22     247932.82
```

15. Write method headers for the following methods:

- a. Computing a sales commission, given the sales amount and the commission rate.
- b. Printing the calendar for a month, given the month and year,
- c. Computing a square root.
- d. Testing whether a number is even, and returning true if it is.
- e. Printing a message a specified number of times.
- f. Computing the monthly payment, given the loan amount, number of years, and annual interest rate.

```
import java.util.Scanner;
public class MethodHeaders {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter method name:");
        String methodName = scanner.nextLine();
        switch (methodName) {
            case "computeSalesCommission":
                computeSalesCommission(1000.0, 0.05);
                break;
            case "printCalendar":
                printCalendar(2, 2024);
                break;
            case "computeSquareRoot":
                computeSquareRoot(25.0);
                break;
            case "isEven":
                isEven(6);
                break;
            case "printMessage":
                printMessage("Hello World", 3);
                break;
            case "computeMonthlyPayment":
                computeMonthlyPayment(50000.0, 5, 0.06);
                break;
            default:
                System.out.println("Invalid method name");
        }
    }
}
```

```

        }
    }

    public static void computeSalesCommission(double salesAmount, double commissionRate)
    {
        double commission = salesAmount * commissionRate;
        System.out.println("Sales Commission: " + commission);
    }

    public static void printCalendar(int month, int year) {
        System.out.println("Calendar for " + month + "/" + year + ":");
    }

    public static void computeSquareRoot(double number) {
        double squareRoot = Math.sqrt(number);
        System.out.println("Square Root: " + squareRoot);
    }

    public static void isEven(int number) {
        boolean isEven = number % 2 == 0;
        System.out.println(number + " is even: " + isEven);
    }

    public static void printMessage(String message, int times) {
        for (int i = 0; i < times; i++) {
            System.out.println(message);
        }
    }

    public static void computeMonthlyPayment(double loanAmount, int numberOfYears, double annualInterestRate)
    {
        double monthlyInterestRate = annualInterestRate / 12;
        int numberOfMonths = numberOfYears * 12;
        double monthlyPayment = (loanAmount * monthlyInterestRate) /
            (1 - Math.pow(1 + monthlyInterestRate, -numberOfMonths));
        System.out.println("Monthly Payment: " + monthlyPayment);
    }
}

```

Output

```

java -cp /tmp/TECBzS37ji MethodHeaders
Enter method name:
computeSquareRoot
Square Root: 5.0

```

16. Write a program that reads ten numbers, computes their average, and finds out how many numbers are above the average. [Use this keyword]

```

import java.util.Scanner;
public class AverageAndAbove {
    private double[] numbers;
    private double average;
    public AverageAndAbove() {
        this.numbers = new double[10];
        this.average = 0.0;
    }
    public void readNumbers() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter ten numbers:");
        for (int i = 0; i < 10; i++) {
            System.out.print("Enter number " + (i + 1) + ": ");
            this.numbers[i] = scanner.nextDouble();
            this.average += this.numbers[i];
        }
    }
}

```

```
        }
        this.average /= 10;
    }
    public void displayAboveAverage() {
        int countAbove = 0;
        for (double num : this.numbers) {
            if (num > this.average) {
                countAbove++;
            }
        }
        System.out.println("Average: " + this.average);
        System.out.println("Numbers above the average: " + countAbove);
    }
    public static void main(String[] args) {
        AverageAndAbove obj = new AverageAndAbove();
        obj.readNumbers();
        obj.displayAboveAverage();
    }
}
```

Output

```
java -cp /tmp/vNUq5ExTwd AverageAndAbove
Enter ten numbers:
Enter number 1: 9
Enter number 2: 50
Enter number 3: 36
Enter number 4: 78
Enter number 5: 1
Enter number 6: 45
Enter number 7: 67
Enter number 8: 28
Enter number 9: 18
Enter number 10: 33
Average: 36.5
Numbers above the average: 4
```

17. Write a program that reads ten integers and displays them in the reverse of the order in which they were read.

```
import java.util.Scanner;
public class ReverseOrder {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int[] numbers = new int[10];
        System.out.println("Enter ten integers:");
        for (int i = 0; i < 10; i++) {
            System.out.print("Enter integer " + (i + 1) + ": ");
            numbers[i] = scanner.nextInt();
        }
        System.out.println("Integers in reverse order:");
        for (int i = numbers.length - 1; i >= 0; i--) {
            System.out.println(numbers[i]);
        }
    }
}
```

```

Output
java -cp /tmp/vNUq5ExTwd ReverseOrder
Enter ten integers:
Enter integer 1: 50
Enter integer 2: 12
Enter integer 3: 78
Enter integer 4: 30
Enter integer 5: 3
Enter integer 6: 96
Enter integer 7: 36
Enter integer 8: 44
Enter integer 9: 28
Enter integer 10: 18
Integers in reverse order:
18
28
44
36
96
3
30
78
12
50

```

18. Write a program to demonstrate use of 'this' keyword.

```

import java.util.*;
class example {
    int data; example(int data) {
        this.data = data; } void print() {
            System.out.println("The Data is: " + data); } class q18_example_of_this
{ static Scanner sc = new Scanner(System.in); public static void
main(String args[]) {
    System.out.print("Enter Data: "); int n = sc.nextInt();
    example obj = new example(n); obj.print();}}

```

```

Output
java -cp /gdsyrgeyb6 this
Enter Data: 123
The Data is: 123

```

19. Write a program to demonstrate use of 'static' keyword.

```

import java.util.*;
class q19_example_of_static {
    static Scanner sc = new Scanner(System.in); static int count = 0;
    static void counter() { count++; }
    public static void main(String args[]) {
        System.out.println("The Counter Now Shows 0: " + count); counter();
        System.out.println("The Counter Now Shows 1: " + count); }}

```

```

Output
java -cp /rvgevtyw2 static
The Counter Now Shows 0:0
The Counter Now Shows 1: 1

```

20. Write a program to accept value of apple sales for each day of the week (using array of type float) and then, calculate the average sale of the week.

```

import java.util.Scanner;
public class WeeklySalesAverage {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

```

```

float[] sales = new float[7];
System.out.println("Enter the sales for each day of the week:");
for (int i = 0; i < 7; i++) {
    System.out.print("Enter sales for day " + (i+1) + ": ");
    sales[i] = scanner.nextFloat();
}
float totalSales = 0;
for (float sale : sales) {
    totalSales += sale;
}
float averageSale = totalSales / 7;
System.out.println("Average sale for the week: " + averageSale);
}
}

```

Output

```

java -cp /tmp/vNUq5ExTwD WeeklySalesAverage
Enter the sales for each day of the week:
Enter sales for day 1: 120
Enter sales for day 2: 70
Enter sales for day 3: 360
Enter sales for day 4: 190
Enter sales for day 5: 90
Enter sales for day 6: 200
Enter sales for day 7: 220
Average sale for the week: 178.57143

```

21. Write program, which finds the sum of numbers formed by consecutive digits. Input : 2415 output : 24+41+15=80.

```

import java.util.Scanner;
public class ConsecutiveDigitSum {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Input: ");
        String input = scanner.nextLine();
        int sum = 0;
        for (int i = 0; i < input.length() - 1; i++) {
            int num = Integer.parseInt(input.substring(i, i + 1)) +
                Integer.parseInt(input.substring(i + 1, i + 2));
            sum += num;
            System.out.print(input.substring(i, i + 1) + input.substring(i + 1, i + 2));
            if (i < input.length() - 2) {
                System.out.print("+");
            }
        }
        System.out.println("=" + sum);
    }
}

```

Output

```

java -cp /tmp/vNUq5ExTwD ConsecutiveDigitSum
Input: 2415
24+41+15=17

```

Week: 6

\} Design an abstract class having two methods. Create Rectangle and Triangle classes by inheriting the shape class and override the above methods to suitably implement for Rectangle and Triangle class.

```
Code: abstract class Shape {
    public abstract double calculateArea();
    public abstract double
    calculatePerimeter();
}

class Rectangle extends Shape {
    private double length;
    private double width;
    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }
    public double calculateArea() {
        return length * width;
    }
    public double calculatePerimeter()
    { return 2 * (length + width);
    }
}

class Triangle extends Shape {
    private double side1;
    private double side2;
    private double side3;
    public Triangle(double side1, double side2, double side3)
    { this.side1 = side1;
        this.side2 = side2;
        this.side3 = side3;
    }
    public double calculateArea() { double
        s = (side1 + side2 + side3) / 2;
        return Math.sqrt(s * (s - side1) * (s - side2) * (s - side3));
    }
    public double calculatePerimeter()
    { return side1 + side2 + side3;
    }
}

public class ShapeTest {
    public static void main(String[] args) { Rectangle rectangle = new Rectangle(5,
        4); Triangle triangle = new Triangle(3, 4, 5); System.out.println("Rectangle
        Area: " + rectangle.calculateArea()); System.out.println("Rectangle
        Perimeter: " + rectangle.calculatePerimeter()); System.out.println("Triangle
        Area: " + triangle.calculateArea());
    }
}
```

```

        System.out.println("Triangle Perimeter: " + triangle.calculatePerimeter());
    }
}

Rectangle Area: 20.0
Rectangle Perimeter: 18.0
Triangle Area: 6.0
Triangle Perimeter: 12.0

```

14. Write a program in Java to illustrate the use of interface in Java.

Code:

```

interface Printable {
    void print();
}

class Printer implements Printable {

    public void print() {
        System.out.println("Printing...");
    }
}

public class InterfaceExample {
    public static void main(String[] args) {
        Printable printer = new Printer();
        printer.print();
    }
}

```

= Create a general class ThreeDObject and derive the classes Box, Cube, Cylinder and Cone from it. The class ThreeDObject has methods wholeSurfaceArea() and volume(). Override these two methods in each of the derived classes to calculate the volume and whole surface area of each type of three-dimensional objects. The dimensions of

the objects are to be taken from the users and passed through the respective constructors of each derived class. Write a main method to test these classes.

Code:

```

abstract class ThreeDObject {
    public abstract double wholeSurfaceArea();
    public abstract double volume();
}

class Box extends ThreeDObject {
    private double length;
    private double width;
    private double height;

    public Box(double length, double width, double height) {
        this.length = length;
        this.width = width;
        this.height = height;
    }

    public double wholeSurfaceArea() {
        return 2 * (length * width + length * height + width * height);
    }
}

```

```
public double volume() {
    return length * width * height;
}
}
class Cube extends ThreeDObject {
    private double side;
    public Cube(double side) {
        this.side = side;
    }
    public double wholeSurfaceArea() {
        return 6 * side * side;
    }
    public double volume() {
        return side * side * side;
    }
}
class Cylinder extends ThreeDObject {
    private double radius;
    private double height;
    public Cylinder(double radius, double height) {
        this.radius = radius;
        this.height = height;
    }
    public double wholeSurfaceArea() {
        return 2 * Math.PI * radius * (radius + height);
    }
    public double volume() {
        return Math.PI * radius * radius * height;
    }
}
class Cone extends ThreeDObject {
    private double radius;
    private double height;
    public Cone(double radius, double height) {
        this.radius = radius;
        this.height = height;
    }
    public double wholeSurfaceArea() {
        double slantHeight = Math.sqrt(radius * radius + height * height); return Math.PI * radius * (radius + slantHeight);
    }
    public double volume() {
        return (1.0 / 3.0) * Math.PI * radius * radius * height;
    }
}
public class ThreeDObjectTest {
    public static void main(String[] args) {
        Box box = new Box(2, 3, 4);
        System.out.println("Box Whole Surface Area: " + box.wholeSurfaceArea());
```

```

System.out.println("Box Volume: " + box.volume());
Cube cube = new Cube(5);
System.out.println("Cube Whole Surface Area: " + cube.wholeSurfaceArea());
System.out.println("Cube Volume: " + cube.volume());
Cylinder cylinder = new Cylinder(3, 6);
System.out.println("Cylinder Whole Surface Area: " + cylinder.wholeSurfaceArea());
System.out.println("Cylinder Volume: " + cylinder.volume());
Cone cone = new Cone(4, 7);
System.out.println("Cone Whole Surface Area: " + cone.wholeSurfaceArea());
System.out.println("Cone Volume: " + cone.volume());
}

```

```

Box Whole Surface Area: 52.0
Box Volume: 24.0
Cube Whole Surface Area: 150.0
Cube Volume: 125.0
Cylinder Whole Surface Area: 188.4955592153876
Cylinder Volume: 169.64600329384882
Cone Whole Surface Area: 175.92918860102842
Cone Volume: 117.6470588235294

```

2. Write a program to create a class named Vehicle having protected instance variables regnNumber, speed, color, ownerName and a method showData () to show “This is a vehicle class”. Inherit the Vehicle class into subclasses named Bus and Car having individual private instance variables routeNumber in Bus and manufacturerName in Car and both of them having showData () method showing all details of Bus and Car respectively with content of the super class’s showData () method.

Code: class Vehicle { protected

