College of Engineering, Thiruvananthapuram

Object Oriented Programming Lab

Anirudh A. V.
S2 CSE R2, Roll No. 11

Department of Computer Science
& Engineering

December 23, 2021

1 **Prime Number**

1.1 Aim

To write a java program to check whether a number is prime or not.

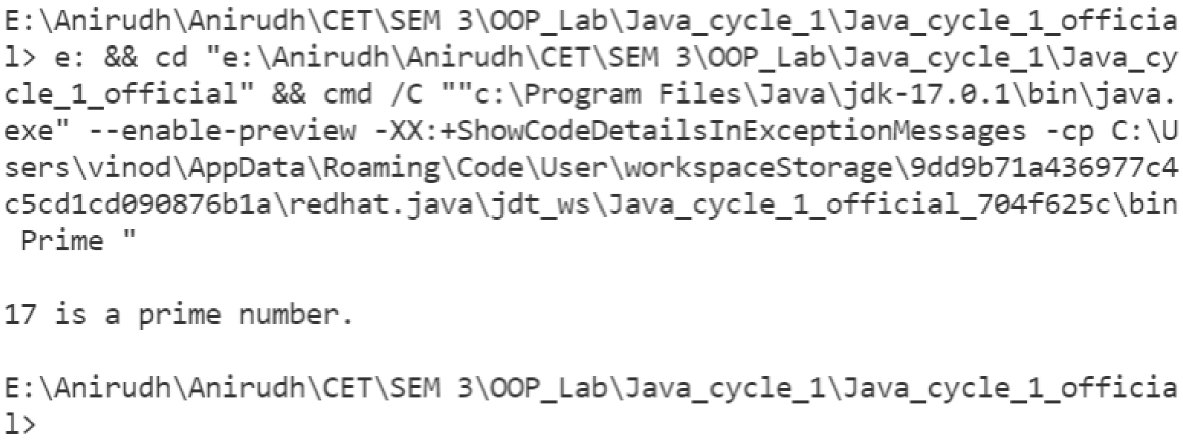
1.2 Algorithm

```
Step - 1   Start
Step - 2   Declare and initialise variables num = 17, flag = 0, i=2.
Step - 3   If num <= 1, Display “number is not a prime number”. Go to step 6.
Step - 4   Repeat the following steps until i < [(num/2) +1].
    Step - 4.1.   If (num % i) = 0, increment flag, and break the loop, go to step 5.
    Step - 4.2.   Increment i.
Step - 5   If flag not equal to zero, Display “number is not a prime number”.
    Step - 5.1.   Else Display “number is a prime number”.
Step - 6   Stop
```

1.3 Code

```
/** This program is intended to check whether a number is prime or not.
    This is written by Anirudh A V on 22 Dec 2021*/

public class Prime{
    public static void main(String[] args){
        int number = 17, flag =0;
        if (number<=1) {
            System.out.println("The number is neither prime nor composite");
            return;
        }
        for (int i = 2;i < number/2 ;i++ ) {
            if (number%i == 0) {
                flag++;
                break;
            }
        }
        if (flag == 1){
            System.out.println(number+" is not a prime number.");
        }
        else{
            System.out.println(number + " is a prime number.");
        }
    }
}
```

1.4 Sample Output



```
E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_officia
l> e: && cd "e:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cy
cle_1_official" && cmd /C ""c:\Program Files\Java\jdk-17.0.1\bin\java.
exe" --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\U
sers\vinod\AppData\Roaming\Code\User\workspaceStorage\9dd9b71a436977c4
c5cd1cd090876b1a\redhat.java\jdt_ws\Java_cycle_1_official_704f625c\bin
 Prime "

17 is a prime number.

E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_officia
l>
```

