



Graphic Era

HILL UNIVERSITY

University under section 2(f) of UGC Act, 1956

Bhimtal Campus

INDEX PAGE

S.no.	Question	Date	Sign
01	Write a java program to print "Hello world"		
02	Write a java program to add two numbers		
03	Write a java program to add two numbers by taking input from user		
04	Write a java program to print the difference of two numbers		
05	Write a java program to print the difference of two numbers by taking input from user		
06	Write a java program to print a pattern: * ** *** **** *****		
07	Write a java program to check whether the given number is even or odd		
08	Write a java program to print the string by taking input from user		
09	Write a java program to demonstrate switch case statement: Case 1: "Graphic Era Bhimtal" Case 2: "Graphic Era Haldwani" Case 3: "Graphic Era Dehradun" Case 4: "Graphic Era Deemed To Be University" default: "GEHU"		
10	Write a java program to print two statements in different lines by line changing method		
11	Write a java program to print "hello world" 100 times by using for loop		
12	Write a java program to print natural numbers up to 10 (i) in horizontal form, (ii) in vertical form by using for loop: Eg: 1 2 3 4 Eg: 1 2 3..		
13	Write a java program to print table of a number by taking input from user using do while loop		
14	Write a java program to print natural numbers up to 11 by using while loop		
15	Write a java program to print the pattern, using nested loop: **** **** **** ****		
16	Write a java program to take input as a command line argument. Your name, course, university roll no and semester. Display the information. Name: University Roll No: Course: Semester:		
17	Using the switch statement, write a menu-driven program to calculate the maturity amount of a bank deposit. The user is given the following options: (i) Term Deposit		

	<p>(ii) Recurring Deposit For option (i) accept Principal (p), rate of interest (r) and time period in years (n). Calculate and output the maturity amount (a) receivable using the formula $a = p[1 + r / 100]n$.</p> <p>For option (ii) accept monthly installment (p), rate of interest (r) and time period in months (n). Calculate and output the maturity amount (a) receivable using the formula $a = p * n + p * n(n + 1) / 2 * r / 100 * 1 / 12$. For an incorrect option, an appropriate error message should be displayed.</p> <p>[Use Scanner Class to take input]</p>		
18	<p>Program to find if the given numbers are Friendly pair or not (Amicable or not). Friendly Pair are two or more numbers with a common abundance.</p> <p>Input & Output format:</p> <ul style="list-style-type: none"> Input consists of 2 integers. The first integer corresponds to number 1 and the second integer corresponds to number 2. <p>If it is a Friendly Pair display Friendly Pair or displays Not Friendly Pair.</p> <p>For example, 6 and 28 are Friendly Pair. (Sum of divisors of 6)/6 = (Sum of divisors of 28)/28.</p> <p>Steps to check whether the given numbers are friendly pair or not</p> <ul style="list-style-type: none"> Input the numbers num1 and num2. Initialize sum1 = sum2 = 0. sum1 = sum of all divisors of num1. sum2 = sum of all divisors of num2. If (sum1 == num1) and (sum2 == num2), then print "Abundant Numbers". Else, print "Not Abundant Numbers". 		
19	<p>Program to replace all 0's with 1 in a given integer. Given an integer as an input, all the 0's in the number has to be replaced with 1. For example, consider the following number Input: 102405 Output: 112415 Input: 56004 Output: 56114</p> <p>Steps to replace all 0's with 1 in a given integer</p> <ul style="list-style-type: none"> Input the integer from the user. Traverse the integer digit by digit. If a '0' is encountered, replace it by '1'. Print the integer. 		
20	<p>Printing an array into Zigzag fashion. Suppose you were given an array of integers, and you are told to sort the integers in a zigzag pattern. In general, in a zigzag pattern, the first integer is less than the second integer, which is greater than the third integer, which is less than the fourth integer, and so on. Hence, the converted array should be in the</p>		

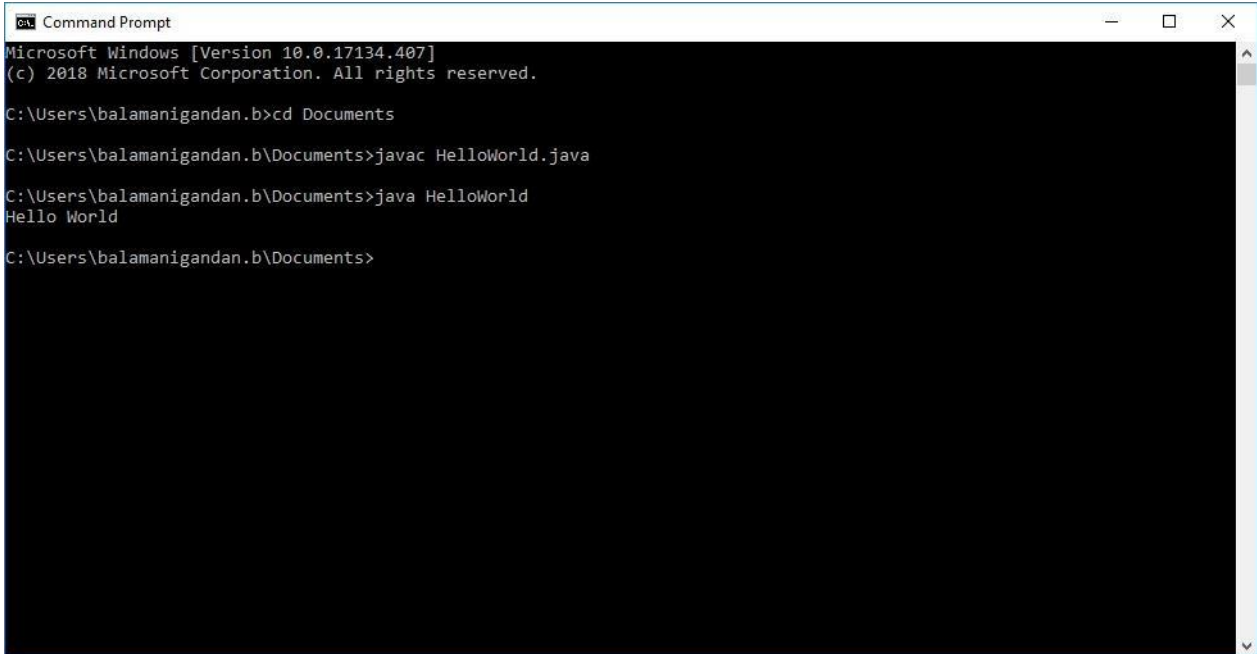
	<p>form of $e1 < e2$</p> <p>$> e3$</p> <p>$< e4$</p> <p>$> e5$</p> <p>$< e6$.</p> <p>Test cases:</p> <p>Input</p> <p>1:</p> <p>4 3 7 8 6 2 1</p> <p>Output 1:</p> <p>3 7 4 8 2 6 1</p> <p>Input 2:</p> <p>4</p> <p>1 4 3 2</p> <p>Output 2: 1 4 2 3</p>		
21	<p>The problem to rearrange positive and negative numbers in an array .</p> <p>Method: This approach moves all negative numbers to the beginning and positive numbers to the end but changes the order of appearance of the elements of the array.</p> <p>Steps:</p> <ol style="list-style-type: none"> 1. Declare an array and input the array elements. 2. Start traversing the array and if the current element is negative, swap the current element with the first positive element and continue traversing until all the elements have been encountered. 3. Print the rearranged array. <p>Test case:</p> <ul style="list-style-type: none"> • Input: 1 -1 2 -2 3 -3 • Output: -1 -2 -3 1 3 2 		

22	<p>Program to find the saddle point coordinates in a given matrix. A saddle point is an element of the matrix, which is the minimum element in its row and the maximum in its column.</p> <p>For example, consider the matrix given below</p> <pre>Mat [3][3] 1 2 3 4 5 6 7 8 9</pre> <p>Here, 7 is the saddle point because it is the minimum element in its row and maximum element in its column.</p> <p>Steps to find the saddle point coordinates in a given matrix.</p> <ol style="list-style-type: none"> 1. Input the matrix from the user. 2. Use two loops, one for traversing the row and the other for traversing the column. 3. If the current element is the minimum element in its row and maximum element in its column, then return its coordinates. <p>Else, continue traversing.</p>		
----	--	--	--

Q1. Write a java program to print “Hello world”.

```
Import
java.util.*; class
HelloWorld{
    public static void main(String args[])
    {
        System.out.println(“Hello World”);
    }
}
```

Output: Hello World

A screenshot of a Windows Command Prompt window. The title bar reads "Command Prompt". The window content shows the following text:
Microsoft Windows [Version 10.0.17134.407]
(c) 2018 Microsoft Corporation. All rights reserved.
C:\Users\balamanigandan.b>cd Documents
C:\Users\balamanigandan.b\Documents>javac HelloWorld.java
C:\Users\balamanigandan.b\Documents>java HelloWorld
Hello World
C:\Users\balamanigandan.b\Documents>
The window has a standard Windows interface with a title bar, maximize, minimize, and close buttons, and a scroll bar on the right side.

Q2. Write a java program to add two numbers

```
import java.util.Scanner;

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

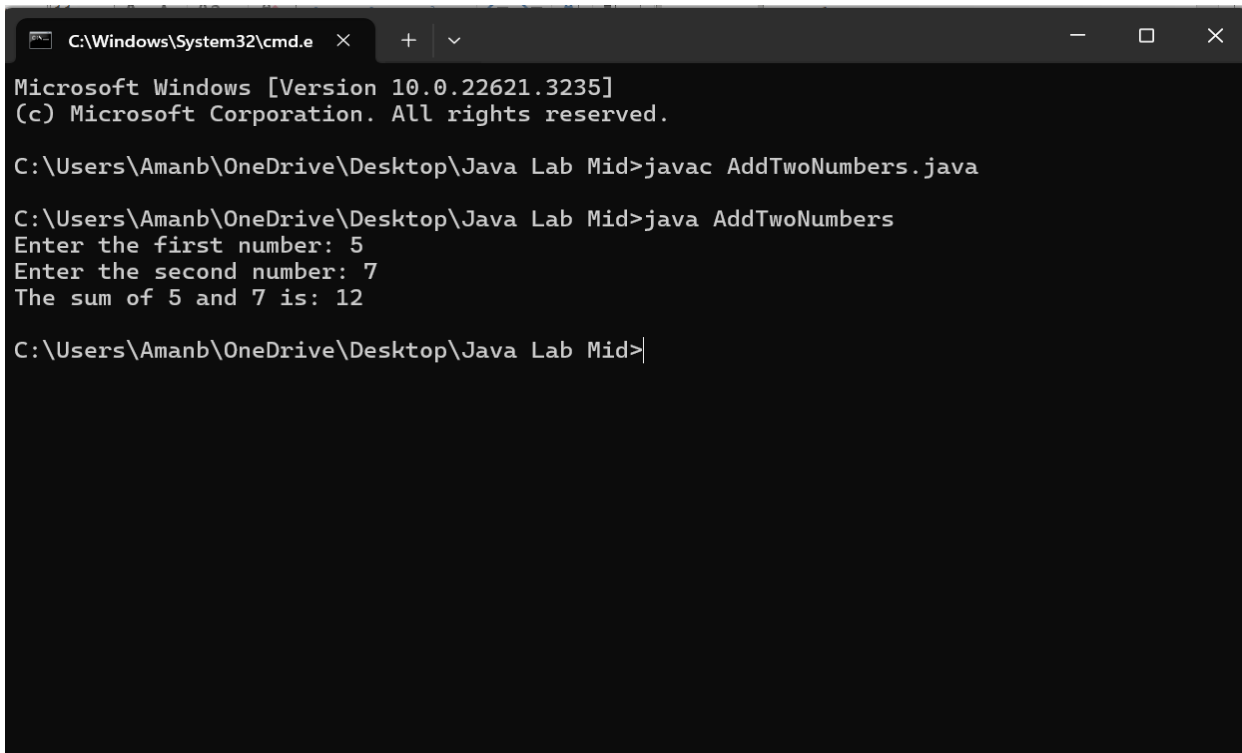
        System.out.print("Enter the first number: ");
        int num1 = scanner.nextInt();

        System.out.print("Enter the second number: ");
        int num2 = scanner.nextInt();

        int sum = num1 + num2;

        System.out.println("The sum of " + num1 + " and " + num2 + " is: " + sum);
    }
}
```

Output: Enter the first number: 5
Enter the second number: 7
The sum of 5 and 7 is: 12

A screenshot of a Windows Command Prompt window. The title bar shows the path 'C:\Windows\System32\cmd.e' and standard window controls. The window content displays the following text:
Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>javac AddTwoNumbers.java

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java AddTwoNumbers
Enter the first number: 5
Enter the second number: 7
The sum of 5 and 7 is: 12