```

String regnNumber;
protected double speed;
protected String color;
protected String ownerName;
public Vehicle(String regnNumber, double speed, String color, String ownerName) {
    this.regnNumber = regnNumber;
    this.speed = speed; this.color =
    color; this.ownerName =
    ownerName;
}
protected void showData() {
    System.out.println("This is a vehicle class");
}
}
```

```

class Bus extends Vehicle {
    private int routeNumber;
    public Bus(String regnNumber, double speed, String color, String ownerName, int routeNumber)
    { super(regnNumber, speed, color, ownerName);
        this.routeNumber = routeNumber;
    }
}
```

```

protected void showData() {
    super.showData();
    System.out.println("Route Number: " + routeNumber);
}
}

class Car extends Vehicle {
    private String manufacturerName;
    public Car(String regnNumber, double speed, String color, String ownerName, String
manufacturerName) {
        super(regnNumber, speed, color, ownerName);
        this.manufacturerName = manufacturerName;
    }
    protected void showData() {
        super.showData();
        System.out.println("Manufacturer Name: " + manufacturerName);
    }
}
}

public class VehicleTest {
    public static void main(String[] args) {
        Bus bus = new Bus("ABC123", 60.0, "Red", "John Doe",
        101); bus.showData();
        Car car = new Car("XYZ456", 100.0, "Blue", "Jane Doe",
        "Toyota"); car.showData();
    }
}
}

```

```

This is a vehicle class
Route Number: 101
This is a vehicle class
Manufacturer Name: Toyota

```

3. Create three interfaces, each with two methods. Inherit a new interface from the three, adding a new method. Create a class by implementing the new interface and also inheriting from a concrete class. Now write four methods, each of which takes one of the four interfaces as an argument. In main (), create an object of your class and pass it to each of the methods.

Code: interface Interface1

```

{ void method1();
  void method2();
}

interface Interface2
{ void method3();
  void method4();
}

interface Interface3 extends Interface1, Interface2 {
  void method5();
}

class MyClass implements Interface3 {

```

```

public void method1() {
    System.out.println("Method 1");
}
public void method2() {
    System.out.println("Method 2");
}
public void method3() {
    System.out.println("Method 3");
}
public void method4() {
    System.out.println("Method 4");
}
public void method5() {
    System.out.println("Method 5");
}
}

public class InterfaceInheritanceTest {
    public static void main(String[] args) {
        MyClass myObj = new MyClass();
        myObj.method1();
        myObj.method2();
        myObj.method3();
        myObj.method4();
        myObj.method5();
    }
}

```

Method 1
 Method 2
 Method 3
 Method 4
 Method 5

4. Create an interface Department containing attributes deptName and deptHead. It also has abstract methods for printing the attributes. Create a class hostel containing hostelName, hostelLocation and numberofRooms. The class contains methods for getting and printing the attributes. Then write Student class extending the Hostel class and implementing the Department interface. This class contains attributes studentName, regdNo, electiveSubject and avgMarks. Write suitable getData and printData methods for this class. Also implement the abstract methods of the Department interface. Write a driver class to test the Student class. The program should be menu driven containing the options:

- i) Admit new student
- ii) Migrate a student
- iii) Display details of a student

For the third option a search is to be made on the basis of the entered registration number.

Code: interface Department {

```
void printDepartment();
}
class Hostel {
    protected String hostelName;
    protected String hostelLocation;
    protected int numberOfRooms;
    public Hostel(String hostelName, String hostelLocation, int numberOfRooms)
        { this.hostelName = hostelName;
        this.hostelLocation = hostelLocation;
        this.numberOfRooms = numberOfRooms;
    }
    public void printHostel() {
        System.out.println("Hostel Name: " + hostelName);
        System.out.println("Hostel Location: " + hostelLocation);
        System.out.println("Number of Rooms: " + numberOfRooms);
    }
}
class Student extends Hostel implements Department
{
    private String studentName;
    private int regdNo;
    private String electiveSubject;
    private double avgMarks;
    public Student(String hostelName, String hostelLocation, int numberOfRooms, String
studentName, int regdNo, String electiveSubject, double avgMarks) {
        super(hostelName, hostelLocation, numberOfRooms);
        this.studentName = studentName; this.regdNo =
        regdNo;
        this.electiveSubject = electiveSubject;
        this.avgMarks = avgMarks;
    }
    public void printDepartment() {
        System.out.println("Student Department Information:");
        System.out.println("Student Name: " + studentName);
        System.out.println("Registration Number: " + regdNo);
        System.out.println("Elective Subject: " +
electiveSubject); System.out.println("Average Marks: " +
avgMarks);
    }
}
public class ClassHierarchyTest {
    public static void main(String[] args) {
        Student student = new Student("ABC Hostel", "XYZ Location", 100, "John Doe",
12345, "Mathematics", 85.5);
        student.printHostel();
        student.printDepartment();
    }
}
```

```

Hostel Name: ABC Hostel
Hostel Location: XYZ Location
Number of Rooms: 100
Student Department Information:
Student Name: John Doe
Registration Number: 12345
Elective Subject: Mathematics
Average Marks: 85.5

```

5. Create an interface called Player. The interface has an abstract method called play() that displays a message describing the meaning of “play” to the class. Create classes called Child, Musician, and Actor that all implement Player. Create an application that demonstrates the use of the classes(UsePlayer.java)

```

Code: interface Player {
    void play();
}

class Child implements Player {
    public void play() {
        System.out.println("Child is playing with toys.");
    }
}

class Musician implements Player {
    public void play() {
        System.out.println("Musician is playing an instrument.");
    }
}

class Actor implements Player {
    public void play() {
        System.out.println("Actor is performing in a play.");
    }
}

public class PlayerTest {
    public static void main(String[] args) {
        Player child = new Child(); Player
        musician = new Musician(); Player
        actor = new Actor(); child.play();

        musician.play();
        actor.play();
    }
}

```

```

Child is playing with toys.
Musician is playing an instrument.
Actor is performing in a play.

```

6. Create an abstract class Accounts with the following details:

Data Members:

7. Balance (b) accountNumber (c) accountHoldersName (d)

address Methods:

8. withdrawl()- abstract

9. deposit()- abstract

10. display() to show the balance of the account number

Create a subclass of this class SavingsAccount and add the following details:

Data Members:

9. rateOfInterest

t Methods:

calculateAount()

Code: abstract class Accounts

```

    {
        protected double balance;
        protected int accountNumber;
        protected String accountHoldersName;
        protected String address;

        public abstract void withdrawl();
        public abstract void deposit();
        public void display() {
            System.out.println("Balance of Account Number " + accountNumber + ":" + balance);
        }
    }
}

class SavingsAccount extends Accounts {
    private double rateOfInterest;
    public SavingsAccount(double balance, int accountNumber, String accountHoldersName,
String address, double rateOfInterest) {
        this.balance = balance;
        this.accountNumber = accountNumber;
        this.accountHoldersName = accountHoldersName;
        this.address = address;
        this.rateOfInterest = rateOfInterest;
    }
    public void calculateAmount() {
        balance += balance * (rateOfInterest / 100);
    }
}

public class AccountsTest {
    public static void main(String[] args) {
        SavingsAccount savingsAccount = new SavingsAccount(1000, 12345, "John Doe", "123 Main
St", 5.0);
        savingsAccount.deposit();
        savingsAccount.withdrawl();
        savingsAccount.calculateAmount();
        savingsAccount.display();
    }
}

```

```
}
```

```
Balance of Account Number 12345: 1050.0
```

10. Create an abstract class MotorVehicle with the following details: Data Members:

(a) modelName (b)modelNumber (c) modelPrice

Methods:

(a) display() to show all the details

Create a subclass of this class Carthat inherits the class MotorVehicle and add the following details:

Data Members:

(b)

discountRate

Methods:

(a) display() method to display the Car name, model number, price and the discount rate.

discount() method to compute the discount.

Code: abstract class MotorVehicle {

```
protected String modelName;  
protected int modelNumber;  
protected double modelPrice;  
public void display() {  
    System.out.println("Model Name: " + modelName);  
    System.out.println("Model Number: " + modelNumber);  
    System.out.println("Model Price: " + modelPrice);  
}
```

```
}
```

```
class Car extends MotorVehicle {
```

```
private double discountRate;  
public Car(String modelName, int modelNumber, double modelPrice, double discountRate)  
{ this.modelName = modelName;  
this.modelNumber = modelNumber;  
this.modelPrice = modelPrice;  
this.discountRate = discountRate;
```

```
}
```

```
public void display() {  
super.display();  
System.out.println("Discount Rate: " + discountRate);  
}
```

```
}
```

```
public class MotorVehicleTest {
```

```
public static void main(String[] args) {  
Car car = new Car("Toyota", 123, 25000, 10.0);  
car.display();  
}
```

```
}
```

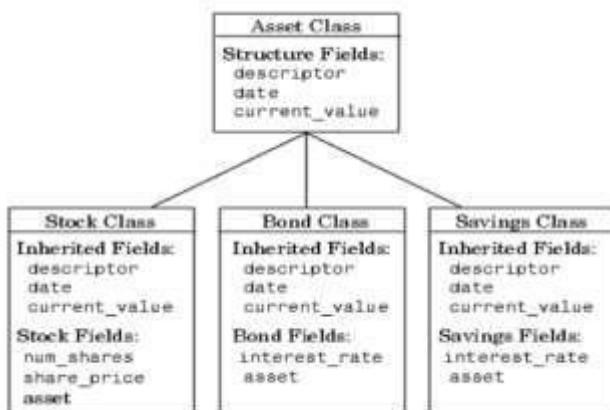
```

Model Name: Toyota
Model Number: 123
Model Price: 25000.0
Discount Rate: 10.0

```

10. Implement the below Diagram.

Here, Asset class is an abstract class containing an abstract method displayDetails() method. Stock, bond and Savings class inherit the Asset class and displayDetails() method is defined in every class.



```

Code: abstract class Asset {
    private String descriptor;
    private String date;
    private double currentValue;
    public Asset(String descriptor, String date, double currentValue) {
        this.descriptor = descriptor;
        this.date = date;
        this.currentValue = currentValue;
    }
    public abstract void displayDetails();
}
class Stock extends Asset {
    private int numShares;
    private double sharePrice;
    public Stock(String descriptor, String date, double currentValue, int numShares, double sharePrice) {
        super(descriptor, date, currentValue);
        this.numShares = numShares;
        this.sharePrice = sharePrice;
    }
    public void displayDetails() {
        System.out.println("Asset: Stock");
        System.out.println("Descriptor: " + descriptor);
        System.out.println("Date: " + date);
        System.out.println("Current Value: $" + currentValue);
        System.out.println("Number of Shares: " + numShares);
    }
}

```

```
        System.out.println("Share Price: $" + sharePrice);
    }
}
class Bond extends Asset {
    private String interestRate;
    private double asset;
    public Bond(String descriptor, String date, double currentValue, String interestRate, double asset) {
        super(descriptor, date, currentValue);
        this.interestRate = interestRate;
        this.asset = asset;
    }
    public void displayDetails() {
        System.out.println("Asset: Bond");
        System.out.println("Descriptor: " + descriptor);
        System.out.println("Date: " + date);
        System.out.println("Current Value: $" + currentValue);
        System.out.println("Interest Rate: " + interestRate);
        System.out.println("Asset Value: $" + asset);
    }
}
class Savings extends Asset {
    private String interestRate;
    public Savings(String descriptor, String date, double currentValue, String interestRate) {
        super(descriptor, date, currentValue);
        this.interestRate = interestRate;
    }
    public void displayDetails() {
        System.out.println("Asset: Savings");
        System.out.println("Descriptor: " + descriptor);
        System.out.println("Date: " + date);
        System.out.println("Current Value: $" + currentValue);
        System.out.println("Interest Rate: " + interestRate);
    }
}
```

```

Stock Details:
Asset: Stock
Descriptor: Apple Inc.
Date: 2023-03-15
Current Value: $12575.0
Number of Shares: 100
Share Price: $125.75

```

```

Bond Details:
Asset: Bond
Descriptor: US Treasury Bond
Date: 2022-09-01
Current Value: $10000.0
Interest Rate: 3.5%
Asset Value: $9850.0

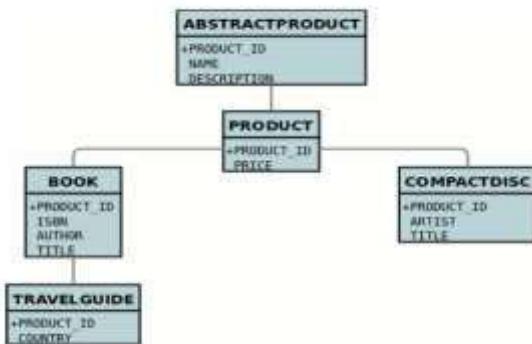
```

```

Savings Details:
Asset: Savings
Descriptor: Savings Account
Date: 2021-06-30
Current Value: $25000.0
Interest Rate: 2.75%

```

11. Implement the below Diagram. Here AbstractProduct is only abstract class.