2 **Palindrome**

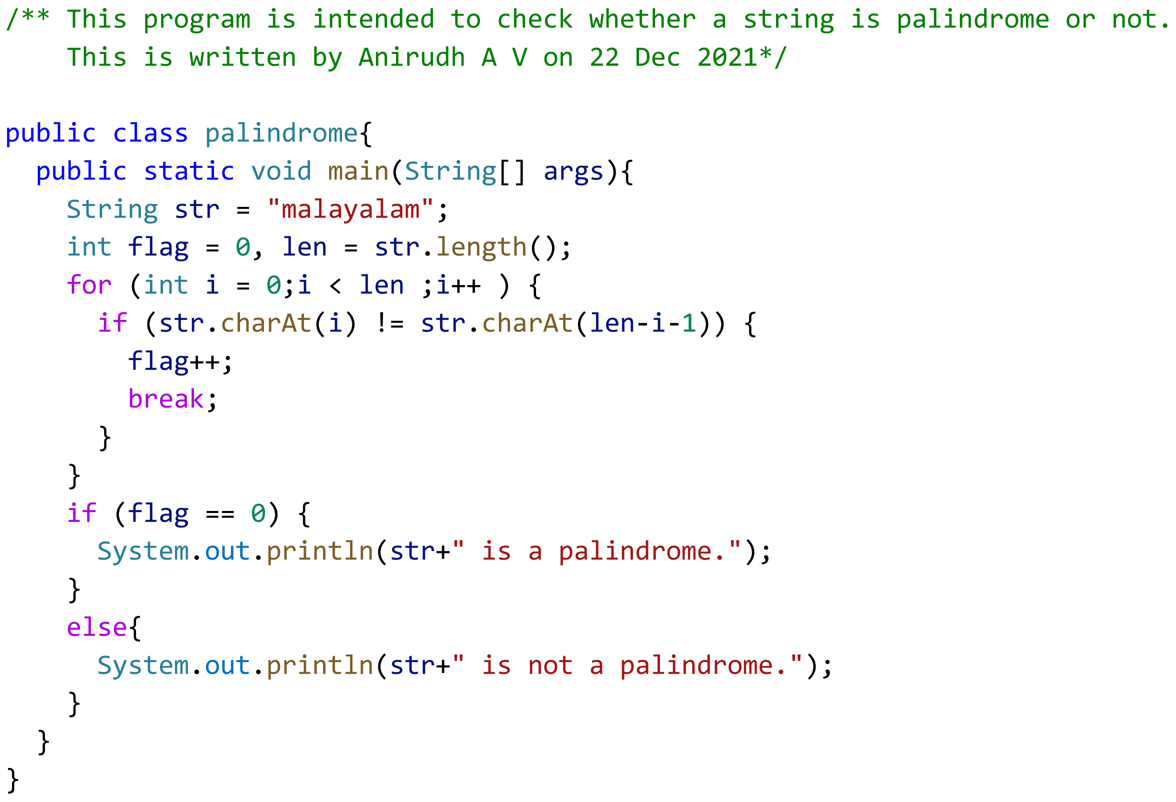
2.1 Aim

To Implement a Java program to check whether the given string is a palindrome or not.

2.2 Algorithm

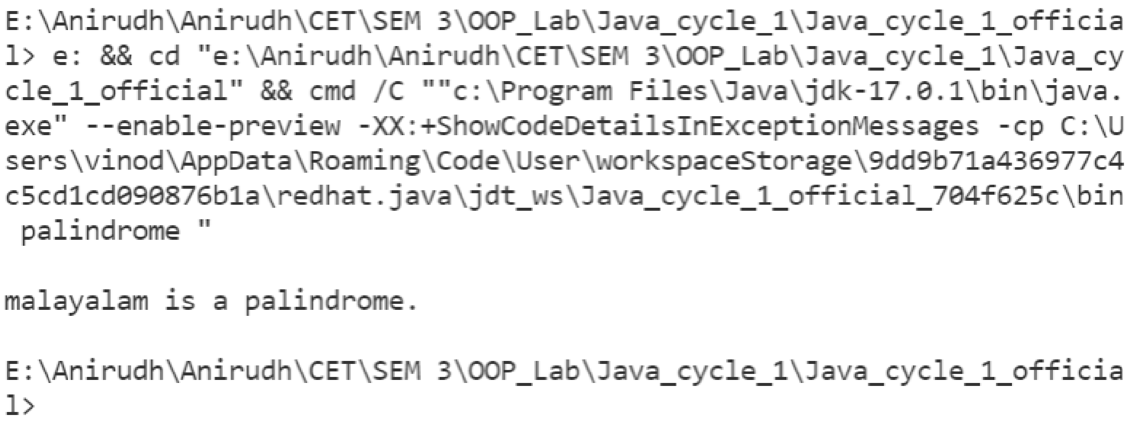
```
Step - 1   Start
Step - 2   Declare and initialize string [] = “malayalam”, flag = 0, i = 0, and len as the length of string [].
Step - 3   Repeat the following steps until i < len/2.
    Step - 3.1.   If string [i] != string [len - i], increment flag and go to step 6.
    Step - 3.2.   Increment i.
Step - 4   If flag != 0, then Display “The entered word is not a palindrome”. Else, Display “The entered word is a palindrome”.
Step - 5   Stop
```