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>|

Q3. Write a java program to add two numbers by taking input from user

```
import java.util.Scanner;

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

        System.out.print("Enter the first number: ");
        double num1 = scanner.nextDouble();

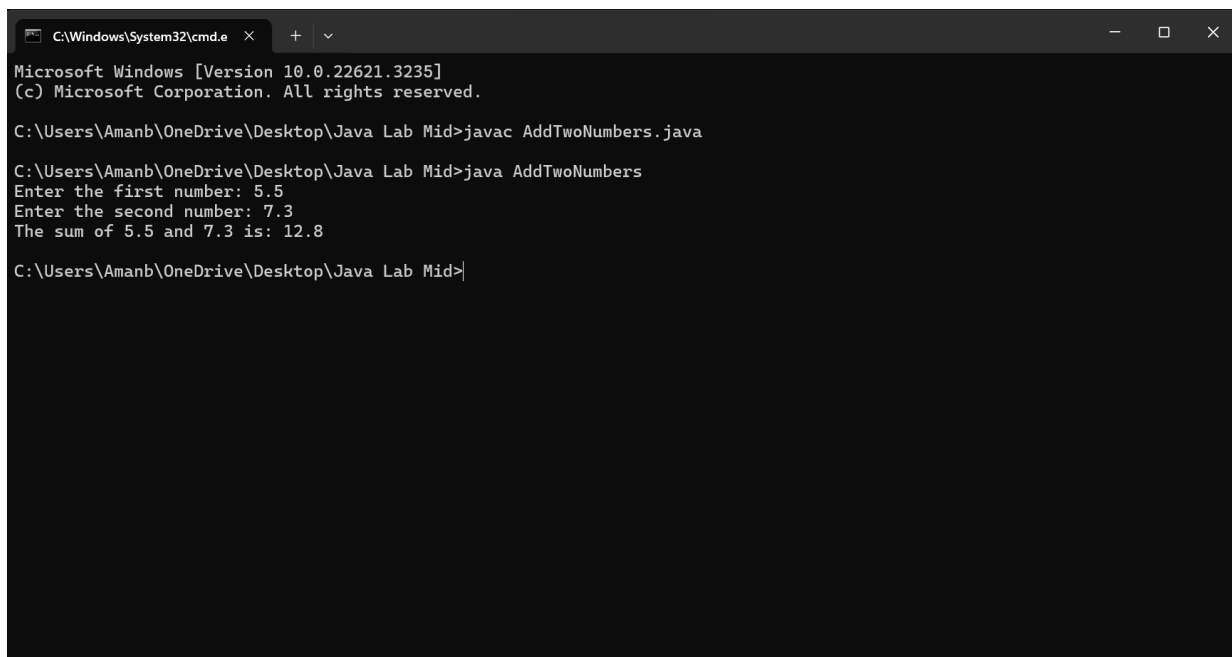
        System.out.print("Enter the second number: ");
        double num2 = scanner.nextDouble();

        double sum = num1 + num2;

        System.out.println("The sum of " + num1 + " and " + num2 + " is: " + sum);

        scanner.close();
    }
}
```

Output: Enter the first number: 5.5
Enter the second number: 7.3
The sum of 5.5 and 7.3 is: 12.8

A screenshot of a Windows Command Prompt window. The title bar shows 'C:\Windows\System32\cmd.e' with standard window controls. The command prompt displays the following text: 'Microsoft Windows [Version 10.0.22621.3235] (c) Microsoft Corporation. All rights reserved. C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>javac AddTwoNumbers.java C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java AddTwoNumbers Enter the first number: 5.5 Enter the second number: 7.3 The sum of 5.5 and 7.3 is: 12.8 C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>'. The background is black and the text is white.

Q4. Write a java program to print the difference of two numbers

```
import java.util.Scanner;

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

        System.out.print("Enter the first number: ");
        double num1 = scanner.nextDouble();

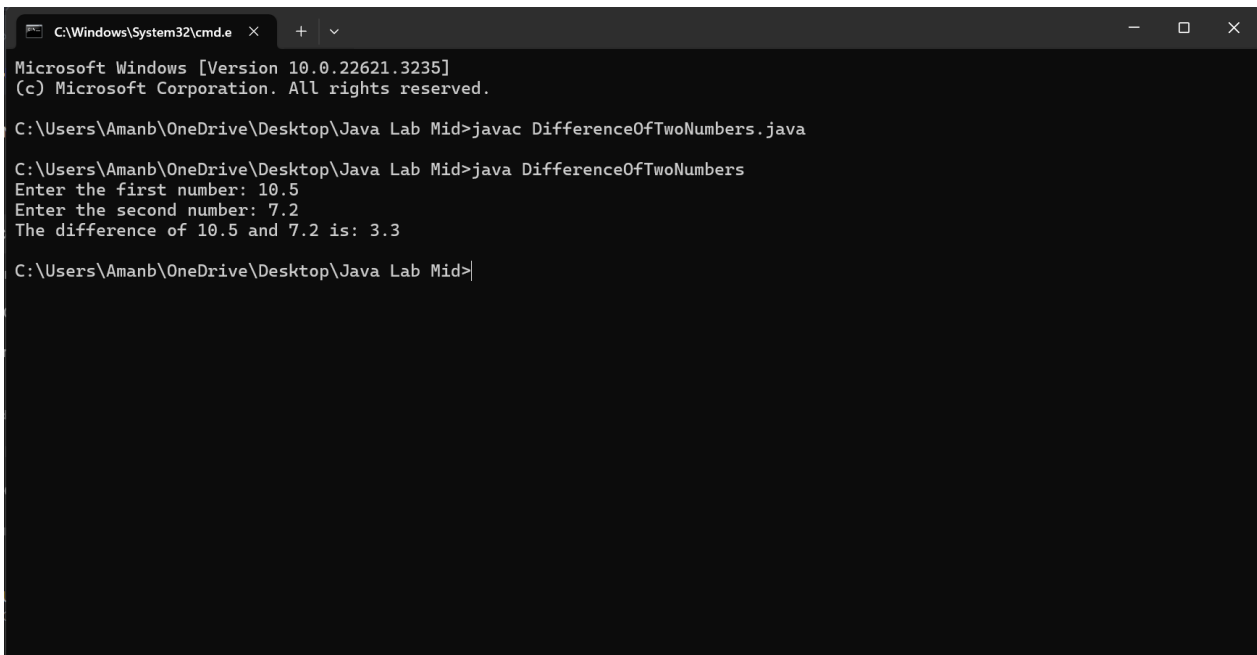
        System.out.print("Enter the second number: ");
        double num2 = scanner.nextDouble();

        double difference = num1 - num2;

        System.out.println("The difference of " + num1 + " and " + num2 + " is: " + difference);

        scanner.close();
    }
}
```

Output: Enter the first number: 10.5
Enter the second number: 7.2
The difference of 10.5 and 7.2 is: 3.3

A screenshot of a Windows Command Prompt window. The title bar shows 'C:\Windows\System32\cmd.e' with standard window controls. The window content displays the following text:
Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>javac DifferenceOfTwoNumbers.java
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java DifferenceOfTwoNumbers
Enter the first number: 10.5
Enter the second number: 7.2
The difference of 10.5 and 7.2 is: 3.3
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>|

Q5. Write a java program to print the difference of two numbers by taking input from user

```
import java.util.Scanner;

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

        System.out.print("Enter the first number: ");
        double num1 = scanner.nextDouble();

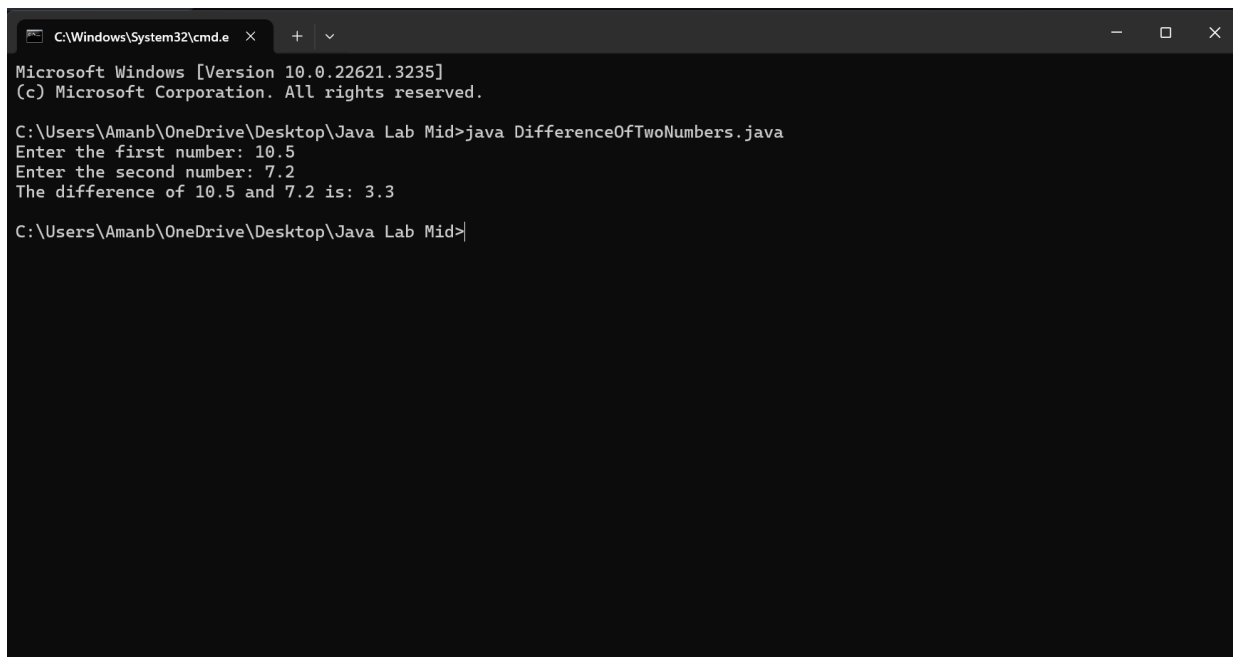
        System.out.print("Enter the second number: ");
        double num2 = scanner.nextDouble();

        double difference = num1 - num2;

        System.out.println("The difference of " + num1 + " and " + num2 + " is: " + difference);

        scanner.close();
    }
}
```

Output: Enter the first number: 10.5
Enter the second number: 7.2
The difference of 10.5 and 7.2 is: 3.3

A screenshot of a Windows Command Prompt window. The title bar shows the path 'C:\Windows\System32\cmd.exe'. The window content displays the output of running a Java program. It starts with the Windows version and copyright information. Then, the command 'C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java DifferenceOfTwoNumbers.java' is entered. The program prompts for the first number (10.5) and the second number (7.2), and finally outputs 'The difference of 10.5 and 7.2 is: 3.3'. The prompt returns to 'C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>'.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java DifferenceOfTwoNumbers.java
Enter the first number: 10.5
Enter the second number: 7.2
The difference of 10.5 and 7.2 is: 3.3

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>
```

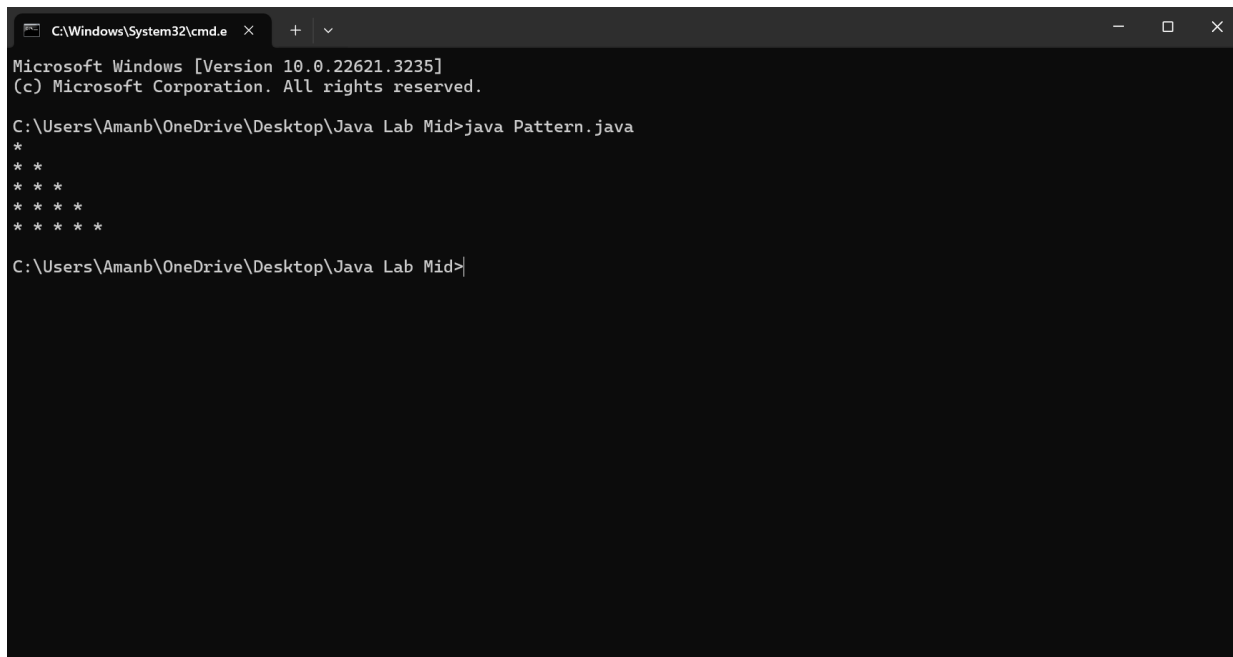
Q6. Write a java program to print a pattern:

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

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

Output: *

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



The screenshot shows a Windows command prompt window with the title bar 'C:\Windows\System32\cmd.e'. The window content displays the following text:

```
Microsoft Windows [Version 10.0.22621.3235]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java Pattern.java  
*  
* *  
* * *  
* * * *  
* * * * *  
  
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>
```

Q7. Write a java program to check whether the given number is even or odd

```
import java.util.Scanner;

