

```
public class IntegerNumber
{
    private int n;

    public IntegerNumber(int pN)
    {
        n = Math.abs(pN);
    }

    public int getNumber()
    {
        return n;
    }

    public boolean isEven()
    {
        // version 1
        if (n % 2 == 0)
            return true;
        else
            return false;

        // version 2 (mieux)
        // return n%2 == 0;
    }

    public boolean isPrime()
    {
        return isPrimeOptimized();
    }

    public boolean isPrimeClassic()
    {
        // Version classique, non-optimisée - plus facile à comprendre
        int count = 0;
        for (int i = 1; i <= n; i++)
        {
            if (n % i == 0)
                count++;
        }

        if (count == 2)
            return true;
        else
            return false;

        // ou encore mieux:
        // return (count == 2);
    }

    public boolean isPrimeOptimized()
    {
        // Version optimisée
        if (n == 1)
            return false;
        if (n == 2)
            return true;

        int i = 2;
        boolean found = false;
        int limit = ((int) Math.sqrt(n)) + 1;
        // Remarque:
        // le "(int)" et le "+1" permettent d'éviter les problèmes d'arrondis et de précision
        // en théorie ils sont de trop et avec précision parfaite (qui n'existe pas en JAVA) il faut écrire:
        // double limit = Math.sqrt(n);

        while (!found && (i <= limit))
        {
            if (n % i == 0)
                found = true; // un diviseur de n trouvé!
            i++;
        }

        return !found;
    }

    public int sumOfDividers(int pN)
    {
        // cette méthode est utilisée plusieurs fois...
        int sumDiv = 0;
        for (int i = 1; i <= pN; i++)
        {
            if (pN % i == 0)
                sumDiv = sumDiv + i;
        }
        return sumDiv;
    }

    public int sumOfDividers()
    {
        return sumOfDividers(n);
    }

    public boolean isPerfect()
    {
        return sumOfDividers() == 2 * n;
    }
}
```

```
public boolean isDeficient()
{
    return sumOfDividers() < 2 * n;
}

public boolean isAbundant()
{
    return sumOfDividers() > 2 * n;
}

public boolean isFriendlyTo(int pN)
{
    return sumOfDividers() == sumOfDividers(pN);
}

public int reverse(int pN)
{
    int res = 0;
    int val = pN;
    while (val != 0)
    {
        res = res * 10 + val % 10;
        val = val / 10;
    }
    return res;
}

public int reverse()
{
    return reverse(n);
}

public boolean isPalindrome()
{
    int rev = reverse();
    return n == rev;
}
}
```

```

public class MainFrame extends javax.swing.JFrame
{
    private IntegerNumber intNbr = null;

    public MainFrame()
    {
        initComponents();
        updateView();
    }

    public void updateView()
    {
        if (intNbr != null)
        {
            if (intNbr.isEven())
                evenLabel.setText("This number is even.");
            else
                evenLabel.setText("This number is not even.");

            if (intNbr.isPrime())
                primeLabel.setText("This number is prime.");
            else
                primeLabel.setText("This number is not prime.");

            if (intNbr.isPerfect())
                perfectLabel.setText("This number is perfect.");
            else
                perfectLabel.setText("This number is not perfect.");

            if (intNbr.isAbundant())
                abundantLabel.setText("This number is abundant.");
            else
                abundantLabel.setText("This number is not abundant.");

            if (intNbr.isDeficient())
                deficientLabel.setText("This number is deficient.");
            else
                deficientLabel.setText("This number is not deficient.");

            if (intNbr.isPalindrome())
                palindromeLabel.setText("This number is a palindrome.");
            else
                palindromeLabel.setText("This number is not a palindrome.");
        }
    }
}
// Skipped: ... initComponents { ... }
private void inputTextFieldActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_inputTextFieldActionPerformed
    intNbr = new IntegerNumber(Integer.valueOf(inputTextField.getText()));

    updateView();
} //GEN-LAST:event_inputTextFieldActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[])
{
    java.awt.EventQueue.invokeLater(new Runnable()
    {
        public void run()
        {
            new MainFrame().setVisible(true);
        }
    });
}

// Variables declaration - do not modify //GEN-BEGIN:variables
private javax.swing.JLabel abundantLabel;
private javax.swing.JLabel deficientLabel;
private javax.swing.JLabel evenLabel;
private javax.swing.JTextField inputTextField;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel palindromeLabel;
private javax.swing.JLabel perfectLabel;
private javax.swing.JLabel primeLabel;
// End of variables declaration //GEN-END:variables
}

```