```

Code: abstract class AbstractProduct {
    private int productId;
    private String description;
    public AbstractProduct(int productId, String description)
    { this.productId = productId;
      this.description = description;
    }
    public int getProductId() {
        return productId;
    }
    public String getDescription() {
        return description;
    }
    public abstract void showDetails();
}
class Product extends AbstractProduct {

```

```
private double price;
public Product(int productId, String description, double price) {
    super(productId, description);
    this.price = price;
}
public double getPrice() {
    return price;
}
public void showDetails() {
    System.out.println("Product ID: " + getProductId());
    System.out.println("Description: " + getDescription());
    System.out.println("Price: $" + price);
}
}
class Book extends Product {
    private String author;
    private String title;
    public Book(int productId, String description, double price, String author, String title) {
        super(productId, description, price);
        this.author = author;
        this.title = title;
    }
    public void showDetails() {
        super.showDetails();
        System.out.println("Author: " + author);
        System.out.println("Title: " + title);
    }
}
class TravelGuide extends Book {
    private String location;
    public TravelGuide(int productId, String description, double price, String author, String title,
String location) {
        super(productId, description, price, author, title);
        this.location = location;
    }
    public void showDetails() {
        super.showDetails();
        System.out.println("Location: " + location);
    }
}
class CompactDisc extends Product {
    private String artist;
    public CompactDisc(int productId, String description, double price, String artist) {
        super(productId, description, price);
        this.artist = artist;
    }
    public void showDetails() {
        super.showDetails();
    }
}
```

```

        System.out.println("Artist: " + artist);
    }
}

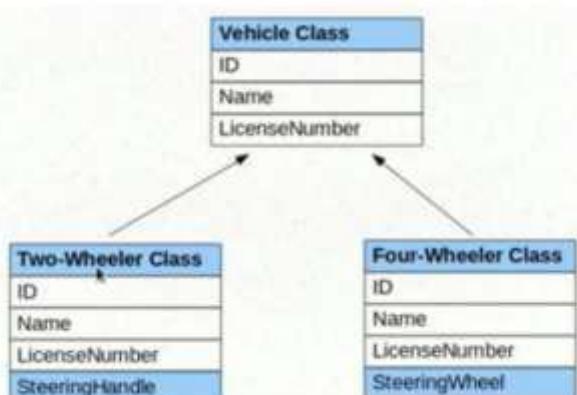
Book Details:
Product ID: 1
Description: Fiction Novel
Price: $14.99
Author: John Doe
Title: The Lost Kingdom

Travel Guide Details:
Product ID: 2
Description: Travel Guide
Price: $19.99
Author: Jane Smith
Title: Explore Italy
Location: Rome

Compact Disc Details:
Product ID: 3
Description: Pop Album
Price: $9.99
Artist: The Melodies

```

12. Implement the below Diagram



```

Code: class Vehicle {
    private int id;
    private String name;
    private String licenseNumber;
    public Vehicle(int id, String name, String licenseNumber)
        { this.id = id;
        this.name = name;
        this.licenseNumber = licenseNumber;
    }
}
class TwoWheeler extends Vehicle {
    private String steeringHandle;
    public TwoWheeler(int id, String name, String licenseNumber, String steeringHandle)
        { super(id, name, licenseNumber);
        this.steeringHandle = steeringHandle;
    }
}

```

```
    }
}

class FourWheeler extends Vehicle {
    private String steeringWheel;
    public FourWheeler(int id, String name, String licenseNumber, String steeringWheel)
        { super(id, name, licenseNumber);
        this.steeringWheel = steeringWheel;
    }
}

Two-Wheeler:
ID: 1
Name: Bike
License Number: ABC123
Steering Handle: HandleBar

Four-Wheeler:
ID: 2
Name: Car
License Number: XYZ456
Steering Wheel: Steering Wheel
```

11. Write a program to implement the Multiple Inheritance (Bank Interface, Customer & Account classes).

```
Code: interface Bank {
    void deposit(double amount);
    void withdraw(double amount);
}

class Customer implements Bank
    { private double balance;
    public Customer(double balance)
        { this.balance = balance;
    }
    public void deposit(double amount) { balance
        += amount; System.out.println("Deposited:
        $" + amount);
    }
    public void withdraw(double amount)
        { if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: $" + amount);
        } else {
            System.out.println("Insufficient balance");
        }
    }
    public void displayBalance() {
        System.out.println("Current Balance: $" +
        balance);
    }
}
```

```

class Account implements Bank {
    private double balance;
    public Account(double balance) {
        this.balance = balance;
    }
    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited: $" + amount);
    }
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: $" + amount);
        } else {
            System.out.println("Insufficient balance");
        }
    }
    public void displayBalance() {
        System.out.println("Current Balance: $" +
            balance);
    }
}
public class MultipleInheritanceBank {
    public static void main(String[] args) {
        Customer customer = new Customer(1000);
        customer.deposit(500);
        customer.withdraw(200);
        customer.displayBalance();
        Account account = new Account(2000);
        account.deposit(1000);
        account.withdraw(500);
        account.displayBalance();
    }
}

```

```

Deposited: $500.0
Withdrawn: $200.0
Current Balance: $1300.0
Deposited: $1000.0
Withdrawn: $500.0
Current Balance: $2500.0

```

12. Write a program to implement the Multiple Inheritance (Gross Interface, Employee & Salary classes).

```

Code: interface Gross {
    double calculateGrossSalary(double basicSalary, double allowances);
}
class Employee implements Gross {
    public double calculateGrossSalary(double basicSalary, double allowances) {

```

```

        return basicSalary + allowances;
    }
}
class Salary implements Gross {
    public double calculateGrossSalary(double basicSalary, double allowances)
        { return basicSalary + allowances;
    }
}
public class MultipleInheritanceGross {
    public static void main(String[] args) {
        Employee employee = new Employee();
        double empGross = employee.calculateGrossSalary(50000, 10000);
        System.out.println("Employee Gross Salary: $" + empGross); Salary
        salary = new Salary();
        double salGross = salary.calculateGrossSalary(60000, 12000);
        System.out.println("Salary Gross Salary: $" + salGross);
    }
}

```

```

Employee Gross Salary: $60000.0
Salary Gross Salary: $72000.0

```

13. Program to create a interface 'Mango' and implement it in classes 'Winter' and 'Summer'. Code:

```

interface Mango {
    void displaySeason();
}

class Winter implements Mango {
    public void displaySeason() {
        System.out.println("Winter mangoes are available from November to February.");
    }
}

class Summer implements Mango {
    public void displaySeason() {
        System.out.println("Summer mangoes are available from March to June.");
    }
}

public class MangoSeasons {
    public static void main(String[] args) {
        Winter winterMango = new Winter();
        System.out.print("Winter Mangoes: ");
        winterMango.displaySeason(); Summer
        summerMango = new Summer();
        System.out.print("Summer Mangoes: ");
        summerMango.displaySeason();
    }
}

```

Winter Mangoes: Winter mangoes are available from November to February.
Summer Mangoes: Summer mangoes are available from March to June.

14. Program to implement the Multiple Inheritance (Exam Interface, Student & Result classes).

Code: interface Exam

```
{ void
    displayExam();
}

class Student implements Exam {
    private String name; private
    int rollNumber;
    public Student(String name, int rollNumber)
        { this.name = name;
        this.rollNumber = rollNumber;
    }
    public void displayExam() {
        System.out.println("Student Name: " + name);
        System.out.println("Roll Number: " + rollNumber);
    }
}

class Result implements Exam {
    private int marks;
    public Result(int marks) {
        this.marks = marks;
    }
    public void displayExam() {
        System.out.println("Marks Obtained: " +
        marks);
    }
}

public class MultipleInheritanceExam {
    public static void main(String[] args) {
        Student student = new Student("John", 101);
        Result result = new Result(85);
        System.out.println("Student Details:");
        student.displayExam();
        System.out.println("Exam Result:");
        result.displayExam();
    }
}
```

```
Student Details:
Student Name: John
Roll Number: 101
Exam Result:
Marks Obtained: 85
```

15. Program to demonstrate use of hierarchical inheritance using interface. Code: interface Shape {

```
        double calculateArea();
    }
interface TwoDimensionalShape extends Shape {
    double calculatePerimeter();
}
interface ThreeDimensionalShape extends Shape {
    double calculateVolume();
}
class Circle implements TwoDimensionalShape {
    private double radius;
    public Circle(double radius) {
        this.radius = radius;
    }
    public double calculateArea() {
        return Math.PI * radius * radius;
    }
    public double calculatePerimeter() {
        return 2 * Math.PI * radius;
    }
}
class Sphere implements ThreeDimensionalShape {
    private double radius;
    public Sphere(double radius) {
        this.radius = radius;
    }
    public double calculateArea() {
        return 4 * Math.PI * radius * radius;
    }
    public double calculateVolume() {
        return (4.0 / 3.0) * Math.PI * radius * radius * radius;
    }
}
public class HierarchicalInheritanceDemo {
    public static void main(String[] args) {
        Circle circle = new Circle(5);
        System.out.println("Circle Area: " + circle.calculateArea());
        System.out.println("Circle Perimeter: " +
            circle.calculatePerimeter()); Sphere sphere = new Sphere(4);
        System.out.println("Sphere Area: " + sphere.calculateArea());
        System.out.println("Sphere Volume: " + sphere.calculateVolume());
    }
}
```

```
Circle Area: 78.53981633974483
Circle Perimeter: 31.41592653589793
Sphere Area: 201.06192982974676
Sphere Volume: 268.082573106329
```

16. Java program to Perform Payroll Using Interface (Multiple Inheritance).

```
Code: interface Payable {
    double calculatePay();
}

class Employee implements Payable {
    private String name;
    private double hourlyRate;
    private int hoursWorked;
    public Employee(String name, double hourlyRate, int hoursWorked)
        { this.name = name;
        this.hourlyRate = hourlyRate;
        this.hoursWorked = hoursWorked;
    }
    public double calculatePay() { return
        hourlyRate * hoursWorked;
    }
    public String getName() {
        return name;
    }
}

class Contractor implements Payable {
    private String name;
    private double rate;
    private int hoursWorked;
    public Contractor(String name, double rate, int hoursWorked)
        { this.name = name;
        this.rate = rate; this.hoursWorked
        = hoursWorked;
    }
    public double calculatePay() {
        return rate * hoursWorked;
    }
    public String getName() {
        return name;
    }
}

public class Payroll {
    public static void main(String[] args) {
        Employee employee = new Employee("John", 25.0, 40);
        Contractor contractor = new Contractor("Jane", 30.0,
        30); displayPay(employee);
        displayPay(contractor);
    }
    public static void displayPay(Payable payable) {
        System.out.println("Name: " + payable.getName());
        System.out.println("Pay: $" +
        payable.calculatePay()); System.out.println();
    }
}
```

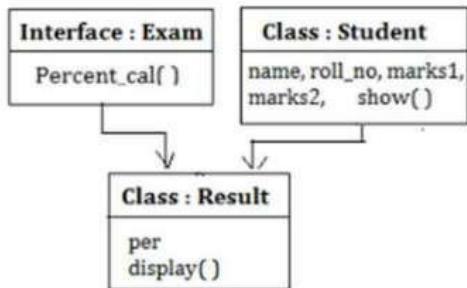
```

}
Name: John
Pay: $1000.0

Name: Jane
Pay: $900.0

```

19. Implement the following diagram.