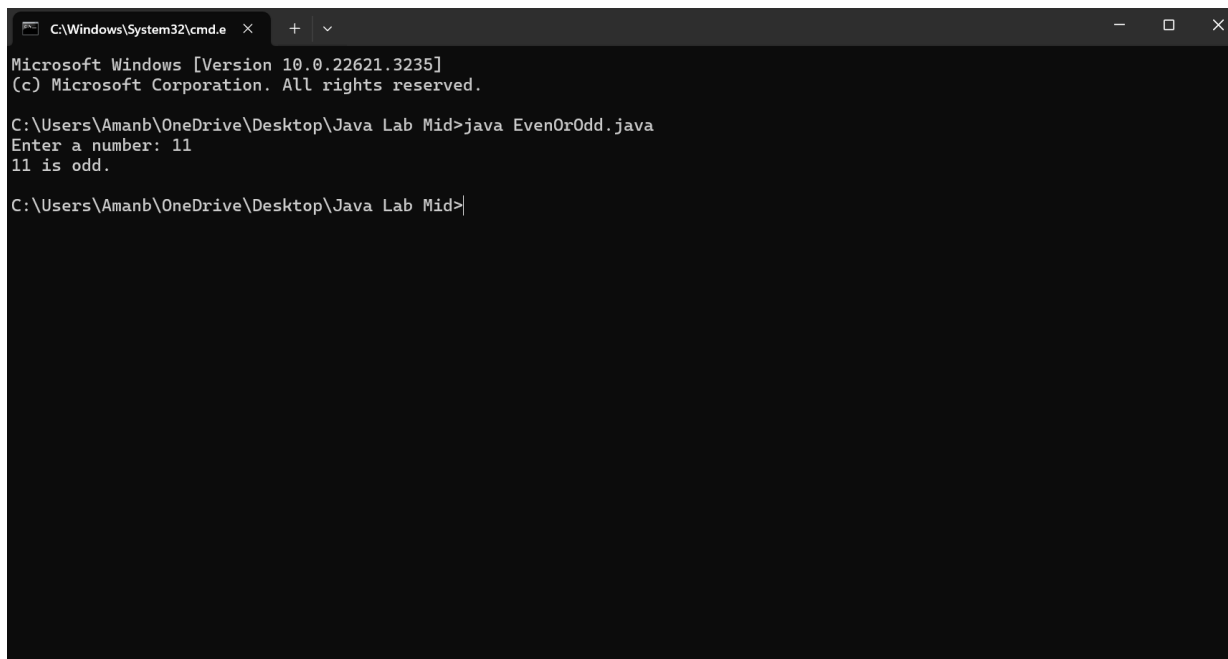
public class EvenOrOdd {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a number: ");
        int number = scanner.nextInt();

        if (number % 2 == 0) {
            System.out.println(number + " is even.");
        } else {
            System.out.println(number + " is odd.");
        }

        scanner.close();
    }
}
```

Output: Enter a number: 6
6 is even.



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java EvenOrOdd.java
Enter a number: 11
11 is odd.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>
```

Q8. Write a java program to print the string by taking input from user

```
import java.util.Scanner;

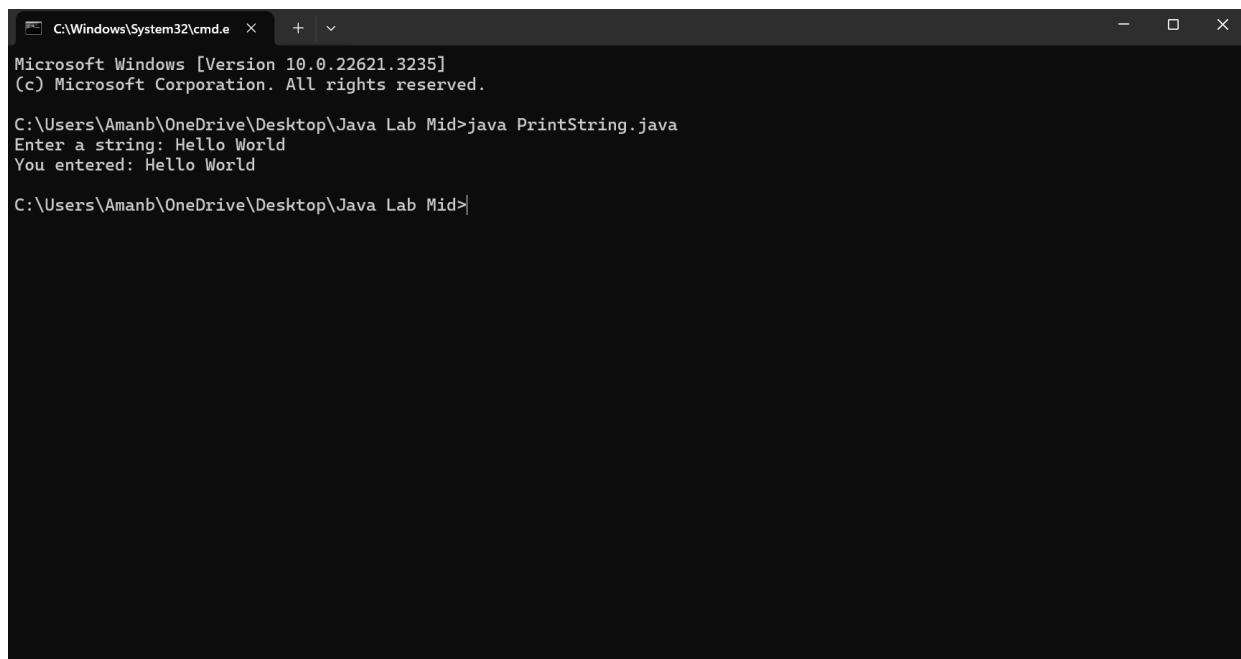
public class PrintString {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");
        String inputString = scanner.nextLine();

        System.out.println("You entered: " + inputString);

        scanner.close();
    }
}
```

Output: Enter a string: Hello World
You entered: Hello World

A screenshot of a Windows Command Prompt window. The title bar shows 'C:\Windows\System32\cmd.e' with standard window controls. The window content displays the following text: 'Microsoft Windows [Version 10.0.22621.3235] (c) Microsoft Corporation. All rights reserved.' followed by the command 'C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java PrintString.java'. The output of the program is shown as 'Enter a string: Hello World' and 'You entered: Hello World'. The prompt 'C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>' is visible at the bottom.

Q9. Write a java program to demonstrate switch case statement:

Case 1: "Graphic Era Bhimtal"

Case 2: "Graphic Era Haldwani"

Case 3: "Graphic Era Dehradun"

Case 4: "Graphic Era Deemed To Be University"

default: "GEHU"

```
import java.util.Scanner;

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

        System.out.println("Enter your choice (1-4):");
        int choice = scanner.nextInt();

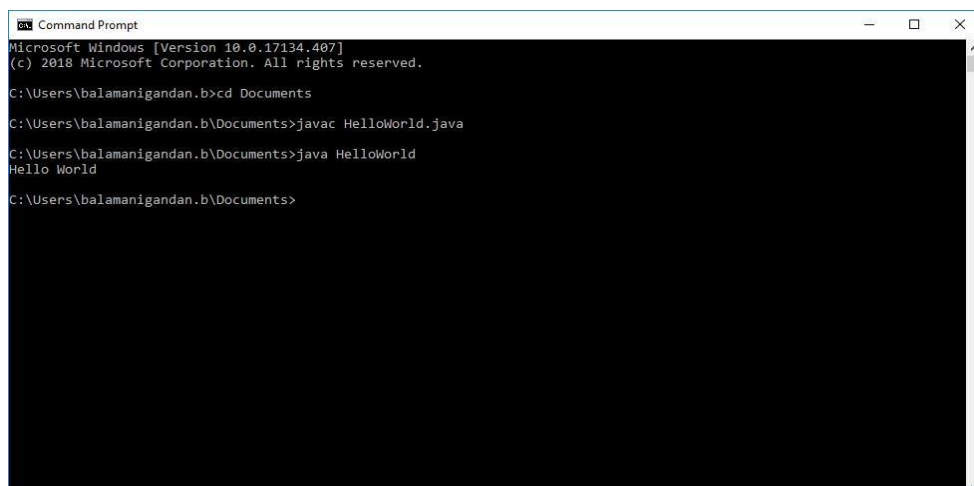
        switch (choice) {
            case 1:
                System.out.println("Graphic Era Bhimtal");
                break;
            case 2:
                System.out.println("Graphic Era Haldwani");
                break;
            case 3:
                System.out.println("Graphic Era Dehradun");
                break;
            case 4:
                System.out.println("Graphic Era Deemed To Be University");
                break;
            default:
                System.out.println("GEHU");
        }

        scanner.close();
    }
}
```

Output: Enter your choice (1-4):

2

Graphic Era Haldwani



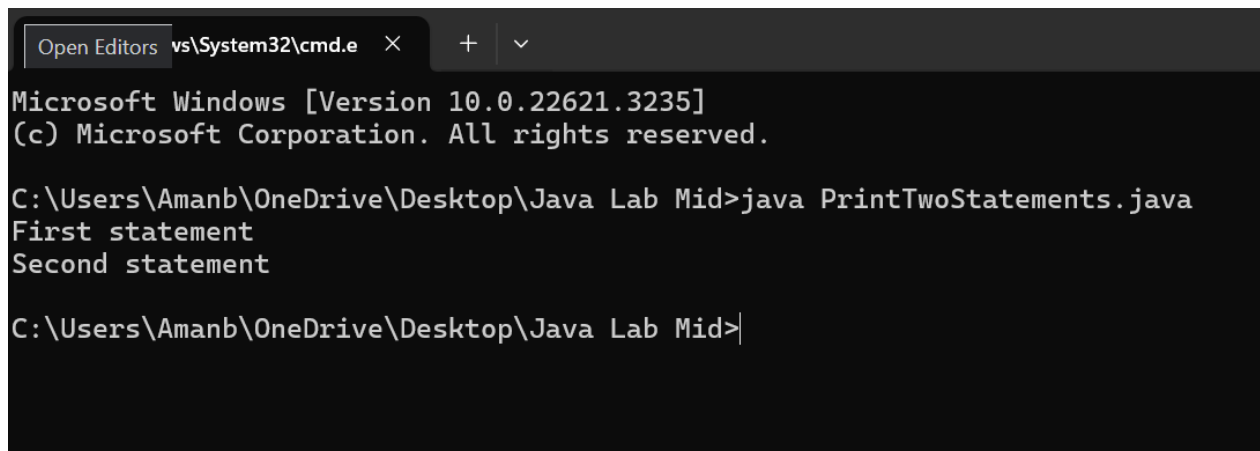
```
Microsoft Windows [Version 10.0.17134.407]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\balamanigandan.b>cd Documents
C:\Users\balamanigandan.b\Documents>javac HelloWorld.java
C:\Users\balamanigandan.b\Documents>java HelloWorld
Hello World
C:\Users\balamanigandan.b\Documents>
```

Q10. Write a java program to print two statements in different lines by line changing method

```
public class PrintTwoStatements {  
    public static void main(String[] args) {  
        System.out.println("First statement");  
        System.out.println("Second statement");  
    }  
}
```

Output: First statement
Second statement Hello World



The screenshot shows a Windows command prompt window with the title bar 'Open Editors vs\System32\cmd.e'. The window displays the following text:

```
Microsoft Windows [Version 10.0.22621.3235]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java PrintTwoStatements.java  
First statement  
Second statement  
  
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>|
```

Q11. Write a java program to print “hello world” 100 times by using for loop

```
public class HelloWorld {
    public static void main(String[] args) {
        for (int i = 0; i < 100; i++) {
            System.out.println("hello world");
        }
    }
}
```

Output: hello world
hello world
hello world...

[illegible]

Q12. Write a java program to print natural numbers up to 10

(i) in horizontal form,

(ii) in vertical form by using for loop:

Eg: 1 2 3 4

Eg: 1

2

3..

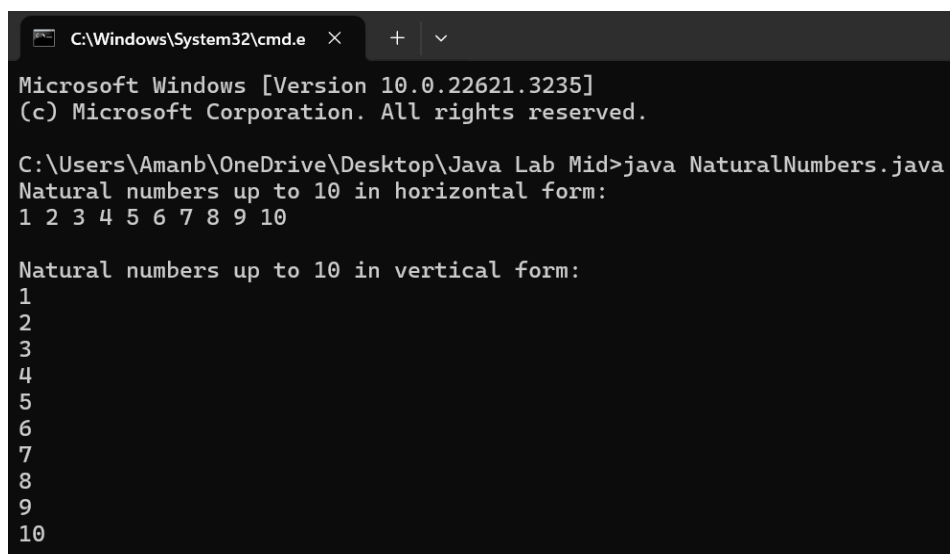
```
public class NaturalNumbers {  
    public static void main(String[] args) {  
        System.out.println("Natural numbers up to 10 in horizontal form:");  
        for (int i = 1; i <= 10; i++) {  
            System.out.print(i + " ");  
        }  
        System.out.println();  
  
        System.out.println("\nNatural numbers up to 10 in vertical form:");  
        for (int i = 1; i <= 10; i++) {  
            System.out.println(i);  
        }  
    }  
}
```

