COMPUTER NETWORKS AND SECURITY LABORATORY

Assignment No. 6 B

NAME :- OJUS P. JAISWAL

ROLL NO. :- TACO19108

YEAR AND DIV :- TE A

Ques :- Write a program to demonstrate Sub-netting and find subnet masks.

Solution :-

Program :

1) Subnet1 =

```
/*Finds the class of IP Address. Network Address and broadcast address is displayed*/
import java.io.*;
import java.net.InetAddress;
public class Subnet1 {

    public static void main(String[] args) throws IOException {

        System.out.println("ENTER IP:");
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        String ip = br.readLine();
        String checkclass = ip.substring(0, 3);



        int cc = Integer.parseInt(checkclass);
        String mask = null;
```

```
if(cc>0)
{
   if(cc<=127)
   {
      mask = "255.0.0.0";
System.out.println("Class A IP Address");
System.out.println("SUBNET MASK:\n"+mask);
   }
   if(cc>=128 && cc<=191)
   {
      mask = "255.255.0.0";
System.out.println("Class B IP Address");
System.out.println("SUBNET MASK:\n"+mask);
   }
   if(cc>=192 && cc<=223)
   {
      mask = "255.255.255.0";
System.out.println("Class C IP Address");
System.out.println("SUBNET MASK:\n"+mask);
   }
if(cc>=224 && cc<=239)
   {
mask = "255.0.0.0";
      System.out.println("Class D IP Address Used for multicasting");
      }
      if(cc>=240 && cc<=254)
      {
   mask = "255.0.0.0";
         System.out.println("Class E IP Address Experimental Use");
      }
   }


   String networkAddr="";
String lastAddr="";
   String[] ipAddrParts=ip.split("\\.");
   String[] maskParts=mask.split("\\.");

   for(int i=0;i<4;i++){
   int x=Integer.parseInt(ipAddrParts[i]);
   int y=Integer.parseInt(maskParts[i]);
   int z=x&y;
   networkAddr+=z+".";
int w=z|(y^255);
lastAddr+=w+".";
   }

System.out.println("First IP of block/ Network Address: "+networkAddr);
```

```
    System.out.println("Last IP of block/Broadcast Address: "+lastAddr);
   }


}
```

2) Subnet =

```
/* Subnet Addressing. Finding the number of 1's in Subnet Mask */

import java.util.Scanner;
class Subnet{
public static void main(String args[])
{
Scanner sc = new Scanner(System.in);
System.out.print("Enter the ip address: ");
String ip = sc.nextLine();
String checkclass = ip.substring(0, 3);
String split_ip[] = ip.split("\\."); //SPlit the string after every .
String split_bip[] = new String[4]; //split binary ip
String bip = "";
for(int i=0;i<4;i++)
{
split_bip[i] = appendZeros(Integer.toBinaryString(Integer.parseInt(split_ip[i]))); // "18" => 18 => 10010 => 00010010
bip += split_bip[i];
}
System.out.println("IP in binary is "+bip);
System.out.print("Enter the number of Subnets: ");
int n = sc.nextInt();

//Calculation of mask
```

```
int bits = (int)Math.ceil(Math.log(n)/Math.log(2)); /*eg if address = 120, log
120/log 2 gives log to the base 2 => 6.9068, ceil gives us upper integer */
System.out.println("Number of bits required for address = "+bits);
int mask;

int cc = Integer.parseInt(checkclass);
   String maskc = null;
          if(cc>0)
    {
      if(cc<=127)
      {
        maskc = "255.0.0.0";
                      mask = 8+bits;
                      System.out.println("Class A IP Address");
    System.out.println("SUBNET MASK:\n"+maskc);
           System.out.println("Number of 1's in subnet mask = "+mask);
      }
      if(cc>=128 && cc<=191)
      {
        maskc = "255.255.0.0";
                      mask = 16+bits;
    System.out.println("Class B IP Address");
    System.out.println("SUBNET MASK:\n"+maskc);
           System.out.println("Number of 1's in subnet mask = "+mask);
      }
```

```
            if(cc>=192 && cc<=223)
            {
                maskc = "255.255.255.0";
                                              mask = 24+bits;
        System.out.println("Class C IP Address");
        System.out.println("SUBNET MASK:\n"+maskc);
                       System.out.println("Number of 1's in subnet mask = "+mask);
            }

        }

}
static String appendZeros(String s)
{
String temp = new String("00000000");
return temp.substring(s.length())+ s;
}
}
```