2.3 Code



```java
/** This program is intended to check whether a string is palindrome or not.
    This is written by Anirudh A V on 22 Dec 2021*/

public class palindrome{
  public static void main(String[] args){
    String str = "malayalam";
    int flag = 0, len = str.length();
    for (int i = 0;i < len ;i++ ) {
      if (str.charAt(i) != str.charAt(len-i-1)) {
        flag++;
        break;
      }
    }
    if (flag == 0) {
      System.out.println(str+" is a palindrome.");
    }
    else{
      System.out.println(str+" is not a palindrome.");
    }
  }
}
```

2.4 Sample Output



```
E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_officia
l> e: && cd "e:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cy
cle_1_official" && cmd /C ""c:\Program Files\Java\jdk-17.0.1\bin\java.
exe" --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\U
sers\vinod\AppData\Roaming\Code\User\workspaceStorage\9dd9b71a436977c4
c5cd1cd090876b1a\redhat.java\jdt_ws\Java_cycle_1_official_704f625c\bin
 palindrome "

malayalam is a palindrome.

E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_officia
l>
```

3 **Occurrence Of a Character in A String**

3.1 Aim

To Implement a Java program to determine the frequency or occurrence of characters in a given sentence.

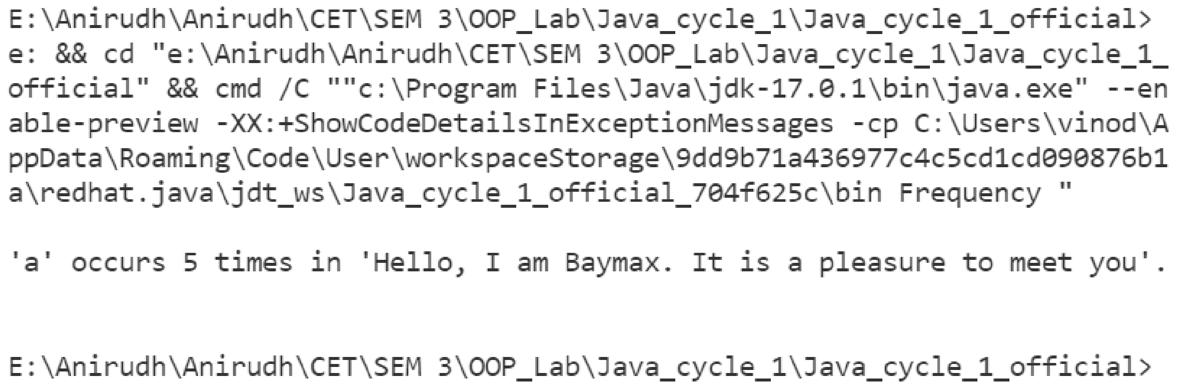
3.2 Algorithm

```
Step - 1   Start
Step - 2   Declare and initialize a String str, and variables i = 0, count = 0,
 key = ‘a’ and len as the length of str.
Step - 3   Repeat the following steps until i = len.
   Step - 3.1.      If str[i] = key, increment count.
   Step - 3.2.      Increment i.
Step - 4   Display count as the number of occurrences of key in str.
Step - 5   Stop
```

3.3 Code

```
/**
 * This program is intended to determine the occurrence or frequency of a
 * character in a given sentence.
 * This is written by Anirudh A V on 22 Dec 2021
 */

public class Frequency {
  public static void main(String[] args){
    String str = "Hello, I am Baymax. It is a pleasure to meet you";
    char key = 'a';
    int len = str.length(), count = 0;
    for (int i = 0;i < len ;i++ ) {
      if (str.charAt(i) == key) {
        count++;
      }
    }
    System.out.println(key + " occurs " + count + " times in '" + str + "'.");
  }
}
```

3.4 Sample Output



```
E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_official>
e: && cd "e:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_
official" && cmd /C ""c:\Program Files\Java\jdk-17.0.1\bin\java.exe" --en
able-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\vinod\A
ppData\Roaming\Code\User\workspaceStorage\9dd9b71a436977c4c5cd1cd090876b1
a\redhat.java\jdt_ws\Java_cycle_1_official_704f625c\bin Frequency "

'a' occurs 5 times in 'Hello, I am Baymax. It is a pleasure to meet you'.


E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_official>
```

4 Reversing a String

4.1 Aim

To implement a Java program to reverse a string.

