```
/* calculate the average of the integers in the input line. If a negative integer is encountered in the input line,
 then throw a user defined NegativeNumberException  */

import java.util.Scanner;
import java.util.StringTokenizer;

class NegativeNumberException extends Exception {
   NegativeNumberException() {
      super("Numbers should be positive ");
   }
}

public class labProgram1 {

   public static float averageOfLine(String line) throws NegativeNumberException {
      float average;
      int size = 0;
      int sum = 0;
      StringTokenizer token = new StringTokenizer(line, " ");
      while (token.hasMoreTokens()) {
         size++;
         int num = Integer.parseInt(token.nextToken());
         if (num < 0)
            throw new NegativeNumberException();
         else
            sum = sum + num;
      }
      average = (float) sum / size;
      return average;
   }

   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter a line of integers separated by space :");
      String line = sc.nextLine();
      try {
         System.out.println("average of integers in  line : " + averageOfLine(line));
      } catch (NegativeNumberException e) {
         System.err.println(e.getMessage());
         e.printStackTrace();
      }
   }
}
```

```
/*
 *
 *
 * ALGORITHM
 *
 * 1]Start the program.
 *
 * 2]Declare a new class called NegativeNumberException which extends the
 *  built-in Exception class. Inside the class, define a constructor that sets
 *  the error message to "Numbers should be positive".
 *
 * 3]Define a new class called labProgram1.
 *
 * 4]Inside the labProgram1 class, define a static method called averageOfLine
 * that takes a string argument line and returns a float. The method can throw a
 * NegativeNumberException if it encounters a negative number in the input
 * string.
 *
 * 5]Declare the local variables average, size, and sum as integers.
 *
 * 6]Create a new StringTokenizer object called token to tokenize the input
 * string with a space as the delimiter.
 *
 * 7]While the token object has more tokens, do the following:
 * a. Increment the size variable by 1.
 * b. Parse the next token to an integer using Integer.parseInt().
 * c. If the number is negative, throw a new NegativeNumberException.
 * d. Otherwise, add the number to the sum variable.
 *
 * 8]Calculate the average of the input integers by dividing sum by size and
 * cast the result to a float.
 *
 * 9]Return the average from the averageOfLine method.
 *
 * 10]In the main method, create a new Scanner object called sc to read input
 * from the console.
 *
 * 11]Print a message to the console asking the user to enter a line of integers
 * separated by space.
 *
 * 12]Read the user input from the console using the nextLine() method of the
 * Scanner object and store it in a string variable called line.
 *
 * 13]Call the averageOfLine method with the line variable as an argument.
 *
 * 14]Print the calculated average to the console.
```

```
 *
 * 15]If a NegativeNumberException is caught, print the error message and stack
 * trace to the console.
 *
 * 16]End the program.
 *
 */


/*
 Java program that reads from a file, writes to another file, and displays the number of
sentences, words, and characters:
*/

import java.io.*;
import java.util.StringTokenizer;

//File copy statistics
public class labProgram2 {
   public static void main(String[] args) {

      BufferedReader reader = null;
      BufferedWriter writer = null;
      FileReader fr = null;
      FileWriter fw = null;
      int sentenceCount = 0;
      int wordCount = 0;
      int charCount = 0;
      try {
         fr = new FileReader("input555.txt");
         fw = new FileWriter("output555.txt");
         reader = new BufferedReader(fr);
         writer = new BufferedWriter(fw);
         String line = reader.readLine();

         while (line != null) {
            sentenceCount++;
            StringTokenizer tokenizer = new StringTokenizer(line, " ");
            wordCount += tokenizer.countTokens();
            charCount += line.length();
            writer.write(line);
            writer.newLine();
            line = reader.readLine();
         }

      } catch (IOException e) {
```

```
            System.err.println("Error in reading/writing files: " + e.getMessage());
        } finally {
            try {
                reader.close();
                writer.close();
                fr.close();
                fw.close();
            } catch (IOException e) {
                System.err.println("Error closing files: " + e.getMessage());
            }
        }

        System.out.println("Number of sentences: " + sentenceCount);
        System.out.println("Number of words: " + wordCount);
        System.out.println("Number of characters: " + charCount);
    }
}
```

ALGORITHM

1]Start the program
2]Declare the BufferedReader, BufferedWriter, FileReader and FileWriter variables as null
3]Initialize the int variables sentenceCount, wordCount and charCount to 0
4]Use a try-catch block
4.1] create object of FileReader fr with argument input.txt ,
and create object of FileWriter fw with argument output.txt
4.2] create a "reader" object of BufferedReader by passing fr as argument
create a "writer" object of BufferedWriter by passing fw as argument
4.3] crate string variable "line"
line = reader.readLine()
4.4] WHILE(line != null) do
1]sentenceCount = sentanceCount + 1
2]create an object "tokenizer" of StringTokenizer ,taking two arguments string "line" and delimiter " "(whitespace)
3]wordCount = wordCount + tokenizer.countTokens();
4]charCount = line.length()
5]writer.write();
6]writer.newLine();
7]line = reader.readLine()
5]if an exception occur in step 4 which throw an exception object and catch block catch it
5.1] display the error message using e.getMessage()
6] inside finally block
6.1] close the BufferedReader, BufferedWriter, FileReader, and FileWriter variables

