

CSA0976 Java Programming

Name: B V HARIKA

Reg no: 192111512

Assignment 3

1.

Program:

```
import java.awt.Color;

import java.awt.Font;

import java.awt.Graphics;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.Random;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.Timer;
```

```
public class DynamicTextColor extends JPanel implements
ActionListener {

private static final long serialVersionUID = 1L;

private final int DELAY = 50;

private final Timer timer;

private final Random random;

private Color color;

private Font font;

private String message;


public DynamicTextColor() {

setDoubleBuffered(true);

timer = new Timer(DELAY, this);

random = new Random();

font = new Font("Arial", Font.BOLD, 36);

message = "Dynamic Text Color";

timer.start();
```

```
}
```

```
@Override
```

```
public void paintComponent(Graphics g) {
```

```
    super.paintComponent(g);
```

```
    g.setFont(font);
```

```
    g.setColor(color);
```

```
    g.drawString(message, 10, 50);
```

```
}
```

```
@Override
```

```
public void actionPerformed(ActionEvent e) {
```

```
    color = new Color(random.nextInt(256),
```

```
    random.nextInt(256), random.nextInt(256));
```

```
    repaint();
```

```
}
```

```
public static void main(String[] args) {  
  
    JFrame frame = new JFrame("Dynamic Text Color");  
  
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
    frame.setSize(400, 100);  
  
    DynamicTextColor panel = new DynamicTextColor();  
  
    frame.add(panel);  
  
    frame.setVisible(true);  
  
    }  
  
}
```

Output:



2}

Program:

```
public class MultiplicationTableThread extends Thread {  
    private int number;  
  
    public MultiplicationTableThread(int number) {  
        this.number = number;  
    }  
  
    public void run() {  
        System.out.println("Multiplication table for " + number);  
        for (int i = 1; i <= 10; i++) {  
            System.out.println(number + " x " + i + " = " + (number  
* i));  
            try {  
                Thread.sleep(100);  
            } catch (InterruptedException e) {  
                e.printStackTrace();  
            }  
        }  
    }  
  
    public static void main(String[] args) {
```

```
MultiplicationTableThread thread1 = new  
MultiplicationTableThread(5);
```

```
MultiplicationTableThread thread2 = new  
MultiplicationTableThread(10);
```

```
thread1.start();
```

```
thread2.start();
```

```
try {
```

```
    thread1.join();
```

```
    thread2.join();
```

```
} catch (InterruptedException e) {
```

```
    e.printStackTrace();
```

```
}
```

```
}
```

```
}
```

```
^ java -cp /tmp/VHiFCgDEdY MultiplicationTableThread
Multiplication table for 5
Multiplication table for 10
10 x 1 = 10
5 x 1 = 5
10 x 2 = 20
5 x 2 = 10
10 x 3 = 30
5 x 3 = 15 10 x 4 = 40
5 x 4 = 20
10 x 5 = 50
5 x 5 = 25
10 x 6 = 60
5 x 6 = 30
10 x 7 = 70
▼ 5 x 7 = 35
```

3}

Program:

```
import java.util.Scanner;

public class Fibonacci
{
    public static void main(String[] args)
    {
        int n, a = 0, b = 0, c = 1;
        Scanner s = new Scanner(System.in);
        System.out.print("Enter value of n:");
        n = s.nextInt();
        System.out.print("Fibonacci Series:");
        for(int i = 1; i <= n; i++)
```

```

    {
        a = b;
        b = c;
        c = a + b;
        System.out.print(a+" ");
    }
}

```

Output:

```

java -cp /tmp/Q6nrgMHi5Q Fibonacci
Enter value of n:12
Fibonacci Series:0 1 1 2 3 5 8 13 21 34 55 89 |

```

4}

Program:

```

public class Solution {
    public boolean isUgly(int n) {

```



```
    if (n <= 0) {  
        return false;  
    }  
  
    while (n % 2 == 0) {  
        n /= 2;  
    }  
    while (n % 3 == 0) {  
        n /= 3;  
    }  
    while (n % 5 == 0) {  
        n /= 5;  
    }  
  
    return n == 1;  
}  
}
```

```
java -cp /tmp/Q6nrgMHi5Q GFG  
150th ugly no. is 5832|
```