3) CIDR =

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package calcuator;

import javax.swing.JOptionPane;

/**
 *
 * @author Ankwabe
 */
public class MainCalculator extends javax.swing.JFrame {

    /**
     * Creates new form MainCalculator
     */
    public MainCalculator() {
        initComponents();
        setTitle("SubNet Calculator");
    }

    /**
```

```
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        oct1 = new javax.swing.JTextField();
        oct2 = new javax.swing.JTextField();
        oct3 = new javax.swing.JTextField();
        oct4 = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        cidrValue = new javax.swing.JTextField();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        networkClass = new javax.swing.JTextField();
        subnetMask = new javax.swing.JTextField();
        networkAddress = new javax.swing.JTextField();
```

```
broadcastAddress = new javax.swing.JTextField();
noSubnets = new javax.swing.JTextField();
hostsPSubnet = new javax.swing.JTextField();
calBtn = new javax.swing.JButton();
resetBtn = new javax.swing.JButton();
jLabel9 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

jLabel1.setText("IP Address");

jLabel2.setText("CIDR values        /");

jLabel3.setText("Subnet Mask");

jLabel4.setText("Network Address");

jLabel5.setText("Broadcast Address");

jLabel6.setText("Number of Subnets");

jLabel7.setText("Hosts Per Subnet");

jLabel8.setText("Network Class");
```

```
networkClass.setEditable(false);

subnetMask.setEditable(false);

networkAddress.setEditable(false);

broadcastAddress.setEditable(false);

calBtn.setText("Calculate");
calBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        calBtnActionPerformed(evt);
    }
});

resetBtn.setText("Reset");
resetBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        resetBtnActionPerformed(evt);
    }
});

jLabel9.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
```

```
        jLabel9.setText("SubNet Calculator");


        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
          layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
          .addGroup(layout.createSequentialGroup()
            .addGap(57, 57, 57)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
              .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILI
NG, false)
                  .addComponent(jLabel4,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                  .addComponent(jLabel3,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                  .addComponent(jLabel5,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
```

```
                        .addComponent(jLabel6,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .addComponent(jLabel7,
javax.swing.GroupLayout.PREFERRED_SIZE, 107,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG, false)
                        .addComponent(subnetMask)
                        .addComponent(networkAddress)
                        .addComponent(broadcastAddress)
                        .addComponent(noSubnets)
                        .addComponent(hostsPSubnet,
javax.swing.GroupLayout.PREFERRED_SIZE, 139,
javax.swing.GroupLayout.PREFERRED_SIZE))
                     .addGap(48, 48, 48)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(calBtn,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(73, 73, 73)
```

```
                .addComponent(resetBtn,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE))
              .addGroup(layout.createSequentialGroup()
                .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 97,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(networkClass,
javax.swing.GroupLayout.PREFERRED_SIZE, 40,
javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addContainerGap())
          .addGroup(layout.createSequentialGroup()
            .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE,
80, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(oct1, javax.swing.GroupLayout.PREFERRED_SIZE,
37, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(oct2, javax.swing.GroupLayout.PREFERRED_SIZE,
37, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
              .addGroup(layout.createSequentialGroup()
                .addComponent(jLabel9)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
              .addGroup(layout.createSequentialGroup()
                .addComponent(oct3,
javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(oct4,
javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(48, 48, 48)
                .addComponent(jLabel2,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(cidrValue,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(161, 161, 161))))))
    );
    layout.setVerticalGroup(
      layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
      .addGroup(layout.createSequentialGroup()
        .addGap(16, 16, 16)
        .addComponent(jLabel9)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(oct1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(oct2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(oct3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(oct4, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE,
23, javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
20, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(cidrValue,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)

               .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
27, javax.swing.GroupLayout.PREFERRED_SIZE)

               .addComponent(jLabel8, javax.swing.GroupLayout.PREFERRED_SIZE,
27, javax.swing.GroupLayout.PREFERRED_SIZE)

               .addComponent(networkClass,
javax.swing.GroupLayout.PREFERRED_SIZE, 27,
javax.swing.GroupLayout.PREFERRED_SIZE)