```

Code: interface Exam {
    double percent_calc();
}

class Student implements Exam {
    private String name;
    private int roll_no;
    private int marks1, marks2;
    public Student(String name, int roll_no, int marks1, int marks2) {
        this.name = name;
        this.roll_no = roll_no;
        this.marks1 = marks1;
        this.marks2 = marks2;
    }
    public double percent_calc() {
        int total_marks = marks1 + marks2;
        return (double) total_marks / 200 * 100;
    }
    public void show() {
        System.out.println("Name: " + name);
        System.out.println("Roll No: " + roll_no);
        System.out.println("Marks 1: " + marks1);
        System.out.println("Marks 2: " + marks2);
    }
}

class Result extends Student {
    private double percentage;
    public Result(String name, int roll_no, int marks1, int marks2)
    { super(name, roll_no, marks1, marks2);
        this.percentage = super.percent_calc();
    }
}

```

```

public void display() {
    super.show();
    System.out.println("Percentage: " + percentage + "%");
}
}

```

```

Name: John Doe
Roll No: 1001
Marks 1: 85
Marks 2: 92
Percentage: 88.5%

```

Week: 7

- 17. Write a Java program to show the use of all keywords for exception handling**

```

Code: public class ExceptionHandlingKeywordsDemo {
    public static void main(String[] args) {
        try {
            int result = 10 / 0;
            int[] arr = new
            int[5]; arr[10] = 50;
        } catch (ArithmaticException ae) {
            System.out.println("Arithmatic Exception
            occurred.");
        } catch (ArrayIndexOutOfBoundsException aioobe) {
            System.out.println("Array Index Out Of Bounds Exception
            occurred.");
        } finally {
            System.out.println("Finally block executed.");
        }
    }
}

```

```

Arithmatic Exception occurred.
Finally block executed.

```

- 19. Write a Java program using try and catch to generate NegativeArrayIndex Exception and Arithmatic Exception.**

```

Code: public class NegativeArrayIndexDemo {
    public static void main(String[] args) {
        try {
            int[] arr = new
            int[5]; arr[-1] = 10;
            int result = 10 / 0;
        } catch (NegativeArraySizeException nae) {
            System.out.println("Negative Array Index Exception
            occurred.");
        } catch (ArithmaticException ae) {

```



```
        System.out.println("Arithmetic Exception occurred.");
    }
}
```

Negative Array Index Exception occurred.

21. Define an exception called "NoMatchFoundException" that is thrown when a string is not equal to "University". Write a program that uses this exception.

```
Code: class NoMatchFoundException extends Exception {  
    public NoMatchFoundException(String message) {  
        super(message);  
    }  
}  
  
public class NoMatchFoundDemo {  
    public static void main(String[] args) {  
        try {  
            String inputString = "College";  
            if (!inputString.equals("University")) {  
                throw new NoMatchFoundException("Input string  
            }  
        } catch (NoMatchFoundException e) {  
            System.out.println(e.getMessage())  
            ;  
        }  
    }  
}
```

Input string does not match 'University'

23. Write a class that keeps a running total of all characters passed to it (one at a time) and throws an exception if it is passed a non-alphabetic character.

```
Code: class NonAlphabeticCharacterException extends Exception
{ public NonAlphabeticCharacterException(String message) {
    super(message);
}
}

public class CharacterTotal
{ private int total;
    public CharacterTotal() {
        total = 0;
    }
    public void addCharacter(char ch) throws NonAlphabeticCharacterException {
        if (!Character.isLetter(ch)) {
            throw new NonAlphabeticCharacterException("Non-alphabetic character encountered: " + ch);
        }
        total++;
    }
    public int getTotal() {
```

```

        return total;
    }
}
public class CharacterTotalDemo {
    public static void main(String[] args) {
        CharacterTotal characterTotal = new CharacterTotal();
        try {
            characterTotal.addCharacter('a');
            characterTotal.addCharacter('b');
            characterTotal.addCharacter('1');
            characterTotal.addCharacter('c');
        } catch (NonAlphabeticCharacterException e)
        { System.out.println(e.getMessage());
        }
        System.out.println("Total characters: " + characterTotal.getTotal());
    }
}

```

Non-alphabetic character encountered: 1
Total characters: 2

3 Write a program called Factorial.java that computes factorials and catches the result in an array of type long for reuse. The long type of variable has its own range. For example 20! Is as high as the range of long type. So check the argument passes and “throw an exception”, if it is too big or too small.

- ❑ If x is less than 0 throw an IllegalArgumentException with a message “Value of x must be positive”.
- ❑ If x is above the length of the array throw an IllegalArgumentException with a message “Result will overflow”. Here x is the value for which we want to find the factorial. Code:

```

public class Factorial {
    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        try {
            long[] factorials = new long[n +
            1]; if (n < 0) {
                throw new IllegalArgumentException("Value of x must be positive");
            } else if (n > factorials.length - 1) {
                throw new IllegalArgumentException("Result will overflow");
            }
            factorials[0] = 1;
            for (int i = 1; i <= n; i++) {
                factorials[i] = factorials[i - 1] * i;
            }
            System.out.println("Factorial of " + n + " is: " +
            factorials[n]); } catch (NumberFormatException e) {
                System.out.println("Invalid input format. Please provide a valid integer.");
            } catch (IllegalArgumentException e) {
                System.out.println(e.getMessage())
                ;
            }
        }
    }
}

```

```

        }
    }
}

java Factorial 5
```

25. Write a class that keeps a running total of all characters passed to it (one at a time) and throws an exception if it is passed a non-alphabetic character.

```

Code: class NonAlphabeticCharacterException extends Exception
{
    public NonAlphabeticCharacterException(String message) {
        super(message);
    }
}

public class CharacterTotal
{
    private int total;
    public CharacterTotal() {
        total = 0;
    }

    public void addCharacter(char ch) throws NonAlphabeticCharacterException {
        if (!Character.isLetter(ch)) {
            throw new NonAlphabeticCharacterException("Non-alphabetic character encountered: " + ch);
        }
        total++;
    }

    public int getTotal() {
        return total;
    }
}

public class CharacterTotalDemo {
    public static void main(String[] args) {
        CharacterTotal characterTotal = new CharacterTotal();
        try {
            characterTotal.addCharacter('a');
            characterTotal.addCharacter('b');
            characterTotal.addCharacter('1'); // Non-alphabetic character
            characterTotal.addCharacter('c');
        } catch (NonAlphabeticCharacterException e) {
            System.out.println(e.getMessage());
        }
        System.out.println("Total characters: " + characterTotal.getTotal());
    }
}
```

```

Non-alphabetic character encountered: 1
Total characters: 2
```

7. Write a program that outputs the name of the capital of the country entered at the

command line. The program should throw a “NoMatchNotFoundException” when it fails to print the capital of the country entered at the command line.

```

Code: class NoMatchNotFoundException extends Exception {
    public NoMatchNotFoundException(String message) {
        super(message);
    }
}

public class CountryCapital {
    public static void main(String[] args) {
        try {
            String country = args[0].toLowerCase();
            String capital = getCapital(country);
            System.out.println("Capital of " + country + " is " + capital);
        } catch (NoMatchNotFoundException e) {
            System.out.println(e.getMessage());
        }
    }
}

public static String getCapital(String country) throws NoMatchNotFoundException
{
    switch (country) {
        case "india":
            return "New Delhi";
        case "usa":
            return "Washington D.C.";
        case "uk":
            return "London";
        default:
            throw new NoMatchNotFoundException("No capital found for country: " + country);
    }
}
}

```

Capital of india is New Delhi

3 Write a program that takes a value at the command line for which factorial is to be computed. The program must convert the string to its integer equivalent. There are three possible user input errors that can prevent the program from executing normally.

- ❑ The first error is when the user provides no argument while executing the program and an **ArrayIndexOutOfBoundsException** is raised. You must write a catch block for this.
- ❑ The second error is **NumberFormatException** that is raised in case the user provides a non-integer (float double) value at the command line.
- ❑ The third error is **IllegalArgumentException**. This needs to be thrown manually if the value at the command line is 0.

```

Code: public class FactorialCalculator {
    public static void main(String[] args) {
        try {
            if (args.length == 0) {

```

```

        throw new ArrayIndexOutOfBoundsException("No argument provided.");
    }
    int num = Integer.parseInt(args[0]);
    if (num <= 0) {
        throw new IllegalArgumentException("Value must be a positive integer greater than 0.");
    }
    long factorial = calculateFactorial(num);
    System.out.println("Factorial of " + num + " is: " +
factorial);
} catch (ArrayIndexOutOfBoundsException e)
{ System.out.println(e.getMessage());
} catch (NumberFormatException e) {
    System.out.println("Invalid input format. Please provide a valid
integer."); } catch (IllegalArgumentException e) {
    System.out.println(e.getMessage());
}
}

public static long calculateFactorial(int n) {
    long factorial = 1;
    for (int i = 1; i <= n; i++) {
        factorial *= i;
    }
    return factorial;
}
}

```

Factorial of 5 is: 120

- 3. Create a user-defined exception named CheckArgument to check the number of arguments passed through the command line. If the number of argument is less than 5, throw the CheckArgumentexception,else print the addition of all the five numbers. Code:** class CheckArgumentException extends Exception {

```

public CheckArgumentException(String message) {
    super(message);
}
}

public class ArgumentChecker { public
static void main(String[] args) {
try {
    if (args.length < 5) {
        throw new CheckArgumentException("Number of arguments must be at least 5.");
    }
    int sum = 0;
    for (String arg : args) {
        sum += Integer.parseInt(arg);
    }
    System.out.println("Sum of all arguments: " + sum);
} catch (CheckArgumentException e) {
    System.out.println(e.getMessage());
}
}

```

```
        } catch (NumberFormatException e) {
            System.out.println("Invalid input format. Please provide valid integers.");
        }
    }
}
```

Sum of all arguments: 15

4. Consider a Student examination database system that prints the mark sheet of students. Input the following from the command line.

(a) Student's Name

(b) Marks in six subjects

These marks should be between 0 to 50. If the marks are not in the specified range, raise a RangeException, else find the total marks and prints the percentage of the students.

```
Code: class RangeException extends Exception {
    public RangeException(String message) {
        super(message);
    }
}

public class MarkSheet {
    public static void main(String[] args) {
        try {
            if (args.length != 7) {
                throw new IllegalArgumentException("Invalid number of arguments. Expected 7.");
            }
            String name = args[0];
            int[] marks = new int[6];
            for (int i = 1; i < args.length; i++) { marks[i - 1] = Integer.parseInt(args[i]); if
                (marks[i - 1] < 0 || marks[i - 1] > 50) {
                    throw new RangeException("Marks should be between 0 to 50.");
                }
            }
            int totalMarks = 0; for
            (int mark : marks) {
                totalMarks += mark;
            }
            double percentage = (double) totalMarks / 300 * 100;
            System.out.println("Student Name: " + name);
            System.out.println("Total Marks: " + totalMarks);
            System.out.println("Percentage: " + percentage +
                "%");
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
        } catch (NumberFormatException e) {
            System.out.println("Invalid input format. Please provide valid integers for marks.");
        } catch (RangeException e) {
            System.out.println(e.getMessage());
        }
    }
}
```



```

        }
    }
}

Student Name: John
Total Marks: 260
Percentage: 86.6666666666667%

```

\} Write a java program to create an custom Exception that would handle at least 2 kind of Arithmetic Exceptions while calculating a given equation (e.g. X+Y*(P/Q)Z-I)

Code: // 11. Write a java program to create a custom Exception that would handle at least 2 kinds of Arithmetic Exceptions while calculating a given equation (e.g. X+Y*(P/Q)Z-I)

```

Code: class CustomArithmeticException extends Exception {
    public CustomArithmeticException(String message) {
        super(message);
    }
}

public class EquationCalculator {
    public static void main(String[] args) {
        try {
            int x = Integer.parseInt(args[0]);
            int y = Integer.parseInt(args[1]);
            int p = Integer.parseInt(args[2]);
            int q = Integer.parseInt(args[3]);
            int z = Integer.parseInt(args[4]);
            int i = Integer.parseInt(args[5]);

            double result = calculateEquation(x, y, p, q, z, i);
            System.out.println("Result of the equation: " +
                result);
        } catch (NumberFormatException e) {
            System.out.println("Invalid input format. Please provide valid integers.");
        } catch (CustomArithmeticException e)
            { System.out.println(e.getMessage());
        }
    }
}

```

```

    public static double calculateEquation(int x, int y, int p, int q, int z, int i)
throws CustomArithmeticException {
    try {
        return x + y * (p / q) * z - i;
    } catch (ArithmetricException ae) {
        throw new CustomArithmeticException("Arithmetric Exception occurred while calculating
the equation.");
    }
}

```

```
java EquationCalculator 10 5 20 0 3 2
```

24. Create two user-defined exceptions named “TooHot” and “TooCold” to check the temperature (in Celsius) given by the user passed through the command line is too hot or too cold.

If temperature > 35, throw exception
☒ “TooHot”. If temperature <5, throw exception
☒ “TooCold”.

☒ Otherwise, print “Normal” and convert it to Farenheit.