Output: Natural numbers up to 10 in horizontal form:

1 2 3 4 5 6 7 8 9 10

Natural numbers up to 10 in vertical form:

1
2
3
4
5
6
7
8
9
10



```
C:\Windows\System32\cmd.e  X  +  v  
Microsoft Windows [Version 10.0.22621.3235]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java NaturalNumbers.java  
Natural numbers up to 10 in horizontal form:  
1 2 3 4 5 6 7 8 9 10  
  
Natural numbers up to 10 in vertical form:  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```


Q13. Write a java program to print table of a number by taking input from user using do while loop

```
import java.util.Scanner;

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

        int number;
        do {
            System.out.print("Enter a number to print its table: ");
            number = scanner.nextInt();

            if (number <= 0) {
                System.out.println("Please enter a positive number.");
            }
        } while (number <= 0);

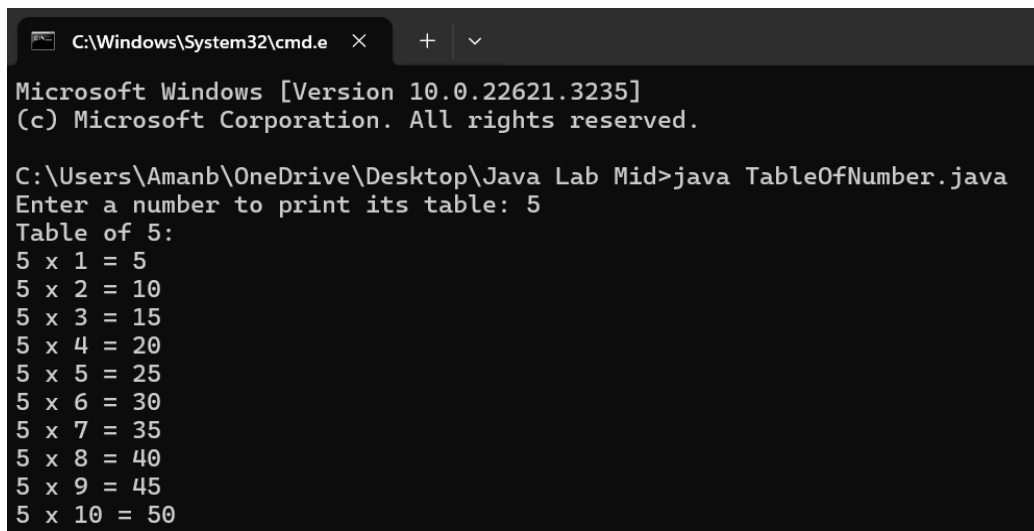
        System.out.println("Table of " + number + ":");
        int i = 1;
        do {
            System.out.println(number + " x " + i + " = " + (number * i));
            i++;
        } while (i <= 10);

        scanner.close();
    }
}
```

Output: Enter a number to print its table: 5

Table of 5:

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



The screenshot shows a Windows command prompt window with the title bar 'C:\Windows\System32\cmd.e'. The window content displays the following text:

```
Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

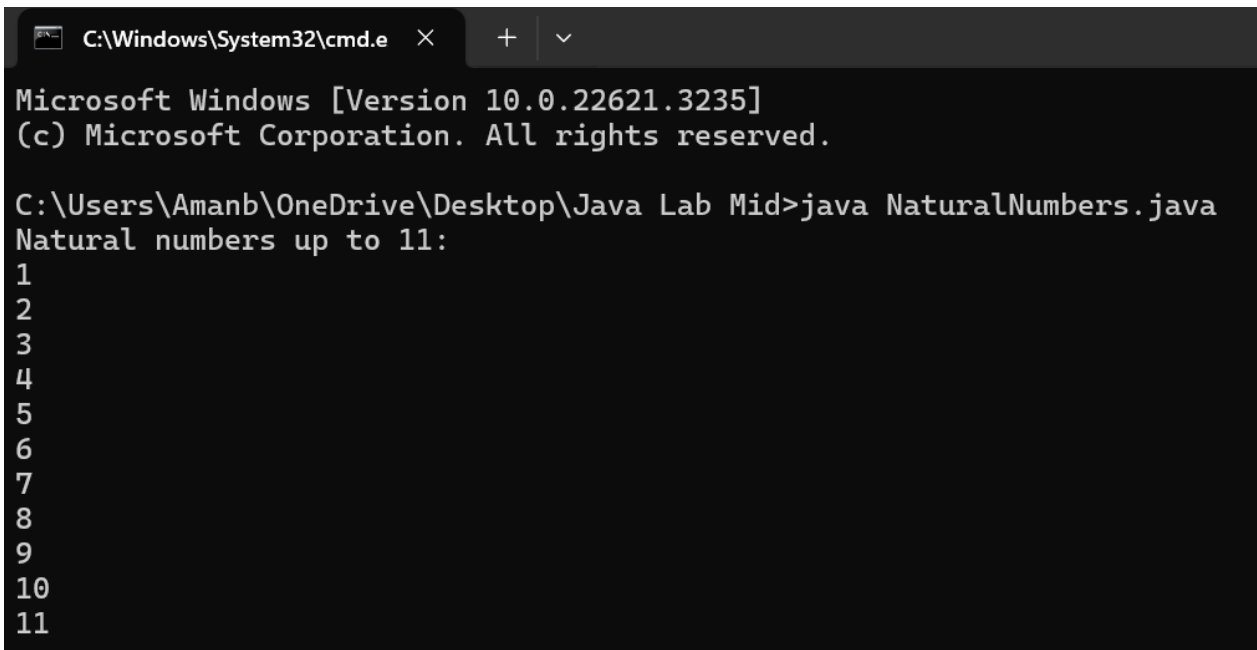
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java TableOfNumber.java
Enter a number to print its table: 5
Table of 5:
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
```

Q14. Write a java program to print natural numbers up to 11 by using while loop

```
public class NaturalNumbers {  
    public static void main(String[] args) {  
        int i = 1;  
        System.out.println("Natural numbers up to 11:");  
        while (i <= 11) {  
            System.out.println(i);  
            i++;  
        }  
    }  
}
```

Output: Natural numbers up to 11:

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11
```



The screenshot shows a Windows Command Prompt window with the title bar 'C:\Windows\System32\cmd.e'. The window content displays the following text:

```
Microsoft Windows [Version 10.0.22621.3235]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java NaturalNumbers.java  
Natural numbers up to 11:  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11
```

Q15. Write a java program to print the pattern, using nested loop:

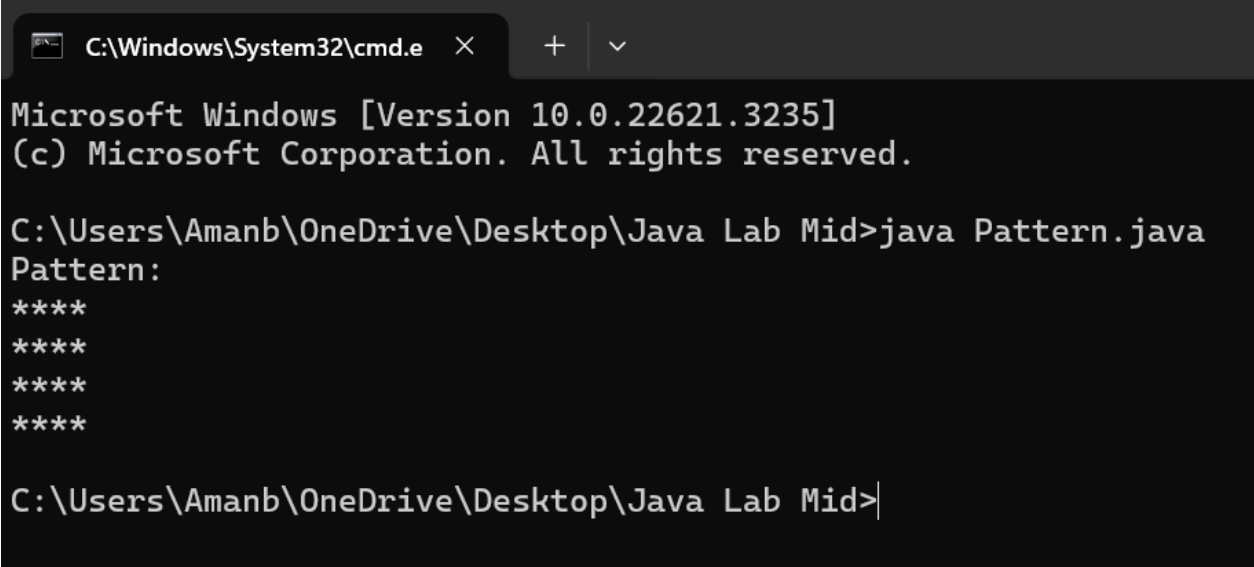
```
****
****
****
****
```

```
public class Pattern {
    public static void main(String[] args) {
        int rows = 4;
        int cols = 4;

        System.out.println("Pattern:");
        for (int i = 1; i <= rows; i++) {
            for (int j = 1; j <= cols; j++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

Output: Pattern:

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

A screenshot of a Windows Command Prompt window. The title bar shows 'C:\Windows\System32\cmd.e' with a close button. The window content displays the Microsoft Windows version (10.0.22621.3235) and copyright information. The user is at the directory 'C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid' and has executed the command 'java Pattern.java'. The output shows the word 'Pattern:' followed by a 4x4 grid of asterisks. The prompt is ready for the next command.

```
C:\Windows\System32\cmd.e  X  +  v

Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java Pattern.java
Pattern:
****
****
****
****

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>
```

Q16. Write a java program to take input as a command line argument.
Your name,course, university roll no and semester. Display the information.

Name: University Roll No: Course: Semester:

```
import java.util.Scanner;

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

        System.out.print("Enter your name: ");
        String name = scanner.nextLine();

        System.out.print("Enter your university roll number: ");
        String rollNo = scanner.nextLine();

        System.out.print("Enter your course: ");
        String course = scanner.nextLine();

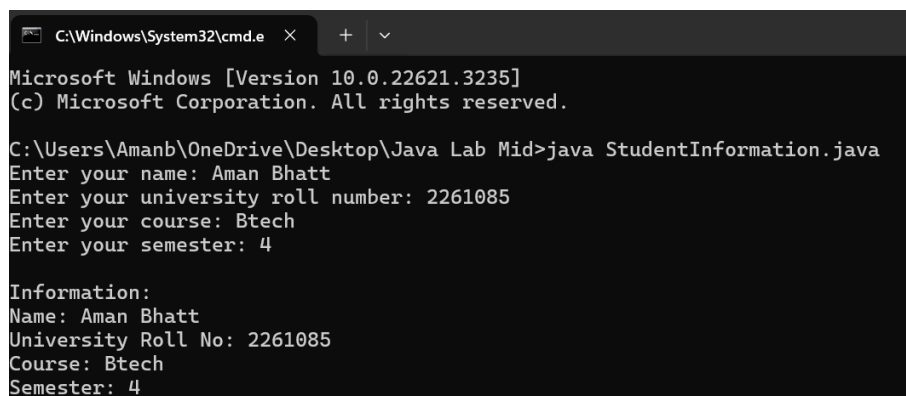
        System.out.print("Enter your semester: ");
        String semester = scanner.nextLine();

        System.out.println("\nInformation:");
        System.out.println("Name: " + name);
        System.out.println("University Roll No: " + rollNo);
        System.out.println("Course: " + course);
        System.out.println("Semester: " + semester);

        scanner.close();
    }
}
```

Output: Enter your name: Aman Bhatt
Enter your university roll number: 2261085
Enter your course: Btech
Enter your semester: 4

Information:
Name: Aman Bhatt
University Roll No: 2261085
Course: Btech
Semester: 4



The screenshot shows a Windows command prompt window with the title bar 'C:\Windows\System32\cmd.e'. The text inside the window is as follows:

```
Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java StudentInformation.java
Enter your name: Aman Bhatt
Enter your university roll number: 2261085
Enter your course: Btech
Enter your semester: 4

Information:
Name: Aman Bhatt
University Roll No: 2261085
Course: Btech
Semester: 4
```

Q17. Using the switch statement, write a menu-driven program to calculate the maturity amount of a bank deposit.

The user is given the following options:

(i) Term Deposit

(ii) Recurring Deposit

For option (i) accept Principal (p), rate of interest (r) and time period in years (n). Calculate and output the maturity amount (a) receivable using the formula $a = p[1 + r / 100]n$.

For option (ii) accept monthly installment (p), rate of interest (r) and time period in months (n). Calculate and output the maturity amount (a) receivable using the formula $a = p * n + p * n(n + 1) / 2 * r / 100 * 1 / 12$. For an incorrect option, an appropriate error message should be displayed.

[Use Scanner Class to take input]

```
import java.util.Scanner;

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

        System.out.println("Menu:");
        System.out.println("1. Term Deposit");
        System.out.println("2. Recurring Deposit");
        System.out.print("Enter your choice (1 or 2): ");
        int choice = scanner.nextInt();

        switch (choice) {
            case 1:
                System.out.print("Enter Principal amount: ");
                double principalTerm = scanner.nextDouble();
                System.out.print("Enter Rate of Interest (in %): ");
                double rateTerm = scanner.nextDouble();
                System.out.print("Enter Time period (in years): ");
                double timeTerm = scanner.nextDouble();

                double maturityTerm = principalTerm * Math.pow(1 + (rateTerm / 100), timeTerm);
                System.out.println("Maturity amount for Term Deposit: " + maturityTerm);
                break;

            case 2:
                System.out.print("Enter Monthly Installment amount: ");
                double installment = scanner.nextDouble();
                System.out.print("Enter Rate of Interest (in %): ");
                double rateRecurring = scanner.nextDouble();
                System.out.print("Enter Time period (in months): ");
                double timeRecurring = scanner.nextDouble();

                double maturityRecurring = installment * timeRecurring + installment * timeRecurring *
                (timeRecurring + 1) / 2 * rateRecurring / 100 * 1 / 12;
                System.out.println("Maturity amount for Recurring Deposit: " + maturityRecurring);
                break;