               .addComponent(subnetMask,
javax.swing.GroupLayout.PREFERRED_SIZE, 27,
javax.swing.GroupLayout.PREFERRED_SIZE))

            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)

               .addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED_SIZE,
27, javax.swing.GroupLayout.PREFERRED_SIZE)

               .addComponent(networkAddress,
javax.swing.GroupLayout.PREFERRED_SIZE, 27,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)

            .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED_SIZE,
27, javax.swing.GroupLayout.PREFERRED_SIZE)

            .addComponent(broadcastAddress,
javax.swing.GroupLayout.PREFERRED_SIZE, 27,
javax.swing.GroupLayout.PREFERRED_SIZE))

          .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)

            .addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED_SIZE,
27, javax.swing.GroupLayout.PREFERRED_SIZE)

            .addComponent(noSubnets,
javax.swing.GroupLayout.PREFERRED_SIZE, 27,
javax.swing.GroupLayout.PREFERRED_SIZE))

          .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)

            .addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED_SIZE,
27, javax.swing.GroupLayout.PREFERRED_SIZE)

            .addComponent(hostsPSubnet,
javax.swing.GroupLayout.PREFERRED_SIZE, 27,
javax.swing.GroupLayout.PREFERRED_SIZE)

            .addComponent(calBtn, javax.swing.GroupLayout.PREFERRED_SIZE,
31, javax.swing.GroupLayout.PREFERRED_SIZE)

            .addComponent(resetBtn, javax.swing.GroupLayout.PREFERRED_SIZE,
31, javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
        .addGap(89, 89, 89))
    );

    setSize(new java.awt.Dimension(642, 345));
    setLocationRelativeTo(null);
  }// </editor-fold>//GEN-END:initComponents

  private void calBtnActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_calBtnActionPerformed

if(!oct1.getText().equals("")&&!oct2.getText().equals("")&&!oct3.getText().equals("")&&!oct4.getText().equals("")&&!cidrValue.getText().equals("")){

     int oct1v = Integer.parseInt(oct1.getText().toString());
    int oct2v = Integer.parseInt(oct2.getText().toString());
    int oct3v = Integer.parseInt(oct3.getText().toString());
    int oct4v = Integer.parseInt(oct4.getText().toString());

    int cidrv = Integer.parseInt(cidrValue.getText().toString());

    switch(cidrv){
      case 1: subnetMask.setText("128.0.0.0"); break;
      case 2: subnetMask.setText("192.0.0.0"); break;
      case 3: subnetMask.setText("224.0.0.0"); break;
      case 4: subnetMask.setText("240.0.0.0"); break;
case 5: subnetMask.setText("248.0.0.0"); break;

case 6: subnetMask.setText("252.0.0.0"); break;

case 7: subnetMask.setText("254.0.0.0"); break;

case 8: subnetMask.setText("255.0.0.0"); break;

case 9: subnetMask.setText("255.128.0.0"); break;

case 10: subnetMask.setText("255.192.0.0"); break;

case 11: subnetMask.setText("255.224.0.0"); break;

case 12: subnetMask.setText("255.240.0.0"); break;

case 13: subnetMask.setText("255.248.0.0"); break;

case 14: subnetMask.setText("255.252.0.0"); break;

case 15: subnetMask.setText("255.254.0.0"); break;

case 16: subnetMask.setText("255.255.0.0"); break;

case 17: subnetMask.setText("255.255.128.0"); break;

case 18: subnetMask.setText("255.255.192.0"); break;

case 19: subnetMask.setText("255.255.224.0"); break;

case 20: subnetMask.setText("255.255.240.0"); break;

case 21: subnetMask.setText("255.255.248.0"); break;

case 22: subnetMask.setText("255.255.252.0"); break;

case 23: subnetMask.setText("255.255.254.0"); break;

case 24: subnetMask.setText("255.255.255.0"); break;

case 25: subnetMask.setText("255.255.255.128"); break;

case 26: subnetMask.setText("255.255.255.192"); break;

case 27: subnetMask.setText("255.255.255.224"); break;

case 28: subnetMask.setText("255.255.255.240"); break;
```

```
          case 29: subnetMask.setText("255.255.255.248"); break;
          case 30: subnetMask.setText("255.255.255.252"); break;
          case 31: subnetMask.setText("255.255.255.254"); break;
          case 32: subnetMask.setText("255.255.255.255"); break;

          default: cidrValue.setText("Invalid");
      }
      if(oct1v>=0&&oct1v<=127){networkClass.setText("A");
          networkAddress.setText(oct1.getText()+".0.0.0");
          int m;