7] Print the number of sentences, words, and characters to the console

8] End the program.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class PrimeChecker extends JFrame implements ActionListener {

   JTextField outputField, inputField;
   JButton submitButton;

   PrimeChecker() {

      setTitle("PRIME CHECKER");
      setBounds(600, 150, 420, 200);
      setResizable(false);

      outputField = new JTextField(10);
      outputField.setEditable(false);
      inputField = new JTextField(10);
      JLabel readLabel = new JLabel("Enter a number :");
      JLabel outputLabel = new JLabel("Output : ");

      submitButton = new JButton("CHECK");
      submitButton.addActionListener(this);

      JPanel panel = new JPanel();
      panel.setLayout(new GridLayout(3, 2));
      panel.add(readLabel);
      panel.add(inputField);
      panel.add(outputLabel);
      panel.add(outputField);
      panel.add(new JLabel());
      panel.add(submitButton);
      add(panel);

      setVisible(true);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   }

   public void actionPerformed(ActionEvent e) {
      if (e.getSource() == submitButton) {
         try {
            int number = Integer.parseInt(inputField.getText());// may throws
NumberFormatException
```

```
            boolean isPrime = true;
            if (number < 2)
               isPrime = false;
            else {
               for (int i = 2; i <= number / 2; i++)
                  if (number % i == 0) {
                     isPrime = false;
                     break;
                  }
            }
            if (isPrime)
               outputField.setText(number + " is  a  prime ");
            else
               outputField.setText(number + " is  not a  prime ");

         } catch (NumberFormatException ex) {
            outputField.setText("Invalid input ! ");
         }
      }
   }

   public static void main(String[] args) {
      new PrimeChecker();
   }

}

/*
   ALGORITHM

   [1] START
   [2] Import the required packages:javax.swing,java.awt,java.awt.event
   [3] Define a class "PrimeChecker" extends JFrame and implements ActionListener.
   [4] Declare instance variables of the class JTextField " outputField "and  " inputField "
,JButton " submitButton "
   [5] Define a constructor for the PrimeChecker class:
      (1) Set the title of the GUI window to "PRIME CHECKER".
      (2) Set the position and size of the GUI window using the setBounds method.
      (3) Instantiate the declared instances in step 3
         outputField: a text field with a width of 10 characters, which is initially not editable.
         inputField: a text field with a width of 10 characters.
         submitButton: a button with the text "CHECK"
      (4) create object of JLabel class " readLabel " with argument text "Enter a number :".
      (5) create object of JLabel class " outputLabel " with argument text "Output:"."
      (6) create a object of JPanel class  " panel " , uses a GridLayout with 3 rows and 2
columns to organize the GUI components.
```

(7) Add an ActionListener to the submitbutton using the addActionListener() method.
(8) Add the GUI components to the panel using add method of panel
(9) Add the panel to the frame.
(10) set the visibilty of frame true using setVisible method
(11)Set the default close operation of the frame to EXIT_ON_CLOSE
[6] Override the actionPerformed() method with argument e (object of ActionEvent)
(1)if e.getSource() is equal to submitButton then do the following steps from 2 to
(2)inside the try block
(1) read the input from the inputField using getText()
(2) convet the text number to integer number using parseInt() method of Integer class and store in variable number
(3) intialize a boolean variable isPrime as true
(4) if(number < 2) then set isPrime = false ;
(5) else then
(1)initialize i =2;
(2) while(i<= number/2) do
(1) if(number%2==0) then set isPrime = false and goto step
(2) i =i+1
(6) if(isPrime==true) then
(6.1) set the text of outputField as "number is prime " using setText() method
(7)else
(7.1) set the text of outputField as "number is not prime " using setText() method
(3) inside the catch block
(3.1) if an NumberFormatException is throws from parseInt() method
then set the text of outputField as "invalid input "
[7] define a main method
(1) create an object of PrimeChecker class
[8] STOP

```
// palindrome checker GUI program

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class PalindromeChecker1 extends JFrame implements ActionListener {

   JTextField inputfield, outputfield;
   JButton checkButton;

   PalindromeChecker1() {
      setTitle("PALINDROME CHECKER");
      setBounds(600, 200, 420, 300);
      inputfield = new JTextField();
```