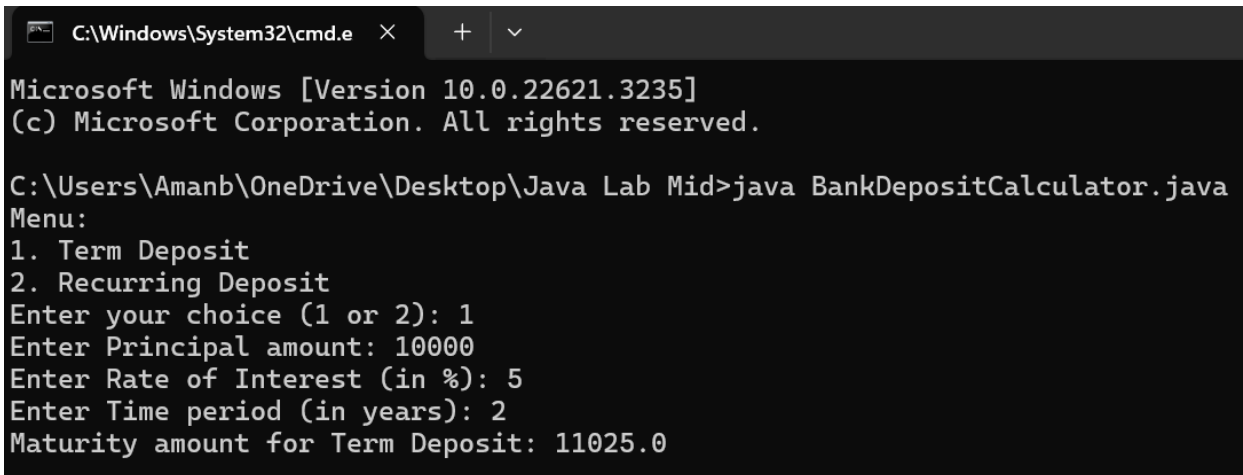
            default:
```

```
        System.out.println("Invalid choice! Please select either 1 or 2.");
    }

    scanner.close();
}
}
```

Output: Menu:

```
1. Term Deposit
2. Recurring Deposit
Enter your choice (1 or 2): 1
Enter Principal amount: 10000
Enter Rate of Interest (in %): 5
Enter Time period (in years): 2
Maturity amount for Term Deposit: 11025.0
```



```
C:\Windows\System32\cmd.e  ×  +  v

Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java BankDepositCalculator.java
Menu:
1. Term Deposit
2. Recurring Deposit
Enter your choice (1 or 2): 1
Enter Principal amount: 10000
Enter Rate of Interest (in %): 5
Enter Time period (in years): 2
Maturity amount for Term Deposit: 11025.0
```

Q18. Program to find if the given numbers are Friendly pair or not (Amicable or not). Friendly Pair are two or more numbers with a common abundance.

Input & Output format:

- Input consists of 2 integers.
- The first integer corresponds to number 1 and the second integer corresponds to number 2.

If it is a Friendly Pair display Friendly Pair or displays Not Friendly Pair.

For example, 6 and 28 are Friendly Pair.

(Sum of divisors of 6)/6 = (Sum of divisors of 28)/28.

Steps to check whether the given numbers are friendly pair or not

- Input the numbers num1 and num2.
- Initialize sum1 = sum2 = 0.
- sum1 = sum of all divisors of num1.
- sum2 = sum of all divisors of num2.
- If (sum1 == num1) and (sum2 == num2), then print "Abundant Numbers".
- Else, print "Not Abundant Numbers".

```
import java.util.Scanner;

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

        System.out.print("Enter number 1: ");
        int num1 = scanner.nextInt();
        System.out.print("Enter number 2: ");
        int num2 = scanner.nextInt();

        int sum1 = sumOfDivisors(num1);
        int sum2 = sumOfDivisors(num2);

        if (sum1 == num1 && sum2 == num2) {
            System.out.println("Friendly Pair");
        } else {
            System.out.println("Not Friendly Pair");
        }

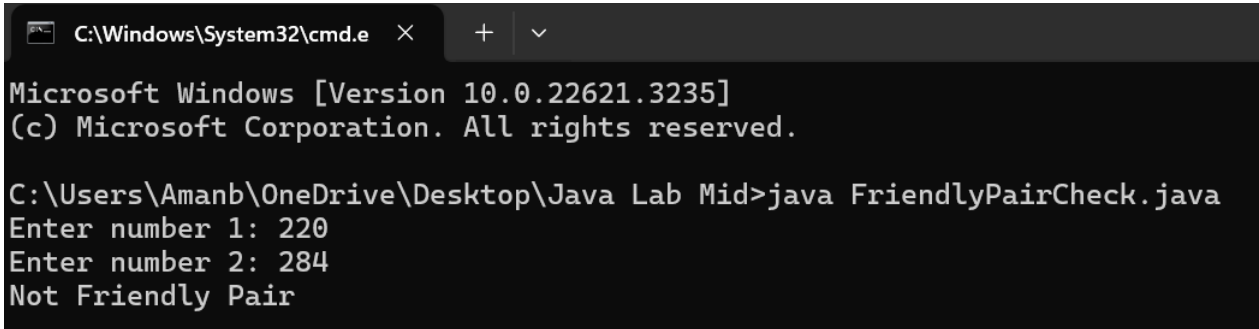
        scanner.close();
    }

    public static int sumOfDivisors(int number) {
        int sum = 1;

        for (int i = 2; i <= Math.sqrt(number); i++) {
            if (number % i == 0) {
                sum += i; // Add divisor
                if (i != number / i) {
                    sum += number / i;
                }
            }
        }
    }
}
```

```
    }  
    return sum;  
}  
}
```

Output: Enter number 1: 220
Enter number 2: 284
Not Friendly Pair



```
C:\Windows\System32\cmd.e  ×  +  ∨  
Microsoft Windows [Version 10.0.22621.3235]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java FriendlyPairCheck.java  
Enter number 1: 220  
Enter number 2: 284  
Not Friendly Pair
```


Q19. Program to replace all 0's with 1 in a given integer. Given an integer as an input, all the 0's in the number has to be replaced with 1.

For example, consider the following number Input: 102405

Output: 112415

Input: 56004

Output: 56114

Steps to replace all 0's with 1 in a given integer

- Input the integer from the user.
- Traverse the integer digit by digit.
- If a '0' is encountered, replace it by '1'.
- Print the integer.

```
import java.util.Scanner;

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

        System.out.print("Enter an integer: ");
        int number = scanner.nextInt();

        String numberString = Integer.toString(number);

        String replacedString = numberString.replace('0', '1');

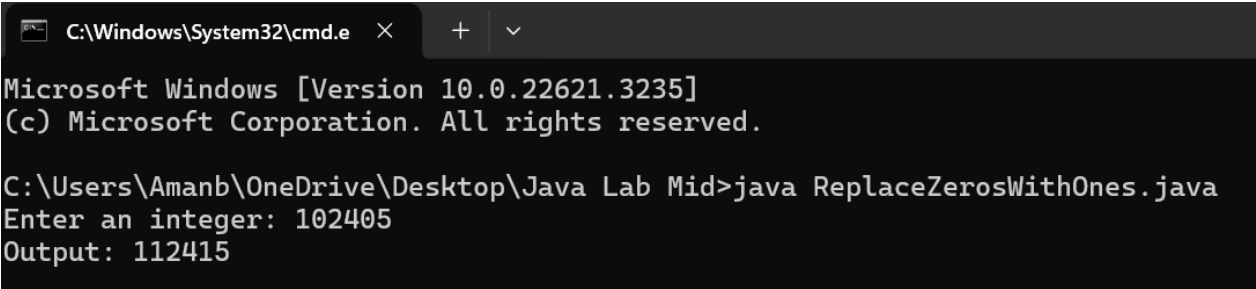
        int replacedNumber = Integer.parseInt(replacedString);

        System.out.println("Output: " + replacedNumber);

        scanner.close();
    }
}
```

Output: Enter an integer: 102405

Output: 112415



```
C:\Windows\System32\cmd.e  ×  +  ▾

Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java ReplaceZerosWithOnes.java
Enter an integer: 102405
Output: 112415
```

Q20. Printing an array into Zigzag fashion. Suppose you were given an array of integers, and you are told to sort the integers in a zigzag pattern. In general, in a zigzag pattern, the first integer is less than the second integer, which is greater than the third integer, which is less than the fourth integer, and so on. Hence, the converted array should be in the form of $e_1 < e_2$

$> e_3$

$< e_4$

$> e_5$

$< e_6$.

Test

cases:

Input

1:7

4 3 7 8 6 2 1

Output 1:

3 7 4 8 2 6 1

Input 2:

4

1 4 3 2

Output 2: 1 4 2 3

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

```
public class ZigzagArray {
    public static void main(String[] args) {
        int[] arr1 = {4, 3, 7, 8, 6, 2, 1};
        int[] arr2 = {1, 4, 3, 2};

        System.out.println("Input 1:");
        System.out.println(Arrays.toString(arr1));
        zigzagSort(arr1);
        System.out.println("Output 1:");
        System.out.println(Arrays.toString(arr1));

        System.out.println("\nInput 2:");
        System.out.println(Arrays.toString(arr2));
        zigzagSort(arr2);
        System.out.println("Output 2:");
        System.out.println(Arrays.toString(arr2));
    }

    public static void zigzagSort(int[] arr) {
        boolean less = true;
        for (int i = 0; i < arr.length - 1; i++) {
            if (less) {
                if (arr[i] > arr[i + 1]) {
                    swap(arr, i, i + 1);
                }
            }
        }
    }
}
```

```

        } else {
            if (arr[i] < arr[i + 1]) {
                swap(arr, i, i + 1);
            }
        }
        less = !less;
    }
}

public static void swap(int[] arr, int i, int j) {
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}
}

```

Output: Input 1:

[4, 3, 7, 8, 6, 2, 1]

Output 1:

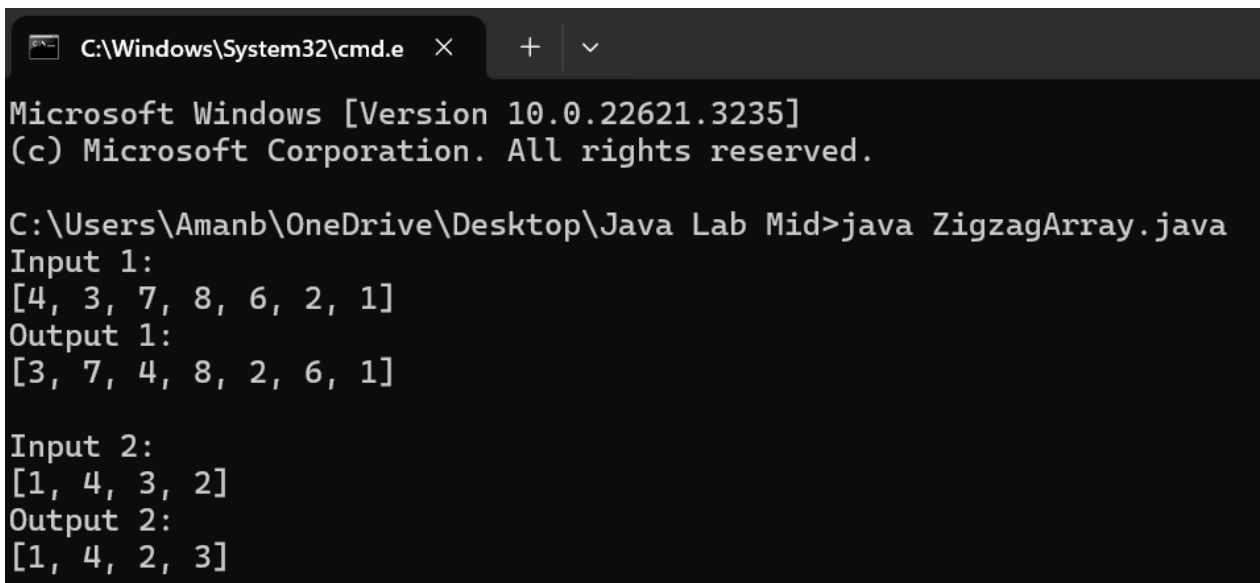
[3, 7, 4, 8, 2, 6, 1]

Input 2:

[1, 4, 3, 2]

Output 2:

[1, 4, 2, 3]



```

C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java ZigzagArray.java
Input 1:
[4, 3, 7, 8, 6, 2, 1]
Output 1:
[3, 7, 4, 8, 2, 6, 1]

Input 2:
[1, 4, 3, 2]
Output 2:
[1, 4, 2, 3]

```

Q21. The problem to rearrange positive and negative numbers in an array .

Method: This approach moves all negative numbers to the beginning and positive numbers to the end but changes the order of appearance of the elements of the array.

Steps:

1. Declare an array and input the array elements.
2. Start traversing the array and if the current element is negative, swap the current element with the first positive element and continue traversing until all the elements have been encountered.
3. Print the rearranged array.

Test case:

- Input: 1 -1 2 -2 3 -3
- Output: -1 -2 -3 1 3 2