       }
      if(oct1v>=128&&oct1v<=191){networkClass.setText("B");
          networkAddress.setText(oct1.getText()+"."+oct2.getText()+".0.0");
      }
      if(oct1v>=192&&oct1v<=223){networkClass.setText("c");

networkAddress.setText(oct1.getText()+"."+oct2.getText()+"."+oct3.getText()+".0");
      }
      if(oct1v>=224&&oct1v<=239)networkClass.setText("D");
      if(oct1v>=240&&oct1v<=255)networkClass.setText("E");



broadcastAddress.setText(oct1.getText()+"."+oct2.getText()+"."+oct3.getText()+".255");
```

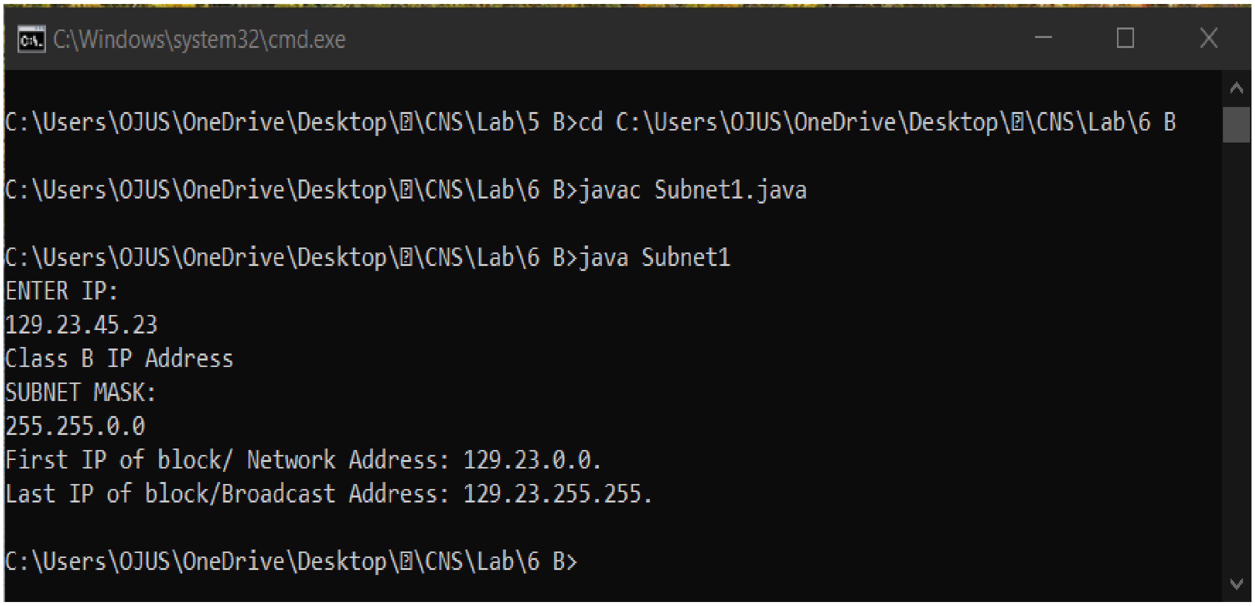
```
        int value=  32-Integer.parseInt(cidrValue.getText());
        int outPut=(int) Math.pow(2, value);
       hostsPSubnet.setText( ""+outPut ) ;

       //now set the no of subnets
       int subnets=Integer.parseInt(hostsPSubnet.getText())/Integer.parseInt(cidrValue.getText());

       noSubnets.setText(""+subnets);
       }else{
       JOptionPane.showMessageDialog(null, "please enter the missing value!!");
       }

    }//GEN-LAST:event_calBtnActionPerformed

    private void resetBtnActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_resetBtnActionPerformed
     oct1.setText("");
      oct2.setText("");
       oct3.setText("");
        oct4.setText("");
        cidrValue.setText("");
        subnetMask.setText("");
        networkAddress.setText("");
        broadcastAddress.setText("");
        noSubnets.setText("");
        hostsPSubnet.setText("");
        networkClass.setText("");