```
Code: class TooHot extends Exception {  
    public TooHot(String message) {  
        super(message);  
    }  
}  
class TooCold extends Exception {  
    public TooCold(String message) {  
        super(message);  
    }  
}  
public class TemperatureCheck {  
    public static void main(String[] args) {  
        try {  
            int temperature = Integer.parseInt(args[0]);  
            if (temperature > 35) {  
                throw new TooHot("Temperature is too hot!");  
            }  
            else if (temperature < 5) {  
                throw new TooCold("Temperature is too cold!");  
            }  
            else {  
                System.out.println("Normal");  
                double fahrenheit = (temperature * 9.0 / 5.0) + 32.0;  
                System.out.println("Temperature in Fahrenheit: " + fahrenheit);  
            }  
        } catch (TooHot e) {  
            System.out.println(e.getMessage())  
        ;  
        } catch (TooCold e) {  
            System.out.println(e.getMessage())  
        ;  
        } catch (NumberFormatException e) {  
            System.out.println("Please provide a valid integer temperature as input.");  
        } catch (ArrayIndexOutOfBoundsException e) {  
            System.out.println("Please provide the temperature as a command line argument.");  
        }  
    }  
}
```


Normal**Temperature in Fahrenheit: 77.0**

```
// Consider an Employee recruitment system that prints the candidate name based  
on the age criteria. The name and age of the candidate are taken as Input.Create two  
user-defined exceptions named “TooOlder” and “TooYounger”  
If age>45, throw exception “TooOlder”.  
If age<20, throw exception “TooYounger”.  
Otherwise, print “Eligible” and print the name of the candidate.  
Code: class TooOlder extends Exception {  
    public TooOlder(String message) {  
        super(message);  
    }  
}  
class TooYounger extends Exception {  
    public TooYounger(String message) {  
        super(message);  
    }  
}  
public class EmployeeRecruitment {  
    public static void main(String[] args) {  
        try {  
            String name = args[0];  
            int age =  
                Integer.parseInt(args[1]); if (age >  
                45) {  
                    throw new TooOlder("Candidate is too old for  
recruitment!"); } else if (age < 20) {  
                    throw new TooYounger("Candidate is too young for recruitment!");  
                } else { System.out.println("Eligible");  
                    System.out.println("Candidate Name: " +  
                        name);  
                }  
            } catch (TooOlder e) {  
                System.out.println(e.getMessage())  
                ;  
            } catch (TooYounger e) {  
                System.out.println(e.getMessage())  
                ;  
            } catch (NumberFormatException e) {  
                System.out.println("Please provide a valid integer age as input.");  
            } catch (ArrayIndexOutOfBoundsException e) {  
                System.out.println("Please provide name and age as command line arguments.");  
            }  
        }  
    }
```



```
Eligible
```

```
Candidate Name: John
```

\} Consider a “Binary to Decimal” Number conversion system which only accepts binary number as Input. If user provides a decimal number a custom Exception

“WrongNumberFormat” exception will be thrown. Otherwise, it will convert into decimal and print into the screen.

Code: class WrongNumberFormatException extends Exception {

```
public WrongNumberFormatException(String message) {
    super(message);
}
```

```
public class BinaryToDecimalConverter
```

```
{
```

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

```
try {
```

```
String binaryNumber = args[0];
```

```
if (!binaryNumber.matches("[01]+")) {
```

```
    throw new WrongNumberFormatException("Input is not a binary number!");
```

```
}
```

```
int decimalNumber = Integer.parseInt(binaryNumber, 2);
```

```
System.out.println("Decimal equivalent: " + decimalNumber);
```

```
} catch (WrongNumberFormatException e) {
```

```
    System.out.println(e.getMessage())
```

```
;
```

```
} catch (NumberFormatException e) {
```

```
    System.out.println("Please provide a valid binary number as input."); } catch (ArrayIndexOutOfBoundsException e) {
```

```
    System.out.println("Please provide the binary number as a command line argument.");
```

```
}
```

```
}
```

```
Decimal equivalent: 10
```

\} Write a Java Program that Implement the Nested Try

Statements. Code: public class NestedTryExample {

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

```
try {
```

```
    System.out.println("Outer try block starts");
```

```
    int result = 10 / 0; // This will throw ArithmeticException
```

```
    try {
```

```
        System.out.println("Inner try block starts");
```

```
        String str = null;
```

```
        System.out.println(str.length());
```

```
        System.out.println("Inner try block ends");
```

```
} catch (NullPointerException e) {
```



```
        System.out.println("Caught NullPointerException: " + e.getMessage());
    }
    System.out.println("Outer try block ends");
} catch (ArithmaticException e) {
    System.out.println("Caught ArithmaticException: " + e.getMessage());
}
}
```

```
Outer try block starts
Caught ArithmeticException: / by zero
Outer try block ends
```

\} Write a Java Program to Create Account with 500 Rs Minimum Balance, Deposit Amount, Withdraw Amount and Also Throws LessBalanceException.

Java Program Which has a Class Called LessBalanceException Which returns the Statement that Says WithDraw Amount(_Rs) is Not Valid

Q Java Program that has a Class Which Creates 2 Accounts, Both Account Deposit Money and One Account Tries to Withdraw more Money Which Generates a LessBalanceException Take Appropriate Action for the Same.

```
Code: class LessBalanceException extends Exception {  
    public LessBalanceException(String message) {  
        super(message);  
    }  
}
```

```
class Account {  
    private double balance;  
    private static final double MIN_BALANCE = 500; // Minimum balance required  
    public Account() {  
        balance = MIN_BALANCE; // Initialize balance with minimum balance  
    }  
    public void deposit(double amount) { balance +=  
        amount; System.out.println("Amount deposited: " +  
        amount); System.out.println("Current balance: " +  
        balance);  
    }  
    public void withdraw(double amount) throws LessBalanceException {  
        if (balance - amount < MIN_BALANCE) {  
            throw new LessBalanceException("Withdrawal amount exceeds available  
        }  
        balance -= amount;  
        System.out.println("Amount withdrawn: " + amount);  
        System.out.println("Current balance: " + balance);  
    }  
}
```

```
public class BankAccountManagement {
```

```

public static void main(String[] args) {
    try {
        Account account = new Account();
        account.deposit(1000);
        account.withdraw(700);
    } catch (LessBalanceException e) {
        System.out.println(e.getMessage());
    }
}
}

```

```

Amount deposited: 1000.0
Current balance: 1500.0
Amount withdrawn: 700.0
Current balance: 800.0

```

2. Consider a Library Management System, where a user wants to find a book. If the book is present in Library (Hint: Use predefined array), then it will print the book. Otherwise it will throw an exception “BookNotFoundException”.

```

Code: class BookNotFoundException extends Exception {
    public BookNotFoundException(String message) {
        super(message);
    }
}

public class LibraryManagementSystem {
    private static final String[] books = {"Book1", "Book2", "Book3", "Book4", "Book5"};
    public static void findBook(String bookTitle) throws BookNotFoundException {
        boolean found = false;
        for (String book : books) {
            if (book.equalsIgnoreCase(bookTitle)) {
                System.out.println("Book found: " +
                    book); found = true;
                break;
            }
        }
        if (!found) {
            throw new BookNotFoundException("Book not found in the library!");
        }
    }

    public static void main(String[] args) {
        try {
            findBook("Book3");
            findBook("Book6");
        } catch (BookNotFoundException e) {
            System.out.println(e.getMessage());
        }
    }
}

```

}

```
Book found: Book3
Book not found in the library!
```

3. Consider a Quiz Management System, where a user needs to answer 5 questions. If any of the answer is wrong, throw an exception “NotCorrectException”. If the answer is correct give a message “good! The answer is correct”.

Code:

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

public class QuizManagementSystem {
    public static void checkAnswer(String userAnswer, String correctAnswer, int questionNumber) throws NotCorrectException {
        if (!userAnswer.equalsIgnoreCase(correctAnswer)) {
            throw new NotCorrectException("Incorrect answer for question " + questionNumber + "!");
        }
        System.out.println("Good! The answer to question " + questionNumber + " is correct.");
    }

    public static void main(String[] args) {
        try {
            String[] questions = {"Q1: What is the capital of France?", "Q2: What is the capital of Japan?",
                "Q3: What is the largest ocean?", "Q4: Who wrote Romeo and Juliet?", "Q5: What is the chemical symbol for water?"};

            String[] correctAnswers = {"Paris", "Tokyo", "Pacific", "William Shakespeare", "H2O"};
            String[] userAnswers = {"Paris", "Tokyo", "Atlantic", "William Shakespeare", "H2O"};
            for (int i = 0; i < questions.length; i++) {
                checkAnswer(userAnswers[i], correctAnswers[i], i + 1);
            }
        } catch (NotCorrectException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

```
Good! The answer to question 1 is correct.
Good! The answer to question 2 is correct.
Incorrect answer for question 3!
```

5. Write a program to raise a user defined exception if username is less than 6 characters and password does not match.

Code:

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

```

}

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

public class AuthenticationSystem {
    public static void authenticate(String username, String password, String confirmPassword)
throws InvalidUsernameException, PasswordMismatchException {
        if (username.length() < 6) {
            throw new InvalidUsernameException("Username must be at least 6 characters long!");
        }
        if (!password.equals(confirmPassword)) {
            throw new PasswordMismatchException("Passwords do not match!");
        }
        System.out.println("Authentication successful!");
    }

    public static void main(String[] args) {
        try {
            String username = "user123";
            String password = "password123";
            String confirmPassword = "password123"; // Correct password
            confirmation.authenticate(username, password, confirmPassword);
        } catch (InvalidUsernameException | PasswordMismatchException e)
        { System.out.println(e.getMessage());
        }
    }
}

```

Authentication successful!

6. Write a program to accept a password from the user and throw 'Authentication Failure' exception if the password is incorrect.

Code:

```

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

public class PasswordAuthentication {
    public static void authenticatePassword(String enteredPassword, String correctPassword)
throws AuthenticationFailureException {
        if (!enteredPassword.equals(correctPassword)) {
            throw new AuthenticationFailureException("Authentication failed! Incorrect password.");
        }
        System.out.println("Authentication successful!");
    }

    public static void main(String[] args) {

```

```

try {
    String correctPassword = "password123";
    String enteredPassword = "password123"; // Correct password
    authenticatePassword(enteredPassword, correctPassword);
} catch (AuthenticationFailureException e)
    { System.out.println(e.getMessage());
}
}
}

```

Authentication successful!

8. Write a program to input name and age of a person and throw a user-defined exception, if the entered age is negative.

Code:

```

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

public class PersonInfo {
    public static void validateAge(int age) throws NegativeAgeException
    { if (age < 0) {
        throw new NegativeAgeException("Age cannot be negative!");
    }
        System.out.println("Name and age input successful!");
    }

    public static void main(String[] args) {
        try {
            String name = "John"; // Assume name is
            predefined int age = -25; // Negative age
            validateAge(age);
        } catch (NegativeAgeException e) {
            System.out.println(e.getMessage());
        }
    }
}

```

Age cannot be negative!

9. Write a program to throw user defined exception if the given number is not positive. **Code:**

```

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

public class PositiveNumberChecker {
    public static void checkPositiveNumber(int number) throws NonPositiveNumberException {
        if (number <= 0) {

```

```

        throw new NonPositiveNumberException("Number must be positive!");
    }
    System.out.println("Number input successful!");
}
public static void main(String[] args) {
    try {
        int number = -10; // Non-positive number
        checkPositiveNumber(number);
    } catch (NonPositiveNumberException e)
        { System.out.println(e.getMessage());
    }
}
}

```

Number must be positive!

11. Write a program to throw a user-defined exception "String Mismatch Exception", if two strings are not equal (ignore the case).

```

Code: class StringMismatchException extends Exception {
    public StringMismatchException(String message) {
        super(message);
    }
}
public class StringComparator {
    public static void compareStrings(String str1, String str2) throws StringMismatchException {
        if (!str1.equalsIgnoreCase(str2)) {
            throw new StringMismatchException("Strings do not match!");
        }
        System.out.println("Strings match!");
    }
    public static void main(String[] args) {
        try {
            String str1 = "Hello";
            String str2 = "hello";
            compareStrings(str1, str2);
        } catch (StringMismatchException e) {
            System.out.println(e.getMessage())
            ;
        }
    }
}

```

Strings do not match!

13. Design a stack class. Provide your own stack exceptions namely push exception and pop exception, which throw exceptions when the stack is full and when the stack is empty respectively. Show the usage of these exceptions in handling a stack object in the main.