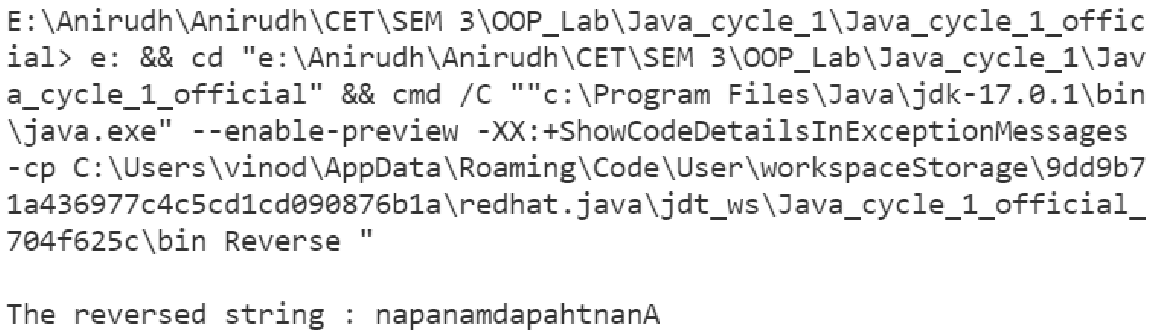
4.2 Algorithm

```
Step - 1   Start
Step - 2   Declare and initialize String str, variables i = 0 and len as the length of str.
Step - 3   Declare a character array rev [] of size len + 1.
Step - 4   Repeat the following steps until i = len.
    Step - 4.1.     rev [i] = str [len - 1 – i]
    Step - 4.2.     Increment i.
Step - 5   Assign rev [len] = ‘\0’.
Step - 6   Print rev
Step - 7   Stop
```

4.3 Code

```java
/**
 * This program is intended to reverse a given string.
 * This is written by Anirudh A V on 22 Dec 2021
 */

public class Reverse{
  public static void main(String[] args){
    String str = "Ananthapadmanapan";
    int len = str.length(),i =0;
    char[] rev = new char[len+1];

    for (i = 0;i < len ;i++ ) {
      rev[i] = str.charAt(len-1-i);
    }
    rev[len] ='\0';
    System.out.printf("The reversed string : ");
    System.out.println(rev);
  }
}
```

4.4 Sample Output



```
E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_offic
ial> e: && cd "e:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Jav
a_cycle_1_official" && cmd /C ""c:\Program Files\Java\jdk-17.0.1\bin
\java.exe" --enable-preview -XX:+ShowCodeDetailsInExceptionMessages
-cp C:\Users\vinod\AppData\Roaming\Code\User\workspaceStorage\9dd9b7
1a436977c4c5cd1cd090876b1a\redhat.java\jdt_ws\Java_cycle_1_official_
704f625c\bin Reverse "

The reversed string : napanamdapahtnanA
```

5 Second Smallest Element of An Array

5.1 Aim

To implement a Java program to find the second smallest element in an array.