```
import java.util.Arrays;

public class RearrangePositiveNegative {
    public static void main(String[] args) {
        int[] arr = {1, -1, 2, -2, 3, -3};
        System.out.println("Input array:");
        System.out.println(Arrays.toString(arr));

        rearrangePositiveNegative(arr);

        System.out.println("Rearranged array:");
        System.out.println(Arrays.toString(arr));
    }

    public static void rearrangePositiveNegative(int[] arr) {
        int j = 0;
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] < 0) {
                if (i != j) {
                    int temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
                j++;
            }
        }
    }
}
```

Output: Input array:

[1, -1, 2, -2, 3, -3]

Rearranged array:

[-1, -2, -3, 1, 2, 3]

```
C:\Windows\System32\cmd.e X + v
Microsoft Windows [Version 10.0.22621.3235]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java RearrangePositiveNegative.java
Input array:
[1, -1, 2, -2, 3, -3]
Rearranged array:
[-1, -2, -3, 1, 3, 2]
```

Q22. Program to find the saddle point coordinates in a given matrix. A saddle point is an element of the matrix, which is the minimum element in its row and the maximum in its column.

For example, consider the matrix given

below Mat [3][3]

1 2 3

4 5 6

7 8 9

Here, 7 is the saddle point because it is the minimum element in its row and maximum element in its column.

Steps to find the saddle point coordinates in a given matrix.

1. Input the matrix from the user.
2. Use two loops, one for traversing the row and the other for traversing the column.
3. If the current element is the minimum element in its row and maximum element in its column, then return its coordinates.

Else, continue traversing

```
import java.util.Scanner;

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

        System.out.print("Enter the number of rows: ");
        int rows = scanner.nextInt();
        System.out.print("Enter the number of columns: ");
        int cols = scanner.nextInt();

        int[][] matrix = new int[rows][cols];

        System.out.println("Enter the elements of the matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }
    }
}
```

```

    }

    int[] saddlePoint = findSaddlePoint(matrix);

    if (saddlePoint == null) {
        System.out.println("No saddle point found.");
    } else {
        System.out.println("Saddle point coordinates: (" + saddlePoint[0] + ", " + saddlePoint[1] + ")");
    }

    scanner.close();
}

public static int[] findSaddlePoint(int[][] matrix) {
    int rows = matrix.length;

    int cols = matrix[0].length;
    for (int i = 0; i < rows; i++) {
        int minInRow = matrix[i][0];
        int colIndexOfMin = 0;
        for (int j = 1; j < cols; j++) {
            if (matrix[i][j] < minInRow) {
                minInRow = matrix[i][j];
                colIndexOfMin = j;
            }
        }

        boolean isSaddlePoint = true;
        for (int k = 0; k < rows; k++) {
            if (matrix[k][colIndexOfMin] > minInRow) {
                isSaddlePoint = false;
                break;
            }
        }

        if (isSaddlePoint) {
            return new int[]{i, colIndexOfMin};
        }
    }

    return null;
}
}

```

Output: Enter the number of rows: 3
Enter the number of columns: 3
Enter the elements of the matrix:
1 2 3
4 5 6

7 8 9

Saddle point coordinates: (2, 0)

```
C:\Windows\System32\cmd.e  ×  +  ∨  
Microsoft Windows [Version 10.0.22621.3235]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\Amanb\OneDrive\Desktop\Java Lab Mid>java SaddlePoint.java  
Enter the number of rows: 3  
Enter the number of columns: 3  
Enter the elements of the matrix:  
1 2 3  
4 5 6  
7 8 9  
Saddle point coordinates: (2, 0)
```

23. String Handling in Java (using String and StringBuffer class): Program to find all the patterns of 0(1+)0 in the given string. Given a string containing 0's and 1's, find the total number of 0(1+)0 patterns in the string and output it. 0(1+)0 - There should be at least one '1' between the two 0's. For example, consider the following string. Input: 01101111010 Output: 3 Explanation: 01101111010 - count = 1.

```
public class ZeroOnePatternCounter {
    public static void main(String[] args) {
        String input = "01101111010";
        int count = countPatterns(input);
        System.out.println("Output: " + count);
    }

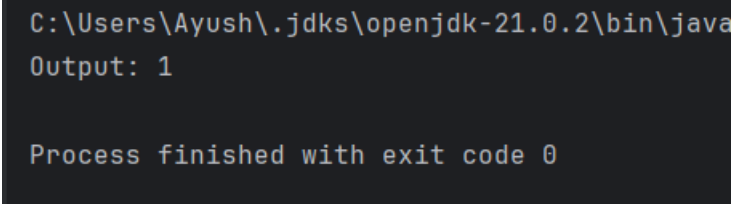
    private static int countPatterns(String input) {
        int count = 0;
        int index = 0;

        while (index < input.length()) {
            int startIndex = input.indexOf("010", index);
            if (startIndex == -1) {
                break;
            }

            count++;
            index = startIndex + 3;
        }

        return count;
    }
}
```

Output:



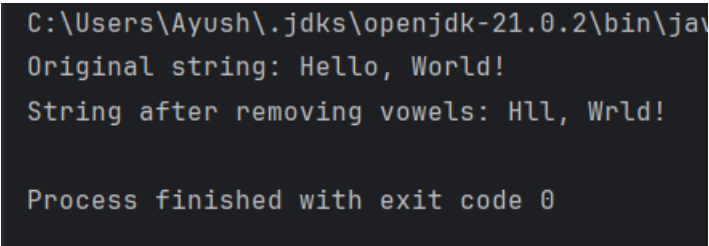
```
C:\Users\Ayush\.jdk\openjdk-21.0.2\bin\java
Output: 1

Process finished with exit code 0
```


24. Write a java program to delete vowels from given string using StringBuffer class.

```
public class VowelRemover {  
    public static void main(String[] args) {  
        String input = "Hello, World!";  
        String result = removeVowels(input);  
        System.out.println("Original string: " + input);  
        System.out.println("String after removing vowels: " + result);  
    }  
  
    private static String removeVowels(String input) {  
        StringBuffer buffer = new StringBuffer(input);  
  
        for (int i = 0; i < buffer.length(); i++) {  
            char ch = buffer.charAt(i);  
            if (isVowel(ch)) {  
                buffer.deleteCharAt(i);  
                i--; // Adjust index after deletion  
            }  
        }  
  
        return buffer.toString();  
    }  
  
    private static boolean isVowel(char ch) {  
        ch = Character.toLowerCase(ch);  
        return ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u';  
    }  
}
```

Output:



```
C:\Users\Ayush\.jdk\openjdk-21.0.2\bin\jav  
Original string: Hello, World!  
String after removing vowels: Hll, Wrld!  
  
Process finished with exit code 0
```

25. Class definition, creating objects and constructors:

Write a java program to create a class named 'Bank ' with the following data members:

- Name of depositor
- Address of depositor
- Account Number
- Balance in account

Class 'Bank' has a method for each of the following:

1. Generate a unique account number for each depositor.
2. For first depositor, account number will be 1001, for second depositor it will be 1002 and so on
3. Display information and balance of depositor
4. Deposit more amount in balance of any depositor
5. Withdraw some amount from balance deposited.
6. Change address of depositor

After creating the class, do the following operations.

1. Enter the information (name, address, account number, balance) of the depositors. Number of depositors is to be entered by the user.
2. Print the information of any depositor.
3. Add some amount to the account of any depositor and then display final information of that depositor.
4. Remove some amount from the account of any depositor and then display final information of that depositor.
5. Change the address of any depositor and then display the final information of that depositor.
6. Randomly repeat these processes for some other bank accounts.

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

```
public class BankManagement {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of depositors: ");
        int numDepositors = scanner.nextInt();

        Bank[] depositors = new Bank[numDepositors];

        for (int i = 0; i < numDepositors; i++) {
            System.out.println("\nEnter details for depositor " + (i + 1) + ":");
            scanner.nextLine(); // Consume newline
            System.out.print("Name: ");
            String name = scanner.nextLine();
            System.out.print("Address: ");
            String address = scanner.nextLine();
            System.out.print("Balance: $");
            double balance = scanner.nextDouble();

            depositors[i] = new Bank(name, address, balance);
        }

        System.out.println("\nDetails of any depositor:");
        depositors[0].displayInfo();

        System.out.println("\nAdding amount to the account of any depositor:");
        depositors[0].deposit(500);
    }
}
```

```
System.out.println("\nDetails after deposit:");
depositors[0].displayInfo();
```

```
System.out.println("\nWithdrawing amount from the account of any depositor:");
depositors[1].withdraw(200);
```

```
System.out.println("\nDetails after withdrawal:");
depositors[1].displayInfo();
```

```
System.out.println("\nChanging address of any depositor:");
depositors[2].changeAddress("New Address");
```

```
System.out.println("\nFinal details after address change:");
depositors[2].displayInfo();
```

```
scanner.close();
```

```
}
```

```
}
```

```
class Bank {
```

```
    private static int accountCounter = 1000;
```

```
    private String name;
```

```
    private String address;
```

```
    private int accountNumber;
```

```
    private double balance;
```

```
    public Bank(String name, String address, double balance) {
```

```
        this.name = name;
```

```
        this.address = address;
```

```
        this.accountNumber = generateAccountNumber();
```

```
        this.balance = balance;
```

```
    }
```

```
    private int generateAccountNumber() {
```

```
        return ++accountCounter;
```

```
    }
```

```
    public void displayInfo() {
```

```
        System.out.println("Name: " + name);
```

```
        System.out.println("Address: " + address);
```

```
        System.out.println("Account Number: " + accountNumber);
```

```
        System.out.println("Balance: $" + balance);
```

```
    }
```

```
    public void deposit(double amount) {
```

```
        balance += amount;
```

```
        System.out.println("Deposit successful. New balance: $" + balance);
```

```
    }
```

```
    public void withdraw(double amount) {
```

```
        if (balance >= amount) {
```

```
            balance -= amount;
```

```
            System.out.println("Withdrawal successful. New balance: $" + balance);
```

```
        } else {
```

```
            System.out.println("Insufficient funds.");
```

```
        }
```

```
    }
```

```

public void changeAddress(String newAddress) {
    address = newAddress;
    System.out.println("Address updated successfully.");
}
}

```

Output:

```

Enter details for depositor 1:
Name: Ayush Debnath
Address: Guwahati
Balance: $10000000000

Enter details for depositor 2:
Name: Mayank Singh
Address: Bareilly
Balance: $1000000

Enter details for depositor 3:
Name: kapil Yadav
Address: KhatimA
Balance: $1000000

Details of any depositor:
Name: Ayush Debnath
Address: Guwahati
Account Number: 1001
Balance: $1.0E10

Adding amount to the account of any depositor:
Deposit of $500.0 successful.

Details after deposit:
Name: Ayush Debnath
Address: Guwahati
Account Number: 1001
Balance: $1.000000005E10

```

```

Withdrawing amount from the account of any depositor:
Withdrawal of $200.0 successful.

```

```

Details after withdrawal:
Name: Mayank Singh
Address: Bareilly
Account Number: 1002
Balance: $999800.0

```

```

Changing address of any depositor:
Address changed successfully.

```

```

Final details after address change:
Name: kapil Yadav
Address: New Address
Account Number: 1003
Balance: $1000000.0

```

26. Define a class Word Example having the following description:

Data members/instance variables:

private String strdata: to store a sentence.

Parameterized Constructor WordExample(String) : Accept a sentence which may be terminated by either '.', '? 'or'!' only. The words may be separated by more than one blank space and are in UPPER CASE.

Member Methods:

void countWord(): Find the number of words beginning and ending with a vowel.

void placeWord(): Place the words which begin and end with a vowel at the beginning, followed by the remaining words as they occur in the sentence

```
public class WordExample {
    private String strdata;

    public WordExample(String strdata) {
        this.strdata = strdata;
    }

    public void countWord() {
        String[] words = strdata.split("\\s+");
        int count = 0;
        for (String word : words) {
            if (startsWithVowel(word) && endsWithVowel(word)) {
                count++;
            }
        }
        System.out.println("Number of words beginning and ending with a vowel: " + count);
    }

    public void placeWord() {
        String[] words = strdata.split("\\s+");
        StringBuilder result = new StringBuilder();

        // First, add words beginning and ending with a vowel
        for (String word : words) {
            if (startsWithVowel(word) && endsWithVowel(word)) {
                result.insert(0, word + " ");
            }
        }

        // Then, add remaining words
        for (String word : words) {
            if (!startsWithVowel(word) || !endsWithVowel(word)) {
                result.append(word).append(" ");
            }
        }

        System.out.println("Sentence after placing words beginning and ending with a vowel at the beginning:");
        System.out.println(result.toString().trim());
    }

    private boolean startsWithVowel(String word) {
        return "AEIOU".indexOf(word.charAt(0)) != -1;
    }

    private boolean endsWithVowel(String word) {
        return "AEIOU".indexOf(word.charAt(word.length() - 1)) != -1;
    }
}
```

```
public static void main(String[] args) {  
    WordExample wordExample = new WordExample("APPLE ORANGE is a FRUIT. ELEPHANT is big and  
STRONG.");  
    wordExample.countWord();  
    wordExample.placeWord();  
}  
}
```

Output:

```
C:\Users\Ayush\.jdk\openjdk-21.0.2\bin\java.exe --enable-preview "-javaagent:C:\Prog  
Number of words beginning and ending with a vowel: 2  
Sentence after placing words beginning and ending with a vowel at the beginning:  
ORANGE APPLE is a FRUIT. ELEPHANT is big and STRONG.
```

27. Method overloading (Compile time Polymorphism):

Write a Java program to create a class called ArrayDemo and overload arrayFunc() function.

void arrayFunc(int [], int) To find all pairs of elements in an Array whose sum is equal to a given number :

Array numbers= [4, 6, 5, -10, 8, 5, 20], target=10

Output :

Pairs of elements whose sum is 10 are :4 + 6 = 10

5 + 5 = 10 -10 + 20 = 10

void arrayFunc(int A[], int p, int B[], int q) Given two sorted arrays A and B of size p and q, Overload method arrayFunc() to merge elements of A with B by maintaining the sorted order i.e. fill A with first p smallest elements and fill B with remaining elements.

Example:

Input :

int[] A = { 1, 5, 6, 7, 8, 10 }

int[] B = { 2, 4, 9 }

Output:

Sorted Arrays:

A: [1, 2, 4, 5, 6, 7]

B: [8, 9, 10]

(Use Compile time Polymorphism MethodOverloading)

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

```
public class ArrayDemo {
    // Method to find all pairs of elements whose sum is equal to a given number
    public void arrayFunc(int[] arr, int target) {
        System.out.println("Pairs of elements whose sum is " + target + " are:");
        for (int i = 0; i < arr.length; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] + arr[j] == target) {
                    System.out.println(arr[i] + " + " + arr[j] + " = " + target);
                }
            }
        }
    }

    // Method to merge elements of A with B by maintaining the sorted order
    public void arrayFunc(int[] A, int p, int[] B, int q) {
        int[] mergedArray = new int[p + q];
        int i = 0, j = 0, k = 0;

        // Merge elements maintaining sorted order
        while (i < p && j < q) {
            if (A[i] < B[j]) {
                mergedArray[k++] = A[i++];
            } else {
                mergedArray[k++] = B[j++];
            }
        }

        // Copy remaining elements of A and B
        while (i < p) {
            mergedArray[k++] = A[i++];
        }
    }
}
```

```

        while (j < q) {
            mergedArray[k++] = B[j++];
        }

        // Split merged array into A and B
        System.arraycopy(mergedArray, 0, A, 0, p);
        System.arraycopy(mergedArray, p, B, 0, q);

        // Sort arrays A and B
        Arrays.sort(A);
        Arrays.sort(B);

        // Print sorted arrays
        System.out.println("Sorted Arrays:");
        System.out.println("A: " + Arrays.toString(A));
        System.out.println("B: " + Arrays.toString(B));
    }

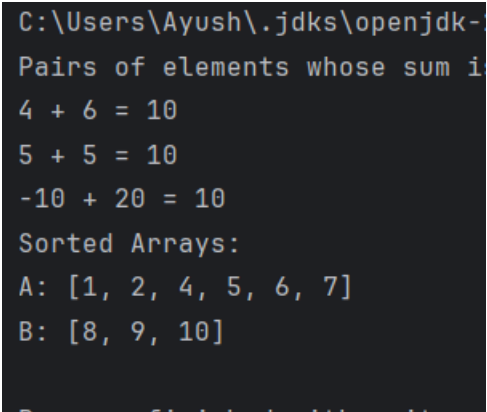
    public static void main(String[] args) {
        ArrayDemo demo = new ArrayDemo();

        // Test case 1 for finding pairs
        int[] numbers = {4, 6, 5, -10, 8, 5, 20};
        int target = 10;
        demo.arrayFunc(numbers, target);

        // Test case 2 for merging arrays
        int[] A = {1, 5, 6, 7, 8, 10};
        int[] B = {2, 4, 9};
        demo.arrayFunc(A, A.length, B, B.length);
    }
}

```

Output:



```

C:\Users\Ayush\.jdk\openjdk-11.0.10\bin\java.exe
Pairs of elements whose sum is 10
4 + 6 = 10
5 + 5 = 10
-10 + 20 = 10
Sorted Arrays:
A: [1, 2, 4, 5, 6, 7]
B: [8, 9, 10]
Process finished with exit code 0

```


28. Method overriding (Runtime Polymorphism), Abstract class and Abstract method, Inheritance, interfaces:

Write a java program to calculate the area of a rectangle, a square and a circle.

Create an abstract class 'Shape' with three abstract methods namely rectangleArea() taking two parameters, squareArea() and circleArea() taking one parameter each.

Now create another class 'Area' containing all the three methods

rectangleArea(), squareArea() and circleArea() for printing the area of rectangle, square and circle respectively. Create an object of class Area and call all the three methods.

(Use Runtime Polymorphism)

```
public class Main {
    public static void main(String[] args) {
        Area area = new Area();
        double length = 5.0;
        double width = 3.0;
        double side = 4.0;
        double radius = 2.5;
        System.out.println("Area of Rectangle: " + area.rectangleArea(length, width));
        System.out.println("Area of Square: " + area.squareArea(side));
        System.out.println("Area of Circle: " + area.circleArea(radius));
    }
}

abstract class Shape {
    abstract double rectangleArea(double length, double width);
    abstract double squareArea(double side);
    abstract double circleArea(double radius);
}

class Area extends Shape {
    @Override
    double rectangleArea(double length, double width) {
        return length * width;
    }
    @Override
    double squareArea(double side) {
        return side * side;
    }
    @Override
    double circleArea(double radius) {
        return Math.PI * radius * radius;
    }
}
```

Output:

```
C:\Users\Ayush\.jdk\openjdk-21.0.2\bin
Area of Rectangle: 15.0
Area of Square: 16.0
Area of Circle: 19.634954084936208
```

29. Write a java program to implement abstract class and abstract method with following details:

Create a abstract Base Class TemperatureData members:

double temp;

Method members:

void setTempData(double) abstract void changeTemp()

Sub Class Fahrenheit (subclass of Temperature) Data members:

double ctemp;

method member:

Override abstract method changeTemp() to convert Fahrenheit temperature into degree Celsius by using formula $C = 5/9 * (F - 32)$ and display converted temperature

Sub Class Celsius (subclass of Temperature)

Data member:

double ftemp;

Method member:

Override abstract method changeTemp() to convert degree Celsius into Fahrenheit temperature by using formula $F = 9/5 * c + 32$ and display converted Temperature

```
abstract class Temperature {
double temp;
void setTempData(double temp) {
this.temp = temp;
}
abstract void changeTemp();
}
class Fahrenheit extends Temperature {
void changeTemp() {
double ctemp = 5.0 / 9.0 * (temp - 32);
System.out.println(temp + " Fahrenheit is " + ctemp + " Celsius");
}
}
class Celsius extends Temperature {
void changeTemp() {
double ftemp = (9.0 / 5.0) * temp + 32;
System.out.println(temp + " Celsius is " + ftemp + " Fahrenheit");
}
}
public class TemperatureConverter {
public static void main(String[] args) {
Fahrenheit fahrenheit = new Fahrenheit();
fahrenheit.setTempData(98.6);
fahrenheit.changeTemp();
Celsius celsius = new Celsius();
celsius.setTempData(37);
celsius.changeTemp();
}
}
```

Output

Clear

```
java -cp /tmp/CSILBk0Lg7/TemperatureConverter
98.6 Fahrenheit is 37.0 Celsius
37.0 Celsius is 98.60000000000001 Fahrenheit
=== Code Execution Successful ===
```

30. Write a java program to create an interface that consists of a method to display **volume ()** as an abstract method and redefine this method in the derived 2 classes to suit their requirements.

Create classes called **Cone**, **Hemisphere** and **Cylinder** that implements the interface. Using these three classes, design a program that will accept dimensions of a cone, cylinder and hemisphere interactively and display the volumes.

Volume of cone = $(1/3)\pi r^2 h$ Volume of hemisphere = $(2/3)\pi r^3$ Volume of cylinder = $\pi r^2 h$

```
import java.util.Scanner;
interface Shape {
    void displayVolume();
}
class Cone implements Shape {
    private double radius;
    private double height;
    public Cone(double radius, double height) {
        this.radius = radius;
        this.height = height;
    }
    public void displayVolume() {
        double volume = (1.0 / 3) * Math.PI * Math.pow(radius, 2) * height;
        System.out.printf("Volume of Cone: %.2f%n", volume);
    }
}
class Hemisphere implements Shape {
    private double radius;
    public Hemisphere(double radius) {
        this.radius = radius;
    }
    public void displayVolume() {
        double volume = (2.0 / 3) * Math.PI * Math.pow(radius, 3);
        System.out.printf("Volume of Hemisphere: %.2f%n", volume);
    }
}
class Cylinder implements Shape {
    private double radius;
    private double height;
    public Cylinder(double radius, double height) {
        this.radius = radius;
        this.height = height;
    }
    public void displayVolume() {
        double volume = Math.PI * Math.pow(radius, 2) * height;
        System.out.printf("Volume of Cylinder: %.2f%n", volume);
    }
}
public class ShapeVolumeCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the radius of the cone: ");
        double coneRadius = scanner.nextDouble();
        System.out.print("Enter the height of the cone: ");
        double coneHeight = scanner.nextDouble();
        Cone cone = new Cone(coneRadius, coneHeight);
```

```
System.out.print("Enter the radius of the hemisphere: ");
double hemisphereRadius = scanner.nextDouble();
Hemisphere hemisphere = new Hemisphere(hemisphereRadius);
System.out.print("Enter the radius of the cylinder: ");
double cylinderRadius = scanner.nextDouble();
System.out.print("Enter the height of the cylinder: ");
double cylinderHeight = scanner.nextDouble();
Cylinder cylinder = new Cylinder(cylinderRadius, cylinderHeight);
System.out.println();
cone.displayVolume();
hemisphere.displayVolume();
cylinder.displayVolume();
}
}
```

Output

```
java -cp /tmp/Y1fq4E0KsX/ShapeVolumeCalculator
Enter the radius of the cone: 45
Enter the height of the cone: 67
Enter the radius of the hemisphere: 34
Enter the radius of the cylinder: 23
Enter the height of the cylinder: 56

Volume of Cone: 142078.53
Volume of Hemisphere: 82318.11
Volume of Cylinder: 93066.54
```

31. Write a java program to accept and print the employee details during runtime. The details will include employee id, name, dept_ Id. The program should raise an exception if user inputs incomplete or incorrect data. The entered value should meet the following conditions:

- a. First Letter of employee name should be in capital letter.
- b. Employee id should be between 2001 and 5001
- c. Department id should be an integer between 1 and 5.

If the above conditions are not met, then the application should raise specific exception else should complete normal execution.

```
import java.util.Scanner;
class InvalidNameException extends Exception {
public InvalidNameException(String message) {
super(message);
}
}
class InvalidEmployeeIdException extends Exception {
public InvalidEmployeeIdException(String message) {
super(message);
}
}
class InvalidDeptIdException extends Exception {
public InvalidDeptIdException(String message) {
super(message);
}
}
class Employee {
private int employeeId;
private String name;
private int deptId;
public Employee(int employeeId, String name, int deptId) {
this.employeeId = employeeId;
this.name = name;
this.deptId = deptId;
}
public void displayDetails() {
System.out.println("Employee ID: " + employeeId);
System.out.println("Employee Name: " + name);
System.out.println("Department ID: " + deptId);
}
public static Employee inputEmployeeDetails(Scanner scanner) throws InvalidNameException,
InvalidEmployeeIdException, InvalidDeptIdException {
System.out.print("Enter Employee ID: ");
int employeeId = scanner.nextInt();
scanner.nextLine();
System.out.print("Enter Employee Name: ");
String name = scanner.nextLine();
System.out.print("Enter Department ID: ");
int deptId = scanner.nextInt();
if (employeeId < 2001 || employeeId > 5001) {
throw new InvalidEmployeeIdException("Employee ID must be between 2001 and 5001.");
}
if (!Character.isUpperCase(name.charAt(0))) {
throw new InvalidNameException("First letter of employee name should be in capital letter.");
}
}
```

```

    if (deptId < 1 || deptId > 5) {
    throw new InvalidDeptIdException("Department ID must be between 1 and 5.");
    }
    return new Employee(employeeId, name, deptId);
    }
    }
    public class EmployeeDetails {
    public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
try {
    Employee employee = Employee.inputEmployeeDetails(scanner);
    System.out.println("\nEmployee Details:");
    employee.displayDetails();
} catch (InvalidNameException | InvalidEmployeeIdException | InvalidDeptIdException e) {
    System.err.println("Error: " + e.getMessage());
} finally {
    scanner.close();
}
    }
    }
}

```

Output

Clear

```
java -cp /tmp/7QubuxPmti/EmployeeDetails
```

```
Enter Employee ID: 2005
```

```
Enter Employee Name: Deepanshu Kandpal
```

```
Enter Department ID: 03
```

```
Employee Details:
```

```
Employee ID: 2005
```

```
Employee Name: Deepanshu Kandpal
```

```
Department ID: 3
```

```
=== Code Execution Successful ===
```

32. Create a class MyCalculator which consists of a single method power (int, int). This method takes two integers, n and p, as parameters and finds n^p . If either n or p is negative, then the method must throw an exception which says, "n and p should be non- negative".

Input Format

Each line of the input contains two integers, n and p. Output Format

Each line of the output contains the result, if neither of n and p is negative.

Otherwise, the output contains "n and p should be non- negative".

java.lang.Exception: n and p should not be zero. java.lang.Exception: n or p should not be negative. java.lang. Exception: n or p should not be negative.

Explanation

In the first two cases, both n and p are positive. So, the power function returns the answer correctly.

In the third case, both n and p are zero. So, the exception, "n and p should not be zero." is printed.

In the last two cases, at least one out of n and p is negative. So, the exception, "n or p should not be negative." is printed for these two cases.