```

Code: class PushException extends Exception {
    public PushException(String message) {

```

```
        super(message);
    }
}
class PopException extends Exception {
    public PopException(String message) {
        super(message);
    }
}
class MyStack {
    private int[] stackArray;
    private int top;
    private int maxSize;
    public MyStack(int size) {
        maxSize = size;
        stackArray = new int[maxSize];
        top = -1; // Empty stack initially
    }
    public void push(int element) throws PushException {
        if (top == maxSize - 1) {
            throw new PushException("Stack overflow! Cannot push element onto full stack.");
        }
        stackArray[++top] = element;
    }
    public int pop() throws PopException {
        if (top == -1) {
            throw new PopException("Stack underflow! Cannot pop element from empty stack.");
        }
        return stackArray[top--];
    }
    public boolean isEmpty() {
        return (top == -1);
    }
    public boolean isFull() {
        return (top == maxSize - 1);
    }
}
public class StackDemo {
    public static void main(String[] args) {
        MyStack stack = new MyStack(5);
        try {
            for (int i = 1; i <= 5; i++) {
                stack.push(i);
            }
            stack.push(6);
        } catch (PushException e) {
            System.out.println(e.getMessage());
        }
        try {
```

```
        while (!stack.isEmpty()) {
            System.out.println("Popped: " + stack.pop());
        }
        stack.pop();
    } catch (PopException e) {
        System.out.println(e.getMessage());
    }
}

Stack overflow! Cannot push element onto full stack.
Popped: 5
Popped: 4
Popped: 3
Popped: 2
Popped: 1
Stack underflow! Cannot pop element from empty stack.
}
```

\} Write an application that displays a series of at least five student ID numbers (that you have stored in an array) and asks the user to enter a numeric test score for the student. Create a ScoreException class, and throw a ScoreException for the class if the user does not enter a valid score (zero to 100). Catch the ScoreException and then display an appropriate message. In addition, store a 0 for the student's score. At the end of the application, display all the student IDs and scores.

Code:

```
class ScoreException extends Exception {
    public ScoreException(String message) {
        super(message);
    }
}
public class StudentScores {
    public static void main(String[] args) {
        int[] studentIDs = {101, 102, 103, 104, 105}; int[] scores =
            new int[studentIDs.length]; java.util.Scanner scanner = new
            java.util.Scanner(System.in); for (int i = 0; i <
            studentIDs.length; i++) {
                System.out.print("Enter test score for student " + studentIDs[i] + ":" +
                    "); try {
                    int score =
                        scanner.nextInt(); if (score <
                        0 || score > 100) {
                            throw new ScoreException("Invalid test score! Score must be between 0 and 100.");
                        }
                    scores[i] = score; // Store valid score
                } catch (ScoreException e) {
                    System.out.println(e.getMessage());
                    scores[i] = 0;
                }
            }
        }
```



```

        System.out.println("\nStudent IDs and Test Scores:");
        for (int i = 0; i < studentIDs.length; i++) {
            System.out.println("Student ID: " + studentIDs[i] + ", Test Score: " + scores[i]);
        }
        scanner.close();
    }
}

Enter test score for student 101: 90
Enter test score for student 102: 105
Invalid test score! Score must be between 0 and 100.
Enter test score for student 103: -5
Invalid test score! Score must be between 0 and 100.
Enter test score for student 104: 80
Enter test score for student 105: 95

Student IDs and Test Scores:
Student ID: 101, Test Score: 90
Student ID: 102, Test Score: 0
Student ID: 103, Test Score: 0
Student ID: 104, Test Score: 80
Student ID: 105, Test Score: 95

```

Week: 8

- 15. Write a Java program for calculating Factorial. Number should be taken through user input (Using Scanner, BufferedReader both).**

```

Code: import java.util.Scanner;
public class FactorialCalculator {

    public static void main(String[] args) { Scanner scanner = new
        Scanner(System.in); System.out.print("Enter a number to
        calculate its factorial: "); int number = scanner.nextInt();

        int factorial = calculateFactorial(number);
        System.out.println("Factorial of " + number + " is: " +
        factorial); scanner.close();
    }

    private static int calculateFactorial(int n)
    { if (n == 0)
        return
    1; else
        return n * calculateFactorial(n - 1);
    }
}

```

```
Enter a number to calculate its factorial: 10
Factorial of 10 is: 3628800
PS D:\RN_1\Coding\JAVA\WEEK8\src> 
```

2. Design a palindrome class that will input a string from console and check whether the string is palindrome or not.

Code: import java.util.Scanner;

```
public class PalindromeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string to check if it's a palindrome: ");
        String input = scanner.nextLine(); if (isPalindrome(input))

            System.out.println(input + " is a palindrome.");
        else
            System.out.println(input + " is not a palindrome.");
        scanner.close();
    }

    private static boolean isPalindrome(String str) { StringBuilder
        reversed = new StringBuilder(str).reverse(); return
        str.equals(reversed.toString());
    }
}

Enter a string to check if it's a palindrome: madam
madam is a palindrome.
```

3. Write a Java program to merge two strings. Code:

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

```
public class StringMerger {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the first string: ");
        String first = scanner.nextLine();
        System.out.print("Enter the second string: ");
        String second = scanner.nextLine();
        String merged = mergeStrings(first, second);
        System.out.println("Merged string: " + merged);
        scanner.close();
    }

    String mergeStrings(String first, String second) {
        return first + second;
    }
}
```

```

    }
private static String mergeStrings(String first, String second) {
    return first + second;
}
}

```

```

Enter the first string: UEM
Enter the second string: Kolkata
Merged string: UEMKolkata

```

4. Write a Java program for reverse a string. (String will be taken as user input through console).

Code: import java.util.Scanner;

```

public class StringReverser {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string to reverse: ");
        String input = scanner.nextLine();
        String reversed = reverseString(input);
        System.out.println("Reversed string: " + reversed);
        scanner.close();
    }
    private static String reverseString(String str) {
        return new StringBuilder(str).reverse().toString();
    }
}

```

```

Enter a string to reverse: UEM
Reversed string: MEU

```

5. Write a Java Program to Concatenate Two

Strings. **Code:** import java.util.Scanner;

```

public class StringConcatenation {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the first string: ");
        String first = scanner.nextLine();
        System.out.print("Enter the second string: ");
        String second = scanner.nextLine();
        String concatenated = first.concat(second);
        System.out.println("Concatenated string: " + concatenated);
        scanner.close();
    }
}

```

```

    }
}

Enter the first string: UEM
Enter the second string: K
Concatenated string: UEMK

```

6. Write a Java Program to check if a Given String is getChar from Specific**Index.** **Code:** import java.util.Scanner;

```

public class CharAtIndexChecker {
    public static void main(String[] args) {
        Scanner scanner = new
        Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = scanner.nextLine();
        System.out.print("Enter an index: ");
        int index = scanner.nextInt();
        char character = getCharAtIndex(str, index);
        if (character != '\0')
            System.out.println("Character at index " + index + " is: " +
        character); else
            System.out.println("Invalid index.");
        scanner.close();
    }
}

```

```

private static char getCharAtIndex(String str, int index)
{ if (index >= 0 && index < str.length())
    return
    str.charAt(index); else
    return '\0';
}
}

```

```

Enter a string: UEMK
Enter an index: 2
Character at index 2 is: M

```

7. Write a Java Program to Find the Length of the**String.** **Code:** import java.util.Scanner;

```

public class StringLengthFinder {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = scanner.nextLine();
    }
}

```

```
int length = findStringLength(str);
System.out.println("Length of the string is: " + length);
scanner.close();
}

private static int findStringLength(String str) {
    return str.length();
}
}

Enter a string: UEMK
Length of the string is: 4
```

8. Write a Java Program to Find All Possible Subsets of given Length in String.

Code :

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class SubsetsOfGivenLength {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = scanner.nextLine();
        System.out.print("Enter the length of subsets: ");
        int length = scanner.nextInt();

        List<String> subsets = findAllSubsetsOfLength(str, length);
        System.out.println("All subsets of length " + length + " "
            + "are:");
        for (String subset : subsets) {
            System.out.println(subset);
        }
        scanner.close();
    }

    private static List<String> findAllSubsetsOfLength(String str, int length) {
        List<String> subsets = new ArrayList<>();
        for (int i = 0; i <= str.length() - length; i++) {
            subsets.add(str.substring(i, i + length));
        }
        return subsets;
    }
}
```

```
Enter a string: UEMK
Enter the length of subsets: 2
All subsets of length 2 are:
UE
EM
MK
```

9. Write a Java Program to Remove the White Spaces from a String.

Code :

```
import java.util.Scanner;
public class RemoveWhiteSpace {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string with white spaces: ");
        String str = scanner.nextLine();
        String stringWithoutSpaces = removeWhiteSpace(str);
        System.out.println("String without white spaces: " + stringWithoutSpaces);
        scanner.close();
    }

    private static String removeWhiteSpace(String str) {
        return str.replaceAll("\\s", "");
    }
}
```

```
Enter a string with white spaces: u e m k
String without white spaces: uemk _
```

10. Write a Java Program to Compare two Strings.

Code :

```
import java.util.Scanner;
public class StringComparer {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the first string: ");
        String first = scanner.nextLine();
        System.out.print("Enter the second string: ");
        String second = scanner.nextLine();
        boolean isEqual = compareStrings(first, second);
        if (isEqual)
            System.out.println("Both strings are equal.");
        else
            System.out.println("Both strings are not equal.");
    }

    private static boolean compareStrings(String first, String second) {
        return first.equals(second);
    }
}
```

```
        System.out.println("Strings are not equal.");
        scanner.close();
    }

private static boolean compareStrings(String first, String second)
    { return first.equals(second);
    }
}

Enter the first string: UEM
Enter the second string: K
Strings are not equal.
```

11. Write a Java Program to Compare Performance of Two

```
Strings. Code : public class StringPerformanceComparator {
public static void main(String[] args) {
    String string1 = "Hello";
    String string2 =
    "World";
    long startTimeConcat1 =
    System.nanoTime(); for (int i = 0; i < 100000;
    i++) {
        String result = string1 + string2;
    }
    long endTimeConcat1 = System.nanoTime();
    long durationConcat1 = endTimeConcat1 - startTimeConcat1;

    long startTimeBuilder1 = System.nanoTime();
    for (int i = 0; i < 100000; i++) {
        StringBuilder builder = new StringBuilder(string1);
        builder.append(string2);
        String result = builder.toString();
    }
    long endTimeBuilder1 = System.nanoTime();
    long durationBuilder1 = endTimeBuilder1 - startTimeBuilder1;
    long startTimeConcat2 = System.nanoTime(); for (int i = 0; i <
    100000; i++) {
        String result = string2 + string1;
    }
    long endTimeConcat2 = System.nanoTime();
    long durationConcat2 = endTimeConcat2 - startTimeConcat2;
    long startTimeBuilder2 = System.nanoTime(); for (int i = 0; i <
    100000; i++) {
```



```

        StringBuilder builder = new StringBuilder(string2);
        builder.append(string1);
        String result = builder.toString();
    }
    long endTimeBuilder2 = System.nanoTime();
    long durationBuilder2 = endTimeBuilder2 - startTimeBuilder2;
    System.out.println("Performance comparison of string concatenation:");
    System.out.println("String1 + String2: " + durationConcat1 + " nanoseconds");
    System.out.println("StringBuilder for String1: " + durationBuilder1 + "
nanoseconds"); System.out.println("String2 + String1: " + durationConcat2 + "
nanoseconds"); System.out.println("StringBuilder for String2: " + durationBuilder2 + "
nanoseconds");
}
}

Performance comparison of string concatenation:
String1 + String2: 16216000 nanoseconds
StringBuilder for String1: 18277900 nanoseconds
String2 + String1: 9156100 nanoseconds
StringBuilder for String2: 4513700 nanoseconds

```

6 Write a Java Program to Use Equals Method In a String Class.

Code : import java.util.Scanner;

```

public class EqualsMethodUsage {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the first string: ");
        String first = scanner.nextLine();
        System.out.print("Enter the second string:
"); String second = scanner.nextLine();
        boolean areEqual = useEqualsMethod(first,
second); if (areEqual)
            System.out.println("Both strings are
equal."); else
            System.out.println("Strings are not equal.");
        scanner.close();
    }
    private static boolean useEqualsMethod(String first, String second)
    { return first.equals(second);
    }
}

Enter the first string: UEM
Enter the second string: Kolkata
Strings are not equal.

```

Write a Java Program to Use EqualsIgnoreCase Method In a String Class.

Code :

```
import java.util.Scanner;
public class EqualsIgnoreCaseMethodUsage {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the first string: ");
        String first = scanner.nextLine();
        System.out.print("Enter the second string: ");
        String second = scanner.nextLine();
        boolean areEqualIgnoreCase = useEqualsIgnoreCaseMethod(first, second);
        if (areEqualIgnoreCase)
            System.out.println("Both strings are equal ignoring
case.");
        else
            System.out.println("Strings are not equal ignoring case.");
        scanner.close();
    }
    private static boolean useEqualsIgnoreCaseMethod(String first, String second)
    { return first.equalsIgnoreCase(second);
    }
}
```

Enter the first string: UEM
Enter the second string: UEm
Both strings are equal ignoring case.