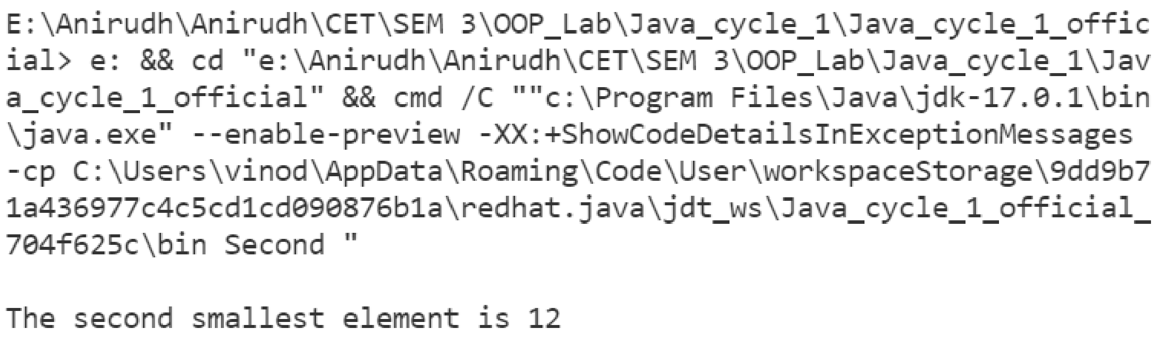
5.2 Algorithm

```
Step - 1   Start
Step - 2   Declare and initialize an array and, variables i = 0, and len as the length of the array.
Step - 3   Repeat the following steps until i = 0.
  Step - 3.1.      Initialize j = i + 1, repeat the following steps until j = len.
    Step - 3.1.1.  If array [i] > array [j], swap array [i] & array [j].
    Step - 3.1.2.  Increment j.
  Step - 3.2.      Increment i.
Step - 4   Display array [1] as the second smallest element.
Step - 5   Stop
```

5.3 Code

```
/**
 * This program is intended to find the second smallest element
 * in an array.
 * This is written by Anirudh A V on 22 Dec 2021
 */

public class Second{
  public static void main(String[] args){
    int[] array = {12, 23, 1, 56, 789, 24};
    int len = 6, temp;
    for (int i = 0;i < len ;i++ ) {
      for (int j= i+1;j<len ;j++ ) {
        if (array[i]>array[j]) {
          temp = array[i];
          array[i] = array[j];
          array[j] = temp;
        }
      }
    }
    System.out.println("The second smallest element is "+ array[1]);
  }
}
```

5.4 Sample Output



```
E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_offic
ial> e: && cd "e:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Jav
a_cycle_1_official" && cmd /C ""c:\Program Files\Java\jdk-17.0.1\bin
\java.exe" --enable-preview -XX:+ShowCodeDetailsInExceptionMessages 
-cp C:\Users\vinod\AppData\Roaming\Code\User\workspaceStorage\9dd9b7
1a436977c4c5cd1cd090876b1a\redhat.java\jdt_ws\Java_cycle_1_official_
704f625c\bin Second "

The second smallest element is 12
```

6 Multiplication Of Two Matrices

6.1 Aim

To implement a Java program to multiply two matrices and display their product.

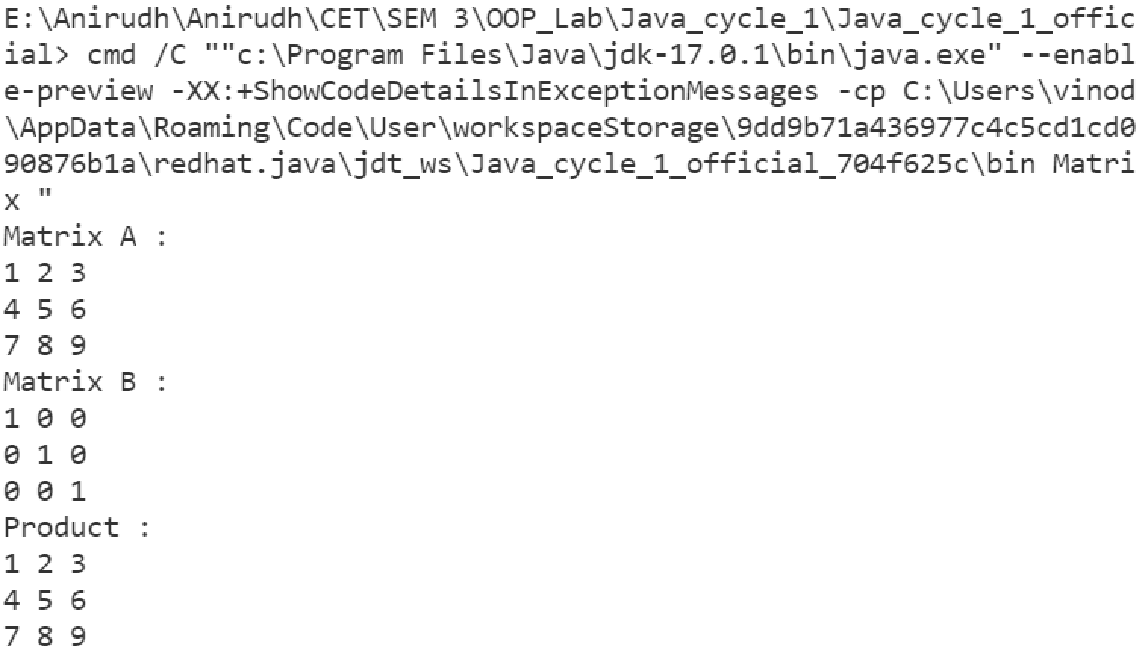
6.2 Algorithm

```
Step - 1   Start
Step - 2   Declare and initialize 2D integer arrays MatrixA, MatrixB and mult. And i = 0, j = 0, k = 0.
Step - 3   Check if the matrix can be multiplied or not, if column of MatrixA is not equal to row of MatrixB, matrices can't be multiplied and an error message is generated.
Step - 4   Repeat until i < row of MatrixA
  Step - 4.1.     Repeat until j < column of MatrixB
    Step - 4.1.1. Initialize mult[i][j] = 0.
    Step - 4.1.2. Repeat until k < row of MatrixA
      Step - 4.1.2.1.   Set mult[i][j] = mult[i][j] + MatrixA[i][k] * MatrixB[k][j]
      Step - 4.1.2.2.   Increment k.
    Step - 4.1.3.   Increment j.
  Step - 4.2.     Increment i
Step - 5   Display Mult as the required matrix.
Step - 6   Stop
```

