

21BCE7371

RADHA KRISHNA GARG

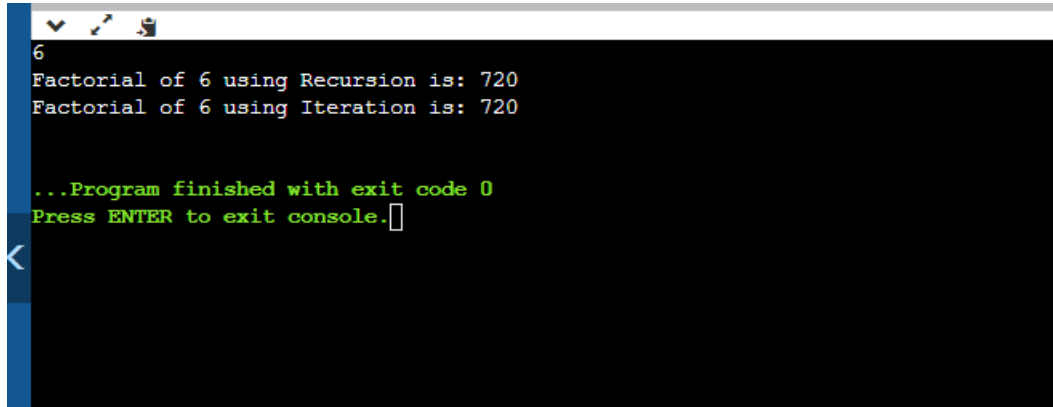
DSA LAB ASSIGNMENT-1

1. Write a Program to find the factorial of a given number using recursion and analyze the time complexity.

#### INPUT

```
1 // Java program to find factorial of given number
2 import java.util.*;
3 class Main {
4
5     // ----- Recursion -----
6     // method to find factorial of given number
7     static int factorialUsingRecursion(int n)
8     {
9         if (n == 0)
10            return 1;
11
12        // recursion call
13        return n * factorialUsingRecursion(n - 1);
14    }
15
16    // ----- Iteration -----
17    // Method to find the factorial of a given number
18    static int factorialUsingIteration(int n)
19    {
20        int fact = 1, i;
21
22        // using iteration
23        for (i = 2; i <= n; i++)
24            fact *= i;
25
26        return fact;
27    }
28
29    // Driver method
30    public static void main(String[] args)
31    {
32        Scanner sc = new Scanner(System.in);
33        int num = sc.nextInt();
34        //int num = 5;
35        System.out.println("Factorial of " + num
36                           + " using Recursion is: "
37                           + factorialUsingRecursion(num));
38
39
40        System.out.println("Factorial of " + num
41                           + " using Iteration is: "
42                           + factorialUsingIteration(num));
43    }
44 }
```

## OUTPUT

A screenshot of a Java IDE's console window. The window has a dark background with light blue and green text. The output shows the number '6' on the first line, followed by 'Factorial of 6 using Recursion is: 720' and 'Factorial of 6 using Iteration is: 720' on the next two lines. The third line says '...Program finished with exit code 0' and the fourth line says 'Press ENTER to exit console.' with a cursor. The IDE's toolbar is visible at the top of the console window.

```
6
Factorial of 6 using Recursion is: 720
Factorial of 6 using Iteration is: 720

...Program finished with exit code 0
Press ENTER to exit console.
```

// Java program to find factorial of given number

```
import java.util.*;
```

```
class Main {
```

```
    // ----- Recursion -----
```

```
    // method to find factorial of given number
```

```
    static int factorialUsingRecursion(int n)
```

```
    {
```

```
        if (n == 0)
```

```
            return 1;
```

```
        // recursion call
```

```
        return n * factorialUsingRecursion(n - 1);
```

```
    }
```

```
    // ----- Iteration -----
```

```
    // Method to find the factorial of a given number
```

```
    static int factorialUsingIteration(int n)
```

```
    {
```

```
        int fact = 1, i;
```

```

        // using iteration
        for (i = 2; i <= n; i++)
            fact *= i;

        return fact;
    }

    // Driver method
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int num = sc.nextInt();
        //int num = 5;
        System.out.println("Factorial of " + num
                           + " using Recursion is: "
                           + factorialUsingRecursion(num));

        System.out.println("Factorial of " + num
                           + " using Iteration is: "
                           + factorialUsingIteration(num));
    }
}

```

## TIME COMPLEXITY

$T(n) = T(n-1) + 3$  (3 is for as we must do three constant operations like

multiplication, and checking the value of  $n$  in each recursive call)

$$= T(n-2) + 6 \text{ (Second recursive call)}$$

$$= T(n-3) + 9 \text{ (Third recursive call)}$$

.

.

.