```
import java.util.Scanner;
class MyCalculator {
    public int power(int n, int p) throws Exception {
        if (n < 0 || p < 0) {
            throw new Exception("n or p should not be negative.");
        }
        if (n == 0 && p == 0) {
            throw new Exception("n and p should not be zero.");
        }
        return (int) Math.pow(n, p);
    }
}
public class Solution {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        MyCalculator myCalculator = new MyCalculator();
        while (scanner.hasNext()) {
            int n = scanner.nextInt();
            int p = scanner.nextInt();
            try {
                int result = myCalculator.power(n, p);
                System.out.println(result);
            } catch (Exception e) {
                System.out.println(e);
            }
        }
        scanner.close();
    }
}
```

```
java -cp ./tmp/VS7QUUM1AB7/Solution
3 5
243
2 4
16
0 0
java.lang.Exception: n and p should not be zero.
-1 -2 -1 -2
java.lang.Exception: n or p should not be negative.
-1 3 -1 3
java.lang.Exception: n or p should not be negative.
```

33. File Handling in Java:

Write a java file handling program to count and display the number of palindromes present in a text file "myfile.txt".

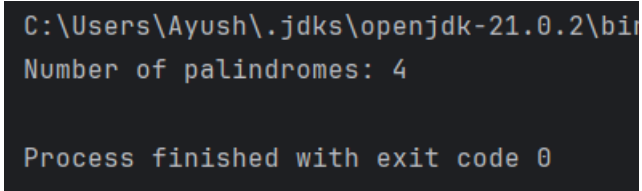
Example: If the file "myfile.txt" contains the following lines,

My name is NITIN

Hello aaa and bbb wordHow are You

ARORA is my friendOutput will be => 4

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
public class PalindromeCounter {
    public static void main(String[] args) {
        String filename = "myfile.txt";
        int palindromeCount = 0;
        try (BufferedReader br = new BufferedReader(new FileReader(filename))) {
            String line;
            while ((line = br.readLine()) != null) {
                String[] words = line.split("\\s+");
                for (String word : words) {
                    if (isPalindrome(word)) {
                        palindromeCount++;
                    }
                }
            }
        } catch (IOException e) {
            System.err.println("Error reading the file: " + e.getMessage());
        }
        System.out.println("Number of palindromes: " + palindromeCount);
    }
    private static boolean isPalindrome(String word) {
        int left = 0;
        int right = word.length() - 1;
        while (left < right) {
            if (Character.toLowerCase(word.charAt(left)) != Character.toLowerCase(word.charAt(right))) {
                return false;
            }
            left++;
            right--;
        }
        return true;
    }
}
```



```
C:\Users\Ayush\.jdk\openjdk-21.0.2\bin
Number of palindromes: 4

Process finished with exit code 0
```


34. Multithreaded programming:

Write a program MultiThreads that creates two threads-one thread with the name CSthread and the other thread named ITthread. Each thread should display its respective name and execute after a gap of 500 milliseconds. Each thread should also display a number indicating the number of times it got a chance to execute.

```
class CustomThread extends Thread {
private String threadName;
private int executionCount;
public CustomThread(String name) {
    this.threadName = name;
    this.executionCount = 0;
}
public void run() {
    try {
        while (true) {
            executionCount++;
            System.out.println(threadName + " - Execution count: " + executionCount);
            Thread.sleep(500);
        }
    } catch (InterruptedException e) {
        System.out.println(threadName + " interrupted.");
    }
}
}

public class MultiThreads {
public static void main(String[] args) {
    CustomThread csthread = new CustomThread("CSthread");
    CustomThread itthread = new CustomThread("ITthread");
    csthread.start();
    itthread.start();
}
}
```

Output Clear

```
java -cp /tmp/fIEl0jc1ob/MultiThreads
ITthread - Execution count: 1
CSthread - Execution count: 1
ITthread - Execution count: 2
CSthread - Execution count: 2
ITthread - Execution count: 3
CSthread - Execution count: 3
ITthread - Execution count: 4
CSthread - Execution count: 4
ITthread - Execution count: 5
CSthread - Execution count: 5
ITthread - Execution count: 6
CSthread - Execution count: 6
ITthread - Execution count: 7
CSthread - Execution count: 7
ITthread - Execution count: 8
CSthread - Execution count: 8
ITthread - Execution count: 9
CSthread - Execution count: 9
ITthread - Execution count: 10
CSthread - Execution count: 10
```

35. Write a java program for to solve producer consumer problem in which a producer produces a value and consumer consume the value before producer generate the next value

```
class SharedResource {
private int data;
private boolean isProduced = false;
public synchronized void produce(int value) throws InterruptedException {
    while (isProduced) {
        wait();
    }
    data = value;
    isProduced = true;
    System.out.println("Produced: " + data);
    notify();
}
public synchronized void consume() throws InterruptedException {
    while (!isProduced) {
        wait();
    }
    System.out.println("Consumed: " + data);
    isProduced = false;
    notify();
}
}

class Producer extends Thread {
private SharedResource sharedResource;
public Producer(SharedResource sharedResource) {
    this.sharedResource = sharedResource;
}
public void run() {
    try {
        for (int i = 0; i < 10; i++) {
            sharedResource.produce(i);
            Thread.sleep(500);
        }
    } catch (InterruptedException e) {
        Thread.currentThread().interrupt();
    }
}

class Consumer extends Thread {
private SharedResource sharedResource;
public Consumer(SharedResource sharedResource) {
    this.sharedResource = sharedResource;
}
public void run() {
    try {
        for (int i = 0; i < 10; i++) {
            sharedResource.consume();
            Thread.sleep(500);
        }
    } catch (InterruptedException e) {
        Thread.currentThread().interrupt();
    }
}
```

```
    }  
    }  
    public class ProducerConsumer {  
public static void main(String[] args) {  
    SharedResource sharedResource = new SharedResource();  
    Producer producer = new Producer(sharedResource);  
    Consumer consumer = new Consumer(sharedResource);  
    producer.start();  
    consumer.start();  
}  
}
```

Output

[Clear](#)

```
java -cp /tmp/NTjMwF3x2S/ProducerConsumer
```

```
Produced: 0
```

```
Consumed: 0
```

```
Produced: 1
```

```
Consumed: 1
```

```
Produced: 2
```

```
Consumed: 2
```

```
Produced: 3
```

```
Consumed: 3
```

```
Produced: 4
```

```
Consumed: 4
```

```
Produced: 5
```

```
Consumed: 5
```

```
Produced: 6
```

```
Consumed: 6
```

```
Produced: 7
```

```
Consumed: 7
```

```
Produced: 8
```

```
Consumed: 8
```

```
Produced: 9
```

```
Consumed: 9
```

```
=== Code Execution Successful ===
```

36. Collection and Generic Framework:

Write a method `removeEvenLength` that takes an `ArrayList` of `Strings` as a parameter and that removes all the strings of even length from the list.

(Use `ArrayList`)

```
import java.util.ArrayList;
public class RemoveEvenLengthStrings {
public static void removeEvenLength(ArrayList<String> list) {
    list.removeIf(str -> str.length() % 2 == 0);
}
public static void main(String[] args) {
    ArrayList<String> strings = new ArrayList<>();
    strings.add("apple");
    strings.add("banana");
    strings.add("pear");
    strings.add("kiwi");
    strings.add("plum");
    System.out.println("Original list: " + strings);
    removeEvenLength(strings);
    System.out.println("List after removing even length strings: " + strings);
}
}
```

Output

Clear

```
java -cp /tmp/cvh5P2VlWB/RemoveEvenLengthStrings
Original list: [apple, banana, pear, kiwi, plum]
List after removing even length strings: [apple]

=== Code Execution Successful ===
```

37. Write a method `swapPairs` that switches the order of values in an `ArrayList` of `Strings` in a pairwise fashion. Your method should switch the order of the first two values, then switch the order of the next two, switch the order of the next two, and so on.

For example, if the list initially stores these values: `{"four", "score", "and", "seven", "years", "ago"}` your method should switch the first pair, "four", "score", the second pair, "and", "seven", and the third pair, "years", "ago", to yield this list: `{"score", "four", "seven", "and", "ago", "years"}`

If there are an odd number of values in the list, the final element is not moved.

For example, if the original list had been: `{"to", "be", "or", "not", "to", "be", "hamlet"}` It would again switch pairs of values, but the final value, "hamlet" would not be moved, yielding this list: `{"be", "to", "not", "or", "be", "to", "hamlet"}`

```
import java.util.ArrayList;
import java.util.Arrays;
public class SwapPairs {
public static void swapPairs(ArrayList<String> list) {
    for (int i = 0; i < list.size() - 1; i += 2) {
        String temp = list.get(i);
        list.set(i, list.get(i + 1));
        list.set(i + 1, temp);
    }
}
public static void main(String[] args) {
    ArrayList<String> list1 = new ArrayList<>(Arrays.asList("four", "score", "and", "seven", "years", "ago"));
    System.out.println("Original list1: " + list1);
    swapPairs(list1);
    System.out.println("List1 after swapping pairs: " + list1);
    ArrayList<String> list2 = new ArrayList<>(Arrays.asList("to", "be", "or", "not", "to", "be", "hamlet"));
    System.out.println("Original list2: " + list2);
    swapPairs(list2);
    System.out.println("List2 after swapping pairs: " + list2);
}
}
```

Output

Clear

```
java -cp /tmp/pDv1Q4TinF/SwapPairs
```

```
Original list1: [four, score, and, seven, years, ago]
```

```
List1 after swapping pairs: [score, four, seven, and, ago, years]
```

```
Original list2: [to, be, or, not, to, be, hamlet]
```

```
List2 after swapping pairs: [be, to, not, or, be, to, hamlet]
```

```
=== Code Execution Successful ===
```

38. Write a method called `alternate` that accepts two `List`s of integers as its parameters and returns a new `List` containing alternating elements from the two lists, in the following order:

- First element from first list
- First element from second list
- Second element from first list
- Second element from second list
- Third element from first list
- Third element from second list

If the lists do not contain the same number of elements, the remaining elements from the longer list should be placed consecutively at the end. For example, for a first list of (1, 2, 3, 4, 5) and a second list of (6, 7, 8, 9, 10, 11, 12), a call of `alternate(list1, list2)` should return a list containing (1, 6, 2, 7, 3, 8, 4, 9, 5, 10, 11, 12). Do not modify the parameter lists passed in.

```
import java.util.ArrayList;
import java.util.List;
public class AlternateLists {
    public static List<Integer> alternate(List<Integer> list1, List<Integer> list2) {
        List<Integer> result = new ArrayList<>();
        int maxLength = Math.max(list1.size(), list2.size());
        for (int i = 0; i < maxLength; i++) {
            if (i < list1.size()) {
                result.add(list1.get(i));
            }
            if (i < list2.size()) {
                result.add(list2.get(i));
            }
        }
        return result;
    }
    public static void main(String[] args) {
        List<Integer> list1 = List.of(1, 2, 3, 4, 5);
        List<Integer> list2 = List.of(6, 7, 8, 9, 10, 11, 12);
        System.out.println("List 1: " + list1);
        System.out.println("List 2: " + list2);
        List<Integer> result = alternate(list1, list2);
        System.out.println("Alternating List: " + result);
    }
}
```

Output

Clear

```
java -cp /tmp/Z6r9G72aFV/AlternateLists
List 1: [1, 2, 3, 4, 5]
List 2: [6, 7, 8, 9, 10, 11, 12]
Alternating List: [1, 6, 2, 7, 3, 8, 4, 9, 5, 10, 11, 12]

=== Code Execution Successful ===
```

39.AWT & Swing, Event Handling: Write a GUI program to develop an application that receives a string in one text field, and count number of vowels in a string and returns it in another text field, when the button named “CountVowel” is clicked. When the button named “Reset” is clicked it will reset the value of textfield one and Textfield two. When the button named “Exit” is clicked it will close the application.

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class VowelCounterApp extends JFrame implements ActionListener {

    private JTextField inputField;

    private JTextField outputField;

    private JButton countButton;

    private JButton resetButton;

    private JButton exitButton;

    public VowelCounterApp() {

        setTitle("Vowel Counter");

        setSize(400, 200);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setLayout(new GridLayout(4, 2, 10, 10));

        JLabel inputLabel = new JLabel("Enter String:");

        inputField = new JTextField();

        JLabel outputLabel = new JLabel("Number of Vowels:");

        outputField = new JTextField();

        outputField.setEditable(false);

        countButton = new JButton("CountVowel");

        resetButton = new JButton("Reset");

        exitButton = new JButton("Exit");

        add(inputLabel);

        add(inputField);

        add(outputLabel);
```

```
add(outputField);

    add(countButton);

    add(resetButton);

    add(exitButton);

    countButton.addActionListener(this);

    resetButton.addActionListener(this);

    exitButton.addActionListener(this);

    setVisible(true);
}

@Override

public void actionPerformed(ActionEvent e) {

    if (e.getSource() == countButton) {

        String inputText = inputField.getText();

        int vowelCount = countVowels(inputText);

        outputField.setText(String.valueOf(vowelCount));

    } else if (e.getSource() == resetButton) {

        inputField.setText("");

        outputField.setText("");

    } else if (e.getSource() == exitButton) {

        System.exit(0);

    }

}

private int countVowels(String input) {

    int count = 0;

    String vowels = "aeiouAEIOU";

    for (char ch : input.toCharArray()) {

        if (vowels.indexOf(ch) != -1) {

            count++;

        }

    }

}
```



```
        return count;
    }

    public static void main(String[] args) {
        new VowelCounterApp();
    }
}
```