5}

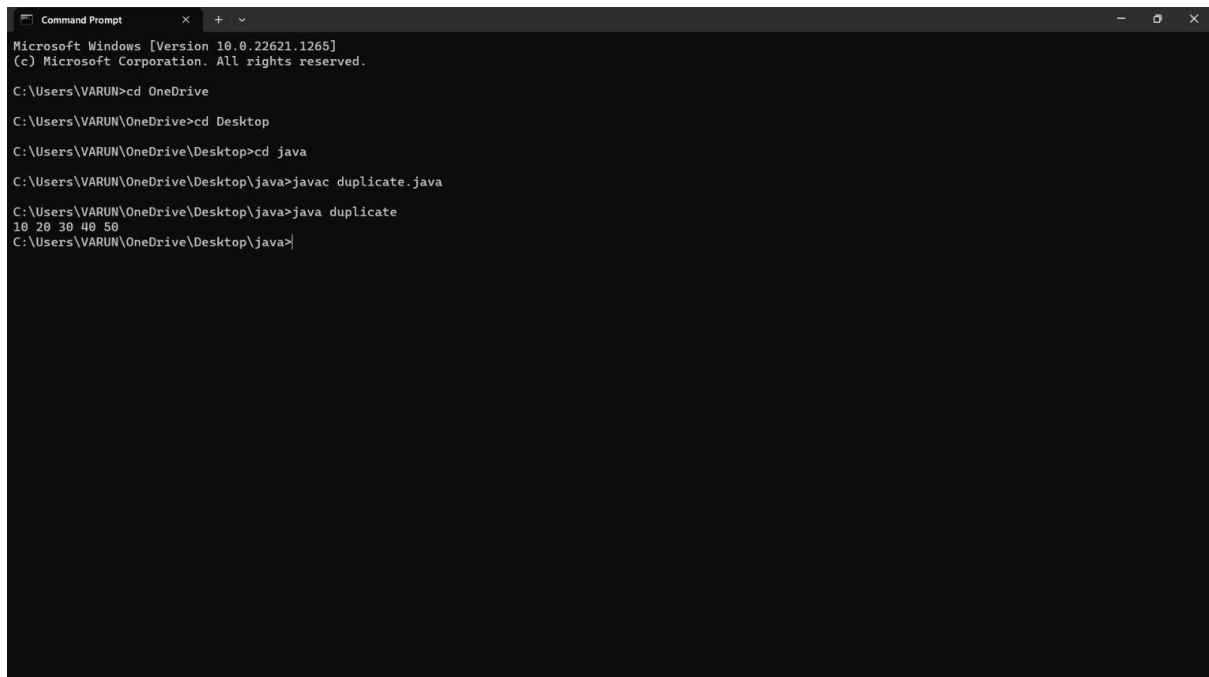
Program:

```
class duplicate
{
    // Function to remove duplicate elements
    // This function returns new size of modified
    // array.
    static int removeDuplicates(int arr[], int n)
    {
        // Return, if array is empty
        // or contains a single element
        if (n==0 || n==1)
            return n;
        int[] temp = new int[n];
        // Start traversing elements
        int j = 0;
        for (int i=0; i<n-1; i++)
            // If current element is not equal
            // to next element then store that
            // current element
            if (arr[i] != arr[i+1])
                temp[j++] = arr[i];
        // Store the last element as whether
```

```
// it is unique or repeated, it hasn't
// stored previously
temp[j++] = arr[n-1];
class duplicate
{
// Function to remove duplicate elements
// This function returns new size of modified
// array.
static int removeDuplicates(int arr[], int n)
{
// Return, if array is empty
// or contains a single element
if (n==0 || n==1)
return n;
int[] temp = new int[n];
// Start traversing elements
int j = 0;
for (int i=0; i<n-1; i++)
// If current element is not equal
// to next element then store that
// current element
if (arr[i] != arr[i+1])
```

```
temp[j++] = arr[i];  
  
// Store the last element as whether  
// it is unique or repeated, it hasn't  
// stored previously  
temp[j++] = arr[n-1];
```

output:



```
Command Prompt  
Microsoft Windows [Version 10.0.22621.1265]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\VARUN>cd OneDrive  
C:\Users\VARUN\OneDrive>cd Desktop  
C:\Users\VARUN\OneDrive\Desktop>cd java  
C:\Users\VARUN\OneDrive\Desktop\java>javac duplicate.java  
C:\Users\VARUN\OneDrive\Desktop\java>java duplicate  
10 20 30 40 50  
C:\Users\VARUN\OneDrive\Desktop\java>
```