13. Write a Java Program to Use compareTo Method In a String Class.

Code :

```
import java.util.Scanner;
public class CompareToMethodUsage {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the first string: ");
        String first = scanner.nextLine();
        System.out.print("Enter the second string: ");
        String second = scanner.nextLine();
        int comparisonResult = useCompareToMethod(first, second);
        if (comparisonResult == 0)
            System.out.println("Both strings are equal.");
        else if (comparisonResult < 0)
            System.out.println("First string is lexicographically smaller than the second
string.");
        else
            System.out.println("First string is lexicographically greater than the second string.");
    }
}
```

```

        scanner.close();
    }

private static int useCompareToMethod(String first, String second)
    { return first.compareTo(second);
    }
}

Enter the first string: UEM
Enter the second string: UEM
Both strings are equal.

```

JJJ. With a Java Program to Use compareTolgnoreCase Method In a String**Class. Code:** import java.util.Scanner;

```

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

        Scanner scanner = new
        Scanner(System.in);
        System.out.println("Enter first string:");
        String str1 = scanner.nextLine();
        System.out.println("Enter second string:");
        String str2 = scanner.nextLine();

        int result =
        str1.compareTo(str2); if (result
        == 0) {
            System.out.println("Both strings are equal.");
        } else if (result < 0) {
            System.out.println("First string is lexicographically less than second
            string."); } else {
            System.out.println("First string is lexicographically greater than second string.");
        }
    }
}

Enter first string:
UEM
Enter second string:
Kolkata
First string is lexicographically greater than second string.

```

15. Write a Java Program to Replace Character or String.**Code:** import java.util.Scanner;

```

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

        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:"); String input =
        scanner.nextLine(); System.out.println("Enter the
        character/string to replace:");
    }
}

```



```

String toReplace = scanner.nextLine();
System.out.println("Enter the replacement character/string:");
String replacement = scanner.nextLine();
String replacedString = input.replace(toReplace,
replacement); System.out.println("String after
replacement:"); System.out.println(replacedString);
}
}

Enter a string:
UEMK
Enter the character/string to replace:
M
Enter the replacement character/string:
K
String after replacement:
UEKK

```

17. Write a Java Program to Search Last Occurrence of a Substring Inside a Substring.

Code : import java.util.Scanner;

```

public class LastOccurrence{
    public static void main(String[] args) { Scanner scanner =
        new Scanner(System.in); System.out.println("Enter a
        string:"); String input = scanner.nextLine();
        System.out.println("Enter the substring to search for:");
        String substring = scanner.nextLine();

        int lastIndex = input.lastIndexOf(substring);

        if (lastIndex != -1) {
            System.out.println("Last occurrence of substring is at index: " + lastIndex);
        } else {
            System.out.println("Substring not found in the string.");
        }
    }
}

Enter a string:
UEM
Enter the substring to search for:
E
Last occurrence of substring is at index: 1

```

18. Write a Java Program to Remove a Particular Character from a String.

Code: import java.util.Scanner;

```
public class RemoveCharacter {  
    public static void main(String[] args) {  
        Scanner scanner = new Scanner(System.in);  
        System.out.println("Enter a string:");  
        String input = scanner.nextLine();  
        System.out.println("Enter the character to remove:");  
        char charToRemove = scanner.nextLine().charAt(0);  
  
        String result = input.replaceAll(String.valueOf(charToRemove), "");  
  
        System.out.println("String after removing '" + charToRemove + "'");  
        System.out.println(result);  
    }  
}  
Enter a string:  
UEMK  
Enter the character to remove:  
K  
String after removing 'K':  
UEM
```

18. Write a Java Program to Replace a Substring Inside a String by Another

One. Code: import java.util.Scanner;

```
public class ReplaceSubstring { public  
    static void main(String[] args) {  
  
        Scanner scanner = new Scanner(System.in);  
        System.out.println("Enter a string:"); String input =  
        scanner.nextLine(); System.out.println("Enter the  
        substring to replace:"); String toReplace =  
        scanner.nextLine(); System.out.println("Enter the  
        replacement substring:"); String replacement =  
        scanner.nextLine();  
  
        String replacedString = input.replace(toReplace,  
        replacement); System.out.println("String after  
        replacement:"); System.out.println(replacedString);  
    }  
}
```

```
Enter a string:  
UEMK  
Enter the substring to replace:  
E  
Enter the replacement substring:  
K  
String after replacement:  
UKMK
```

Write a Java Program to Reverse a

String. Code : import
java.util.Scanner; public class
ReverseString {
 public static void main(String[] args) {
 Scanner scanner = new
 Scanner(System.in);
 System.out.println("Enter a string:"); String
 input = scanner.nextLine();
 String reversed = reverseString(input);
 System.out.println("Reversed
 string:");
 System.out.println(reversed);
 }
 public static String reverseString(String str) { return
 new StringBuilder(str).reverse().toString();
 }
}

```
Enter a string:  
madam  
Reversed string:  
madam
```

22. Write a Java Program to Search a Word Inside a

String. Code : import java.util.Scanner;

```
public class WordSearch {  
    public static void main(String[] args) {  
        Scanner scanner = new Scanner(System.in);  
        System.out.println("Enter a string:");  
        String input = scanner.nextLine();  
        System.out.println("Enter the word to search for:");  
        String word = scanner.nextLine();  
        boolean found = input.contains(word);  
        if (found) {  
            System.out.println("Word " + word + " found in the string.");  
        } else {  
            System.out.println("Word " + word + " not found in the string.");  
        }  
    }  
}
```



```
    }
}

Enter a string:
java
Enter the word to search for:
a
Word 'a' found in the string.
```

22. Write a Java Program to Split a String into a Number of Substrings.

Code :

```
import java.util.Scanner;
public class SplitString {
    public static void main(String[] args) {
        Scanner scanner = new
        Scanner(System.in);
        System.out.println("Enter a string:");
        String
        input = scanner.nextLine();
        System.out.println("Enter the delimiter:");
        String delimiter = scanner.nextLine();

        String[] substrings = input.split(delimiter);

        System.out.println("Substrings:");
        for (String substring : substrings) {
            System.out.println(substring);
        }
    }
}
```

```
Enter a string:
UEM,UEMK
Enter the delimiter:
,
Substrings:
UEM
UEMK
```

23. Write a Java Program to Search a Particular Word in a String.**24. Write a Java Program to Replace All Occurings of a String.**

Code:

```
import java.util.Scanner;
public class ReplaceOccurrences {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine();
        System.out.println("Enter the string to replace:");

        String replacedString = input.replace("oldString", "newString");

        System.out.println("Replaced String:");
        System.out.println(replacedString);
    }
}
```

```

String toReplace = scanner.nextLine();
System.out.println("Enter the replacement string:");
String replacement = scanner.nextLine();

String replacedString = input.replaceAll(toReplace, replacement);

System.out.println("Replaced string:");
System.out.println(replacedString);
}

}

Enter a string:
test
Enter the string to replace:
t
Enter the replacement string:
java
Replaced string:
javaesjava

```

25. Write a Java Program to Make First Character of Each Word in Uppercase. **Code :**

```

import java.util.Scanner;

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

        Scanner scanner = new
        Scanner(System.in);
        System.out.println("Enter a string:");
        String
        input = scanner.nextLine();

        String[] words = input.split("\s+");
        StringBuilder result = new StringBuilder();
        for (String word : words) {
            if (!word.isEmpty()) {
                result.append(Character.toUpperCase(word.charAt(0)))
                    .append(word.substring(1)).append(" ");
            }
        }
        System.out.println("String with first letter of each word in uppercase:");
        System.out.println(result.toString().trim());
    }
}

Enter a string:
java language
String with first letter of each word in uppercase:
Java Language

```

26. Write a Java Program to Delete All Repeated Words in String.

```
Code : import java.util.Scanner;
public class RemoveChar {
    public static void main(String[] args) { Scanner
        scanner = new Scanner(System.in);
        System.out.println("Enter the first string:");
        String first = scanner.nextLine();
        System.out.println("Enter the second
        string:"); String second = scanner.nextLine();

        StringBuilder result = new StringBuilder();
        for (int i = 0; i < second.length(); i++) {
            char currentChar = second.charAt(i);
            if (first.indexOf(currentChar) == -1) {
                result.append(currentChar);
            }
        }

        System.out.println("Resultant string after removal:");
        System.out.println(result.toString());
    }
}

Enter the first string:
java
Enter the second string:
abcd
Resultant string after removal:
bcd
```

27. Write a Java Program to Reverse the String Using Both Recursion and Iteration.

```
Code : import java.util.Scanner;
public class ReverseString {
    public static void main(String[] args) {
        Scanner scanner = new
        Scanner(System.in);
        System.out.println("Enter a string:"); String
        input = scanner.nextLine();
        String reversed = reverseString(input);
        System.out.println("Reversed
        string:");
        System.out.println(reversed);
    }

    public static String reverseString(String str) {
        return new StringBuilder(str).reverse().toString();
    }
}
```



```
}
```

```
}
```

```
Enter a string:
```

```
kolkata
```

```
Reversed string:
```

```
ataklok
```

28. Write a Java Program to Convert a String Totally into Upper Case.

Code :

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

```
public classUpperCaseConversion {
```

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

```
        Scanner scanner = new
```

```
        Scanner(System.in);
```

```
        System.out.println("Enter a string:"); String
```

```
        input = scanner.nextLine();
```

```
        String upperCase = input.toUpperCase();
```

```
        System.out.println("String in upper case:");
```

```
        System.out.println(upperCase);
```

```
    }
```

```
}
```

```
Enter a string:
```

```
kolkata
```

```
String in upper case:
```

```
KOLKATA
```

29. Write a Java Program to Remove all Characters in Second String which are Present in First String.

Code :

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

```
public class ReplaceCharacter { public
```

```
    static void main(String[] args) {
```

```
        Scanner scanner = new Scanner(System.in);
```

```
        System.out.println("Enter a string:"); String input =
```

```
        scanner.nextLine(); System.out.println("Enter the
```

```
        character/string to replace:"); String toReplace =
```

```
        scanner.nextLine(); System.out.println("Enter the replacement
```

```
        character/string:"); String replacement = scanner.nextLine();
```



```
        String replacedString = input.replace(toReplace,
```

```
        replacement); System.out.println("String after
```

```
        replacement:"); System.out.println(replacedString);
```

```
    }
```

```
}
```

```

Enter a string:
java
Enter the character/string to replace:
a
Enter the replacement character/string:
aa
String after replacement:
jaavaa

```

30. Write a Java Program to Find the Consecutive Occurrence of any Vowel in a String.

```

Code : import java.util.Scanner;

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

        Scanner scanner = new
        Scanner(System.in);
        System.out.println("Enter a string:"); String
        input = scanner.nextLine();

        boolean hasConsecutiveVowels =
        checkConsecutiveVowels(input); if (hasConsecutiveVowels) {
            System.out.println("Consecutive occurrence of vowels found.");
        } else {
            System.out.println("No consecutive occurrence of vowels found.");
        }
    }

    public static boolean checkConsecutiveVowels(String input) {
        for (int i = 0; i < input.length() - 1; i++) {
            char current = input.charAt(i);
            char next = input.charAt(i + 1);
            if (isVowel(current) && isVowel(next)) {
                return true;
            }
        }
        return false;
    }

    public static boolean isVowel(char c) {
        return "AEIOUaeiou".indexOf(c) != -1;
    }
}

Enter a string:
jaavaa
Consecutive occurrence of vowels found.

```

31. Write a Java Program to Find the Largest & Smallest Word in a String.

```
Code : import java.util.Scanner;
```

```

public class LargestSmallestWord {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine();

        String[] words = input.split("\\s+");
        String smallest = words[0];
        String largest = words[0];

        for (String word : words) {
            if (word.length() < smallest.length()) {
                smallest = word;
            }
            if (word.length() > largest.length()) {
                largest = word;
            }
        }

        System.out.println("Smallest word: " + smallest);
        System.out.println("Largest word: " + largest);
    }
}

Enter a string:
java is language.
Smallest word: is
Largest word: language.