6.3 Code

```java
/**
 * This program is intended to multiply two matrices and
 * Display their product.
 * This is written by Anirudh A V on 22 Dec 2021
 */

public class Matrix{

  public static void display(int[][] matrix, int row, int col){
    for (int i = 0;i < row ;i++ ) {
      for (int j = 0;j < col ;j++ ) {
        System.out.printf(matrix[i][j] + " ");
      }
      System.out.println("");
    }
  }

  public static void main(String[] args){
    int[][] matrixA = {{1,2,3},{4,5,6},{7,8,9}};
    int[][] matrixB = {{1,0,0},{0,1,0},{0,0,1}};
    int[][] mult = new int[3][3];

    for (int i = 0;i < 3 ;i++ ) {
      for (int j = 0;j < 3 ;j++ ) {
        mult[i][j] = 0;
        for (int k = 0;k < 3 ;k++ ) {
          mult[i][j] = mult[i][j] + matrixA[i][k]*matrixB[k][j];
        }
      }
    }
    System.out.println("Matrix A : ");
    display(matrixA, 3, 3);
    System.out.println("Matrix B : ");
    display(matrixB, 3, 3);
    System.out.println("Product : ");
    display(mult, 3, 3);
  }
}
```

6.4 Sample Output



```
E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_offic
ial> cmd /C ""c:\Program Files\Java\jdk-17.0.1\bin\java.exe" --enabl
e-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\vinod
\AppData\Roaming\Code\User\workspaceStorage\9dd9b71a436977c4c5cd1cd0
90876b1a\redhat.java\jdt_ws\Java_cycle_1_official_704f625c\bin Matri
x "
Matrix A :
1 2 3
4 5 6
7 8 9
Matrix B :
1 0 0
0 1 0
0 0 1
Product :
1 2 3
4 5 6
7 8 9
```

7 Transpose Of a Matrix

7.1 Aim

To implement a Java program to find the transpose of a given matrix.