    }//GEN-LAST:event_resetBtnActionPerformed

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(MainCalculator.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(MainCalculator.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(MainCalculator.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(MainCalculator.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MainCalculator().setVisible(true);
            }
```

```
   });
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JTextField broadcastAddress;
private javax.swing.JButton calBtn;
private javax.swing.JTextField cidrValue;
private javax.swing.JTextField hostsPSubnet;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JLabel jLabel7;
private javax.swing.JLabel jLabel8;
private javax.swing.JLabel jLabel9;
private javax.swing.JTextField networkAddress;
private javax.swing.JTextField networkClass;
private javax.swing.JTextField noSubnets;
private javax.swing.JTextField oct1;
private javax.swing.JTextField oct2;
private javax.swing.JTextField oct3;
private javax.swing.JTextField oct4;
private javax.swing.JButton resetBtn;
```

```
    private javax.swing.JTextField subnetMask;
    // End of variables declaration//GEN-END:variables
}
```

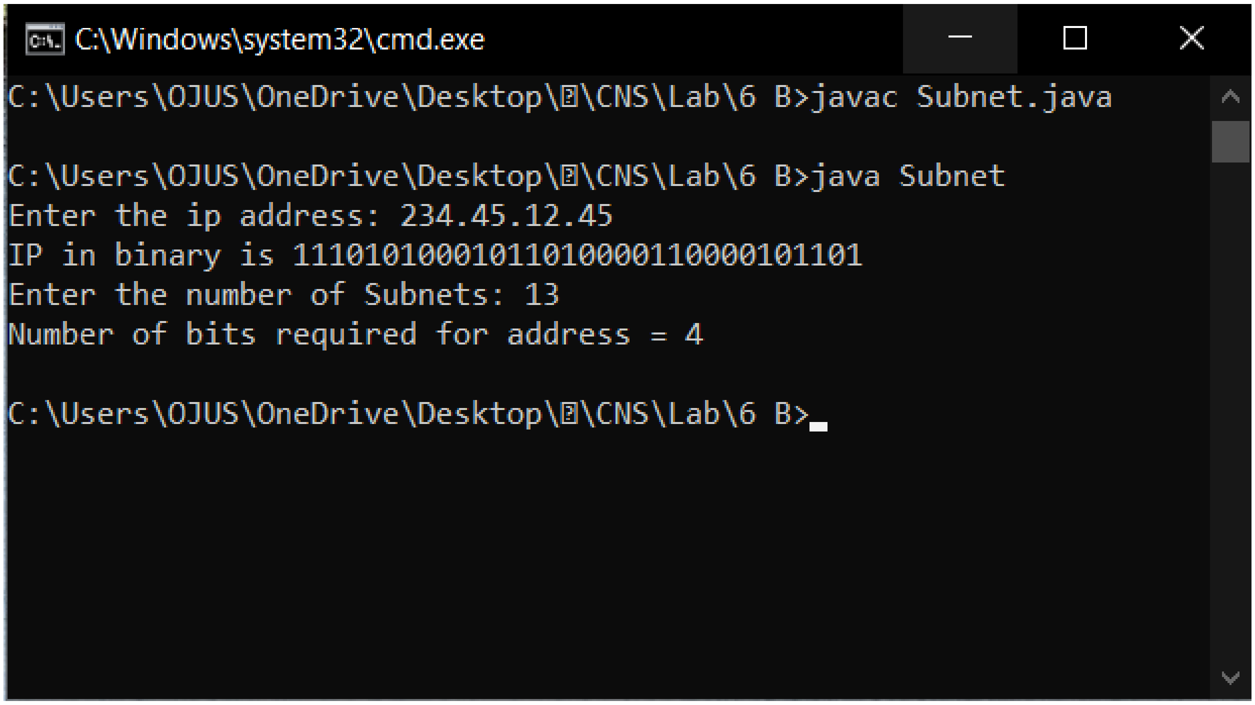
Output :

Subnet1



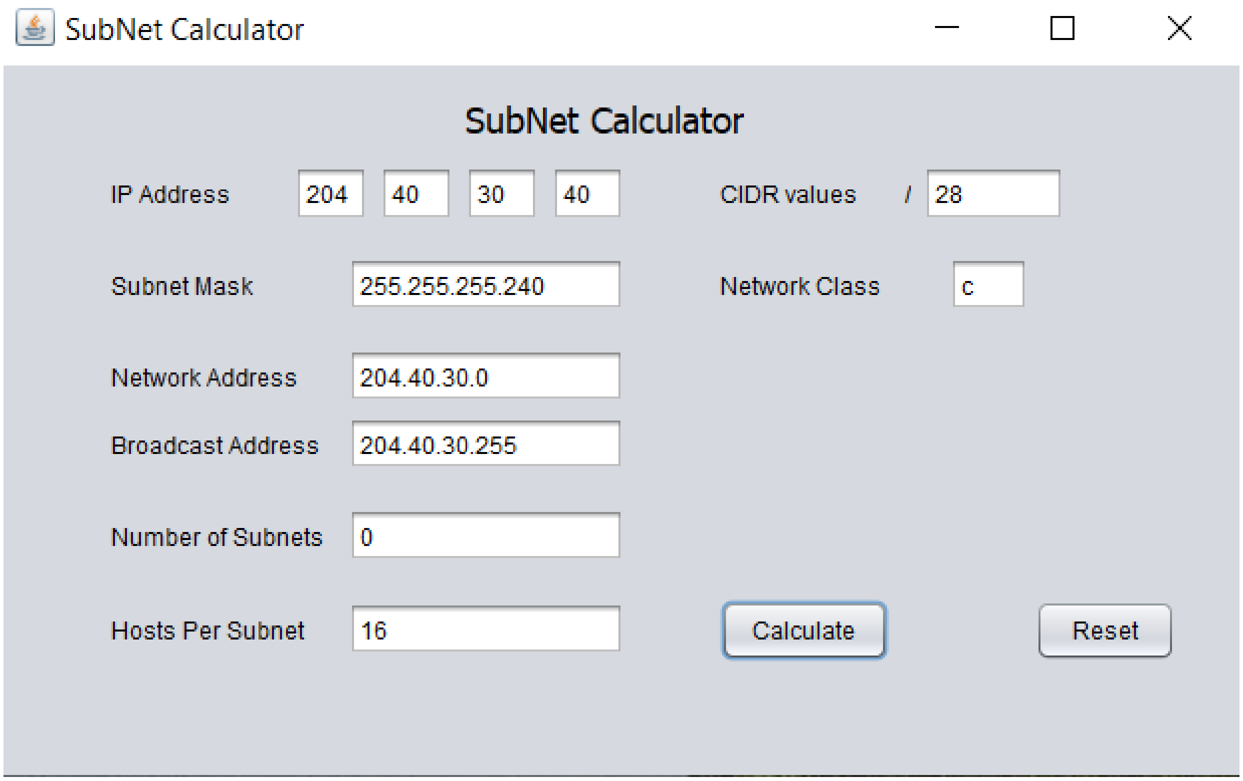
```
C:\Users\OJUS\OneDrive\Desktop\□\CNS\Lab\5 B>cd C:\Users\OJUS\OneDrive\Desktop\□\CNS\Lab\6 B