```

32. Write a Java Program to Find First and Last Occurrence of Given Character in a String. **Code :** import java.util.Scanner;

```

public class FirstLastOccurrence {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine();
        System.out.println("Enter the character to search for:");
        char ch = scanner.next().charAt(0);

        int firstIndex = input.indexOf(ch);
        int lastIndex = input.lastIndexOf(ch);
    }
}

```

```

if (firstIndex == -1) {
    System.out.println("Character '" + ch + "' not found in the
string."); } else {
    System.out.println("First occurrence of '" + ch + "' at index: " + firstIndex);
    System.out.println("Last occurrence of '" + ch + "' at index: " + lastIndex);
}
}

Enter a string:
kolkata
Enter the character to search for:
k
First occurrence of 'k' at index: 0
Last occurrence of 'k' at index: 3

```

33. Write a Java Program to Display the Characters in Prime Position a Given String.

Code : import java.util.Scanner;

```

public class PrimePositionCharacters {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine();

        System.out.println("Characters in prime positions:");
        for (int i = 2; i < input.length(); i++) {
            if (isPrime(i)) {
                System.out.print(input.charAt(i) + " ");
            }
        }
    }

    public static boolean isPrime(int n) {
        if (n <= 1) {
            return false;
        }
        for (int i = 2; i * i <= n; i++) {
            if (n % i == 0) {
                return false;
            }
        }
    }
}

```

```

        return true;
    }
}

Enter a string:
kolkata
Characters in prime positions:
l k t

```

34. Write a Java Program to Sort String Ignoring Whitespaces and Repeating Characters Only Once.**Code :**

```

import java.util.Arrays;
import java.util.Scanner;
public class SortStringIgnoringSpaces {
    public static void main(String[] args) {
        Scanner scanner = new
        Scanner(System.in);
        System.out.println("Enter a string:");
        String
        input = scanner.nextLine();

        input = input.replaceAll("\\s", "");
        char[] chars = input.toCharArray();
        Arrays.sort(chars);

        StringBuilder result = new StringBuilder();
        result.append(chars[0]);
        for (int i = 1; i < chars.length; i++) {
            if (chars[i] != chars[i - 1]) {
                result.append(chars[i]);
            }
        }

        System.out.println("Sorted string ignoring whitespaces and repeating characters once:");
        System.out.println(result.toString());
    }
}

```

```

Enter a string:
UEM Kolkata
Sorted string ignoring whitespaces and repeating characters once:
EKMUaklot

```

35. Write a Java Program to Count Replace First Occurrence of a**String. Code :**

```

import java.util.Scanner;
public class CountReplaceFirstOccurrence {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the input string:");
        String input = scanner.nextLine();

```

```

System.out.println("Enter the string to replace:");
String toReplace = scanner.nextLine();
System.out.println("Enter the replacement string:");
String replacement = scanner.nextLine();

int count = 0;
int index = input.indexOf(toReplace);
if (index != -1) {
    count++;
    input = input.substring(0, index) + replacement + input.substring(index + toReplace.length());
}

System.out.println("String after replacing first occurrence:");
System.out.println(input);
System.out.println("Number of replacements made: " + count);
}
}

Enter the input string:
JAVA
Enter the string to replace:
AB
Enter the replacement string:
A
String after replacing first occurrence:
JAVA
Number of replacements made: 0

```

36. Write a Java Program to Know the Last Index of a Particular Word in a String. Code : import java.util.Scanner;

```

public class LastIndexOfWord {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine();
        System.out.println("Enter the word to search for:");
        String word = scanner.nextLine();

        int lastIndex = input.lastIndexOf(word);

        if (lastIndex != -1) {
            System.out.println("Last index of '" + word + "' in the string: " +
lastIndex); } else {
            System.out.println("Word '" + word + "' not found in the string.");
        }
    }
}

```

```

    }
}

Enter a string:
Kolkata
Enter the word to search for:
a
Last index of 'a' in the string: 6

```

37. Write a Java Program to Access the Index of the Character or String.

Code : import java.util.Scanner;

```

public class IndexOfCharacterString {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine();
        System.out.println("Enter the character/string to search for:");
        String search = scanner.nextLine();

        int index = input.indexOf(search);

        if (index != -1) {
            System.out.println("Index of '" + search + "' in the string: " +
index); } else {
            System.out.println("'" + search + "' not found in the string.");
        }
    }
}

```

```

Enter a string:
KOLKATA
Enter the character/string to search for:
A
Index of 'A' in the string: 4

```

38. Write a Java Program to Access the Characters or the ASCII of the Character Available in the String.

Code : import java.util.Scanner;

```

public class CharactersASCIIInString {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine();

        System.out.println("Characters and corresponding ASCII values:");

```

```

for (int i = 0; i < input.length(); i++) {
    char c = input.charAt(i);
    int asciiValue = (int) c;
    System.out.println("'" + c + "'": " + asciiValue);
}
}

Enter a string:
Kolkata
Characters and corresponding ASCII values:
'K': 75
'o': 111
'l': 108
'k': 107
'a': 97
't': 116
'a': 97

```

39. Write a Java Program to Display the Character and the Corresponding Ascii Present in the String.

Code : import java.util.Scanner;

```

public class CharacterASCII {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a character:");
        char c = scanner.next().charAt(0);

        int asciiValue = (int) c;
        System.out.println("ASCII value of '" + c + "'": " + asciiValue);
    }
}

Enter a character:
A
ASCII value of 'A': 65

```

40. Write a Java Program to Accept 2 String & Check Whether all Characters in First String is Present in Second String & Print.

Code : import java.util.Scanner;

```

public class CheckCharactersInSecondString {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

```

```

System.out.println("Enter the first string:");
String first = scanner.nextLine();
System.out.println("Enter the second string:");
String second = scanner.nextLine();

boolean allPresent = true;
for (char c : first.toCharArray()) {
    if (second.indexOf(c) == -1) {
        allPresent = false;
        break;
    }
}

if (allPresent) {
    System.out.println("All characters in the first string are present in the second
string."); } else {
    System.out.println("Not all characters in the first string are present in the second string.");
}
}
}

Enter the first string:
java
Enter the second string:
jAVA
All characters in the first string are present in the second string.

```

41. Write a Java Program to Check whether a Given Character is Present in a String, Find Frequency & Position of Occurrence.

Code : import java.util.Scanner;

```

public class CharacterOccurrence {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine();
        System.out.println("Enter the character to search for:");
        char target = scanner.next().charAt(0);

        int frequency = 0;
        for (int i = 0; i < input.length(); i++) {
            if (input.charAt(i) == target) {
                frequency++;
                System.out.println("Character " + target + " found at position: " + i);
            }
        }
    }
}

```

```

        }
        if (frequency == 0) {
            System.out.println("Character '" + target + "' not found in the string.");
        } else {
            System.out.println("Frequency of character '" + target + "' : " + frequency);
        }
    }
}

Enter a string:
KOLKATA
Enter the character to search for:
K
Character 'K' found at position: 0
Character 'K' found at position: 3
Frequency of character 'K': 2

```

42. Write a Java Program to Count the Number of Occurrence of Each Character Ignoring the Case of Alphabets & Display them.

```

Code : import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class CharacterFrequencyIgnoringCase {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine().toLowerCase();

        Map<Character, Integer> frequencyMap = new HashMap<>();
        for (char c : input.toCharArray()) {
            if (Character.isAlphabetic(c)) {
                frequencyMap.put(c, frequencyMap.getOrDefault(c, 0) + 1);
            }
        }

        System.out.println("Character frequencies (ignoring case):");
        for (Map.Entry<Character, Integer> entry : frequencyMap.entrySet()) {
            System.out.println("'" + entry.getKey() + "' : " + entry.getValue());
        }
    }
}

```

```
Enter a string:
java
Character frequencies (ignoring case):
'a': 2
've': 1
'j': 1
```

43. Write a Java Program to Give Shortest Sequence of Character Insertions and Deletions that Turn One String Into the Other.

Code : import java.util.Scanner;

```
public class ShortestSequence {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the first string:");
        String str1 = scanner.nextLine();
        System.out.println("Enter the second string:");
        String str2 = scanner.nextLine();

        int[][] dp = new int[str1.length() + 1][str2.length() + 1];
        for (int i = 0; i <= str1.length(); i++) {
            for (int j = 0; j <= str2.length(); j++) {
                if (i == 0 || j == 0)
                    dp[i][j] = i + j;
                else if (str1.charAt(i - 1) == str2.charAt(j - 1))
                    dp[i][j] = dp[i - 1][j - 1];
                else
                    dp[i][j] = 1 + Math.min(dp[i - 1][j], dp[i][j - 1]);
            }
        }

        int i = str1.length();
        int j = str2.length();
        StringBuilder sequence = new StringBuilder();
        while (i > 0 && j > 0) {
            if (str1.charAt(i - 1) == str2.charAt(j - 1)) {
                i--;
                j--;
            } else if (dp[i - 1][j] < dp[i][j - 1]) {
                sequence.append("Delete ").append(str1.charAt(i - 1)).append(", ");
                i--;
            } else {
                sequence.append("Insert ").append(str2.charAt(j - 1)).append(", ");
                j--;
            }
        }
    }
}
```

```

        j--;
    }
}
while (i > 0) {
    sequence.append("Delete ").append(str1.charAt(i - 1)).append(", ");
    i--;
}
while (j > 0) {
    sequence.append("Insert ").append(str2.charAt(j - 1)).append(", ");
    j--;
}

if (sequence.length() > 0) {
    System.out.println("Shortest sequence of insertions and
deletions:"); System.out.println(sequence.substring(0,
sequence.length() - 2));
} else {
    System.out.println("No sequence of insertions and deletions needed.");
}
}
}
Enter the first string:
UEM
Enter the second string:
UEMK
Shortest sequence of insertions and deletions:
Insert K

```

44. Write a Java Program to Check Whether Date is in Proper Format or**Not. Code :** import java.text.ParseException;

```

import java.text.SimpleDateFormat;
import java.util.Scanner;

```

```

public class DateValidation {

    public static void main(String[] args) { Scanner scanner = new
    Scanner(System.in); System.out.println("Enter a date (in
format dd-MM-yyyy):"); String dateStr = scanner.nextLine();

    SimpleDateFormat dateFormat = new SimpleDateFormat("dd-MM-yyyy");
    dateFormat.setLenient(false);
    try {
        dateFormat.parse(dateStr);
        System.out.println("Date is in proper format.");
    }
}

```

```

        } catch (ParseException e) {
            System.out.println("Date is not in proper format.");
        }
    }
}

Enter a date (in format dd-MM-yyyy):
18-03-2024
Date is in proper format.

```

45. Write a Java Program to Validate an Email Address Format.

Code :

```

import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class EmailValidation {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter an email address:");
        String email = scanner.nextLine();

        String regex = "^[a-zA-Z0-9_+&*-]+(?:\\.[a-zA-Z0-9_+&*-]+)*@(?:[a-zA-Z0-9-]+\\.)+[a-zA-Z]{2,7}$";
        Pattern pattern = Pattern.compile(regex);
        Matcher matcher = pattern.matcher(email);

        if (matcher.matches()) {
            System.out.println("Email address is valid.");
        } else {
            System.out.println("Email address is not valid.");
        }
    }
}

Enter an email address:
rohit@gmail.com
Email address is valid.

```

46. Write a Java Program to Store String Literals Using String Buffer.

Code :

```

public class StringBufferExample {
    public static void main(String[] args) {
        StringBuffer buffer = new
StringBuffer(); buffer.append("Hello");
buffer.append(" ");
buffer.append("World");

```

```
        System.out.println(buffer.toString());
    }
}
```

Hello World

47. Write a Java Program to Verify a Class is StringBuffer Class

Method. Code :

```
public class StringBufferCheck {
    public static void main(String[] args) {
        String str = "Hello";
        boolean isStringBuffer = str.getClass().equals(StringBuffer.class);
        System.out.println("Is 'str' an instance of StringBuffer? " + isStringBuffer);
    }
}
```

Is 'str' an instance of StringBuffer? false

48. Write a Java Program to Ask the User His Name and Greets Him With His Name.

Code :

```
import java.util.Scanner;

public class GreetWithName {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter your name:");
        String name = scanner.nextLine();

        System.out.println("Hello, " + name + "! Nice to meet you.");
    }
}
```

Enter your name:

rohit

Hello, rohit! Nice to meet you.

PS D:\RN_1\Coding\JAVA\WEEK8\src>

49. Write a Java Program to Count a Group of Words in a String.

Code :

```
import java.util.Scanner;

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

```

Scanner scanner = new Scanner(System.in);
System.out.println("Enter a string:");
String input = scanner.nextLine();
System.out.println("Enter the word or group of words to count:");
String wordGroup = scanner.nextLine();

int count = countWordGroupOccurrences(input, wordGroup);
System.out.println("Number of occurrences of the word/group: " + count);
}

public static int countWordGroupOccurrences(String input, String wordGroup)
{ int count = 0;
int index = input.indexOf(wordGroup);
while (index != -1) {
    count++;
    index = input.indexOf(wordGroup, index + 1);
}
return count;
}
}

Enter a string:
java
Enter the word or group of words to count:
a
Number of occurrences of the word/group: 2

```

50. Write a Java Program to Count Number of Words in a given Text or Sentence. **Code :** import java.util.Scanner;

```

public class WordCount {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a text or sentence:");
        String text = scanner.nextLine();

        int wordCount = countWords(text);
        System.out.println("Number of words: " + wordCount);
    }

    public static int countWords(String text) {
        String[] words = text.split("\\s+");
        return words.length;
    }
}

```

}

Enter a text or sentence:

my name is java

Number of words: 4