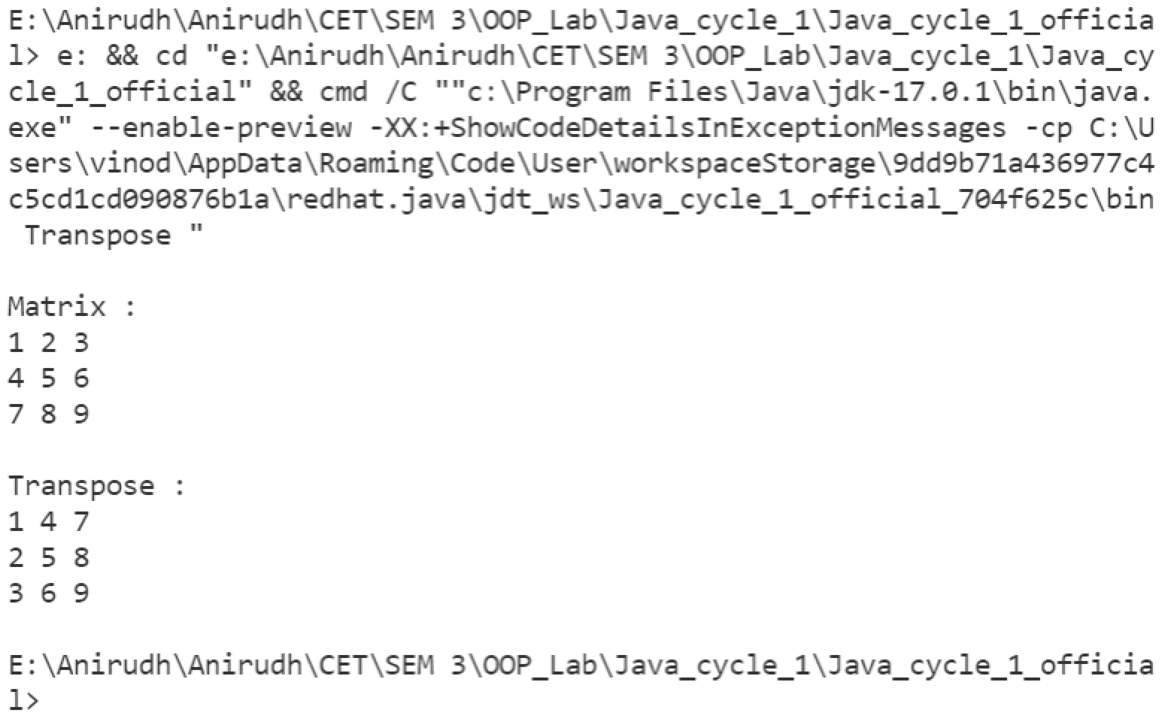
7.2 Algorithm

```
Step - 1   Start
Step - 2   Declare and initialize a two-dimensional array matrix. Initialize variables i = 0, j = 0, row and col as the number of rows and columns of the matrix.
Step - 3   Print the elements of the matrix.
Step - 4   Repeat until i = row
   Step - 4.1.     Repeat until j = col
      Step - 4.1.1.    Print matrix[j][i]
      Step - 4.1.2.    Increment j.
   Step - 4.2.     Print “\n”
   Step - 4.3.     Increment i.
Step - 5   Stop
```

7.3 Code

```
/**
 * This program is intended to find the transpose of a given matrix.
 * This is written by Anirudh A V on 22 Dec 2021
 */

public class Transpose{

  public static void display(int[][] matrix, int row, int col){
    for (int i = 0;i < row ;i++ ) {
      for (int j = 0;j < col ;j++ ) {
        System.out.printf(matrix[i][j] + " ");
      }
      System.out.println("");
    }
  }

  public static void main(String[] args){
    int[][] matrix = {{1,2,3},{4,5,6},{7,8,9}};
    System.out.println("\nMatrix : ");
    display(matrix, 3, 3);
    System.out.println("\nTranspose : ");
    for (int i = 0;i < 3 ;i++ ) {
      for (int j = 0;j < 3 ;j++ ) {
        System.out.printf(matrix[j][i] + " ");
      }
      System.out.println("");
    }
  }
}
```

7.4 Sample Output



```
E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_officia
l> e: && cd "e:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cy
cle_1_official" && cmd /C ""c:\Program Files\Java\jdk-17.0.1\bin\java.
exe" --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\U
sers\vinod\AppData\Roaming\Code\User\workspaceStorage\9dd9b71a436977c4
c5cd1cd090876b1a\redhat.java\jdt_ws\Java_cycle_1_official_704f625c\bin
 Transpose "

Matrix :
1 2 3
4 5 6
7 8 9

Transpose :
1 4 7
2 5 8
3 6 9

E:\Anirudh\Anirudh\CET\SEM 3\OOP_Lab\Java_cycle_1\Java_cycle_1_officia
l>
```