.

$$= T(n-k) + 3k$$

till,  $k = n$

Then,

$$= T(n-n) + 3n$$

$$= T(0) + 3n$$

$$= 1 + 3n$$

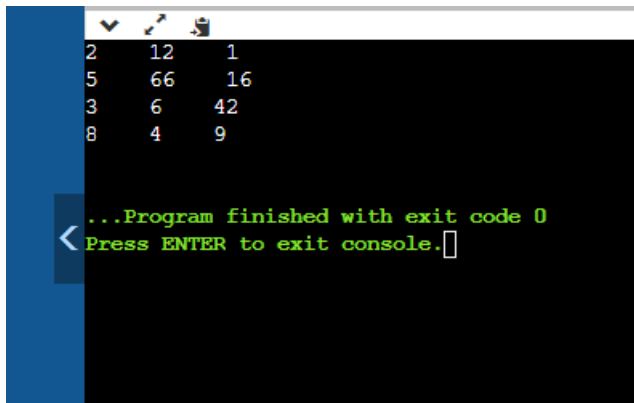
The time complexity of recursive factorial is  $O(n)$

Write a Program to find the transpose of a given matrix and display its time complexity.  
Challenging

INPUT

```
1 class Main
2 {
3     public static void main (String[] args)
4     {
5         // given matrix M whose transpose is to be found
6         int M[][] = {{2,5,3,8},
7                     {12,66,6,4},
8                     {1,16,42,9}};
9
10        // find number of rows and columns in matrix M
11        int n = M.length, m = M[0].length;
12
13        //create empty transpose matrix of size m*n
14        int M_transpose[][] = new int[m][n];
15
16        //traverse matrix M
17        for(int i=0;i<n;i++){
18            for(int j=0;j<m;j++){
19
20                //assign M_transpose[j][i] as M[i][j]
21                M_transpose[j][i] = M[i][j];
22            }
23        }
24
25        // output the transpose matrix
26        for(int i=0;i<m;i++){
27            for(int j=0;j<n;j++){
28                System.out.print(M_transpose[i][j] + " ");
29            }
30            System.out.println();
31        }
32    }
33 }
34 }
```

## OUTPUT



```
2    12    1
5    66    16
3     6    42
8     4     9

...Program finished with exit code 0
Press ENTER to exit console.
```

```
class Main
{
    public static void main (String[] args)
    {
        // given matrix M whose transpose is to be found
        int M[][] = {{2,5,3,8},
                     {12,66,6,4},
                     {1,16,42,9}};

        // find number of rows and columns in matrix M
        int n = M.length, m = M[0].length;

        //create empty transpose matrix of size m*n
        int M_transpose[][] = new int[m][n];

        //traverse matrix M
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){

                //assign M_transpose[j][i] as M[i][j]
                M_transpose[j][i] = M[i][j];
            }
        }

        // output the transpose matrix
        for(int i=0;i<m;i++){
            for(int j=0;j<n;j++){
                System.out.print(M_transpose[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

```
}  
}
```

## TIME COMPLEXITY

Time complexity is  $O(n*m)$

N – number of rows

M—number of columns

- 3 Write a Program to illustrate the difference between recursion and iteration by giving its time complexities.

```

1 // Java program to find factorial of given number
2 import java.util.*;
3 class Main {
4
5     // ----- Recursion -----
6     // method to find factorial of given number
7     static int factorialUsingRecursion(int n)
8     {
9         if (n == 0)
10            return 1;
11
12        // recursion call
13        return n * factorialUsingRecursion(n - 1);
14    }
15
16    // ----- Iteration -----
17    // Method to find the factorial of a given number
18    static int factorialUsingIteration(int n)
19    {
20        int fact = 1, i;
21
22        // using iteration
23        for (i = 2; i <= n; i++)
24            fact *= i;
25
26        return fact;
27    }
28
29    // Driver method
30    public static void main(String[] args)
31    {
32        Scanner sc = new Scanner(System.in);
33        int num = sc.nextInt();
34        //int num = 5;
35        System.out.println("Factorial of " + num
36                           + " using Recursion is: "
37                           + factorialUsingRecursion(num));
38
39
40        System.out.println("Factorial of " + num
41                           + " using Iteration is: "
42                           + factorialUsingIteration(num));
43    }
44 }

```

OUTPUT

```

6
Factorial of 6 using Recursion is: 720
Factorial of 6 using Iteration is: 720

...Program finished with exit code 0
Press ENTER to exit console.

```



```
// Java program to find factorial of given number

import java.util.*;

class Main {

    // ----- Recursion -----
    // method to find factorial of given number
    static int factorialUsingRecursion(int n)
    {
        if (n == 0)
            return 1;

        // recursion call
        return n * factorialUsingRecursion(n - 1);
    }

    // ----- Iteration -----
    // Method to find the factorial of a given number
    static int factorialUsingIteration(int n)
    {
        int fact = 1, i;

        // using iteration
        for (i = 2; i <= n; i++)
            fact *= i;

        return fact;
    }

    // Driver method
```

```

public static void main(String[] args)
{
    Scanner sc = new Scanner(System.in);
    int num = sc.nextInt();
    //int num = 5;
    System.out.println("Factorial of " + num
                        + " using Recursion is: "
                        + factorialUsingRecursion(num));

    System.out.println("Factorial of " + num
                        + " using Iteration is: "
                        + factorialUsingIteration(num));
}
}

```

## TIME COMPLEXITY

$T(n) = T(n-1) + 3$  (3 is for as we must do three constant operations like multiplication, and checking the value of n in each recursive call)

$= T(n-2) + 6$  (Second recursive call)

$= T(n-3) + 9$  (Third recursive call)

.  
.   
.   
.

$$= T(n-k) + 3k$$

till,  $k = n$

Then,

$$= T(n-n) + 3n$$

$$= T(0) + 3n$$

$$= 1 + 3n$$

The time complexity of recursive factorial is  $O(n)$