C:\Users\OJUS\OneDrive\Desktop\□\CNS\Lab\6 B>javac Subnet1.java

C:\Users\OJUS\OneDrive\Desktop\□\CNS\Lab\6 B>java Subnet1
ENTER IP:
129.23.45.23
Class B IP Address
SUBNET MASK:
255.255.0.0
First IP of block/ Network Address: 129.23.0.0.
Last IP of block/Broadcast Address: 129.23.255.255.

C:\Users\OJUS\OneDrive\Desktop\□\CNS\Lab\6 B>
```

Subnet



```
C:\Users\OJUS\OneDrive\Desktop\□\CNS\Lab\6 B>javac Subnet.java

C:\Users\OJUS\OneDrive\Desktop\□\CNS\Lab\6 B>java Subnet
Enter the ip address: 234.45.12.45
IP in binary is 11101010001011010000110000101101
Enter the number of Subnets: 13
Number of bits required for address = 4

C:\Users\OJUS\OneDrive\Desktop\□\CNS\Lab\6 B>
```

CIDR



SubNet Calculator

IP Address 204 40 30 40

CIDR values / 28

Subnet Mask 255.255.255.240

Network Class c

Network Address 204.40.30.0

Broadcast Address 204.40.30.255

Number of Subnets 0

Hosts Per Subnet 16