```
        outputfield = new JTextField();
        outputfield.setEditable(false);
        checkButton = new JButton("CHECK");
        checkButton.addActionListener(this);
        JLabel inputlabel = new JLabel("Enter a string : ");
        JLabel outputlabel = new JLabel("result : ");
        JPanel panel = new JPanel(new GridLayout(3, 2));
        panel.add(inputlabel);
        panel.add(inputfield);
        panel.add(outputlabel);
        panel.add(outputfield);
        panel.add(new JLabel());
        panel.add(checkButton);
        add(panel);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public void actionPerformed(ActionEvent e) {

        if (e.getSource() == checkButton) {
            String str = inputfield.getText();
            int length = str.length();
            boolean isPalindrome = true;
            for (int i = 0; i < length / 2; i++) {
                if (str.charAt(i) != str.charAt(length - i - 1)) {
                    isPalindrome = false;
                    break;
                }
            }
            if (isPalindrome)
                outputfield.setText(str + " is palindrome ");
            else
                outputfield.setText(str + " is not palindrome ");
        }

    }

    public static void main(String args[]) {
        new PalindromeChecker1();
    }
}
```

```
import java.util.Scanner;

class EmployeeClass {

   private String name;
   private double salary;
   private String phoneNumber;

   public void setDetails(String name, double salary, String phoneNumber) {
      this.name = name;
      this.salary = salary;
      this.phoneNumber = phoneNumber;

   }

   public void salaryIncrement() {
      if (salary > 100000) {
         salary = salary + salary*.25;
      }
   }

   public void display() {
      System.out.println("\n DETAILS \n");
      System.out.println("Name : " + name);
      System.out.println("salary :" + salary);
      System.out.println("phoneNumber : " + phoneNumber);
   }

}

public class Main {
   public static void main(String[] args) {

      Scanner sc = new Scanner(System.in);

      System.out.print("Enter the name :");
      String name = sc.nextLine();
      System.out.print("Enter the salary :");
      double salary = sc.nextDouble();
      sc.nextLine();
      System.out.print("Enter the Phone Number :");
      String phoneNumber = sc.nextLine();

      EmployeeClass emp = new EmployeeClass();
      emp.setDetails(name, salary, phoneNumber);
      emp.salaryIncrement();
```

```
        emp.display();
    }
}



import java.util.Random;

class FactorialThread extends Thread {
    int number;

    FactorialThread(int number) {
        this.number = number;
    }

    public int factorial(int num) {
        if (num == 1)
            return 1;
        int fact = num * factorial(num - 1);
        return fact;
    }

    public void run() {
        System.out.println("factorial of " + number + " = " + this.factorial(number));
        this.factorial(number);
    }
}

class CubeThread extends Thread {
    int number;

    CubeThread(int number) {
        this.number = number;
    }

    public void run() {
        System.out.println("cube of " + number + " = " + number * number * number);
    }
}

class RandomThread extends Thread {
    Random r = new Random();
    int number;

    public void run() {
        for (int i = 0; i < 10; i++) {
```

```
            number = r.nextInt(60);
            if ((number > 1) && (number < 20))
               new FactorialThread(number).start();
            else
               new CubeThread(number).start();
         }
      }
   }

public class MultipleThread {
   public static void main(String[] args) {
      RandomThread thread = new RandomThread();
      thread.start();
   }
}
```

```
// read from user and write to a file
import java.io.*;
import java.util.Scanner;

public class FileReverse {
   public static void main(String[] args) {
      BufferedWriter bw1 = null;
      BufferedWriter bw2 = null;
      FileWriter fw1 = null;
      FileWriter fw2 = null;
      try {
         Scanner sc = new Scanner(System.in);
         fw1 = new FileWriter("writeNormal.txt");
         bw1 = new BufferedWriter(fw1);
         fw2 = new FileWriter("writeReverse.txt");
         bw2 = new BufferedWriter(fw2);
         int number = 0;
         int i = 0;
         System.out.println("Enter the number(enter -1 for stop ) : ");
         int numbers[] = new int[50];
         while (true) {
            number = sc.nextInt();
            if (number == -1)
               break;
            bw1.write(Integer.toString(number));
            bw1.newLine();
            numbers[i++] = number;
```

```
            }

            for (int j = i - 1; j >= 0; j--) {
               bw2.write(" " + numbers[j]);
               bw2.newLine();
            }

         } catch (IOException e) {
            System.out.println(e.getMessage());
            e.printStackTrace();
         } finally {
            try {
               bw1.close();
               bw2.close();
               fw1.close();
               fw2.close();
            } catch (IOException e) {
               System.out.println(e.getMessage());
            }
         }

      }

}
```

```
// read n numbers from the file and store in one file and store reverse in another file
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class FileReverse2 {
   public static void main(String[] args) {
      try {
         FileReader fr = new FileReader("numbers.txt");
         FileWriter fw1 = new FileWriter("newNormal.txt");
         FileWriter fw2 = new FileWriter("newReverse.txt");
         System.out.println("how many numbers should read from the file :");
         Scanner sc = new Scanner(System.in);
         int n = sc.nextInt();
         int i, num = 0, rev = 0;
         int j = 0;
         while ((i = fr.read()) != -1 && j < n) {
```

```
            num = (num * 10) + i - 48;
            j++;
         }
         int temp = num;
         while (temp != 0) {
            rev = rev * 10 + temp % 10;
            temp = temp / 10;
         }
         System.out.println(num + " " + rev);
         fw1.write(" " + num);
         fw2.write(" " + rev);
         fr.close();
         fw1.close();
         fw2.close();
      } catch (IOException e) {
         System.out.println(e.getMessage());
      } finally {

      }
   }
}



import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class CalculatorNew extends JFrame implements ActionListener {
   JButton numbers[] = new JButton[10];
   JButton clear;
   JButton add, sub, mul, div, dec, equ;
   char op;
   double result = 0, num1 = 0, num2 = 0;
   JTextField jf;

   CalculatorNew() {
      setTitle("CALCULATOR");
      setBounds(500, 100, 420, 500);
      setResizable(false);
      for (int i = 0; i < 10; i++) {
         numbers[i] = new JButton(Integer.toString(i));
         numbers[i].addActionListener(this);
      }
      add = new JButton("+");
      add.addActionListener(this);
```

```
    sub = new JButton("-");
    sub.addActionListener(this);
    mul = new JButton("*");
    mul.addActionListener(this);
    div = new JButton("/");
    div.addActionListener(this);
    dec = new JButton(".");
    dec.addActionListener(this);
    clear = new JButton("CLEAR");
    clear.addActionListener(this);
    equ = new JButton("=");
    equ.addActionListener(this);

    jf = new JTextField();
    jf.setBounds(0, 0, 410, 80);
    jf.setEditable(false);

    JPanel p = new JPanel();
    p.setLayout(new GridLayout(4, 4, 15, 15));
    p.add(numbers[7]);
    p.add(numbers[8]);
    p.add(numbers[9]);
    p.add(add);

    p.add(numbers[4]);
    p.add(numbers[5]);
    p.add(numbers[6]);
    p.add(sub);

    p.add(numbers[1]);
    p.add(numbers[2]);
    p.add(numbers[3]);
    p.add(mul);

    p.add(numbers[0]);
    p.add(dec);
    p.add(div);
    p.add(equ);
    add(jf, BorderLayout.NORTH);
    add(p, BorderLayout.CENTER);
    add(clear, BorderLayout.SOUTH);

    setVisible(true);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
```

```
public void actionPerformed(ActionEvent e) {
   for (int i = 0; i < 10; i++)
      if (numbers[i] == e.getSource())
         jf.setText(jf.getText() + i);
   if (clear == e.getSource())
      jf.setText("");
   else if (dec == e.getSource())
      jf.setText(jf.getText() + ".");
   else if (add == e.getSource()) {
      num1 = Double.parseDouble(jf.getText());
      jf.setText("");
      op = '+';
   } else if (sub == e.getSource()) {
      num1 = Double.parseDouble(jf.getText());
      jf.setText("");
      op = '-';
   } else if (div == e.getSource()) {
      num1 = Double.parseDouble(jf.getText());
      jf.setText("");
      op = '/';
   } else if (mul == e.getSource()) {
      num1 = Double.parseDouble(jf.getText());
      jf.setText("");
      op = '*';

   } else if (equ == e.getSource()) {
      try {
         num2 = Double.parseDouble(jf.getText());
         switch (op) {
            case '+':
               result = num1 + num2;
               break;
            case '-':
               result = num1 - num2;
               break;
            case '*':
               result = num1 * num2;
               break;
            case '/':
               if (num2 == 0)
                  throw new ArithmeticException("Division by zero not possible");
               result = num1 / num2;
               break;
         }
         jf.setText("" + result);
         num1 = result;
```

```
            } catch (ArithmeticException ex) {
                jf.setText(ex.getMessage());
            }
        }
    }

    public static void main(String[] args) {
        new CalculatorNew();
    }
}



/*
    Write a Java program that simulates a traffic light. The program lets the user select one of
    three lights: red, yellow, or green. When a radio button is selected, the light is turned on,
    and only one light can be on at a time. No light is on when the program starts

*/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class TrafficLight11 extends JPanel implements ActionListener {

    JRadioButton r1;
    JRadioButton r2;
    JRadioButton r3;
    Color redC;
    Color yellowC;
    Color greenC;

    TrafficLight11() {
        r1 = new JRadioButton("Red");
        r2 = new JRadioButton("yellow");
        r3 = new JRadioButton("green");
        redC = getBackground();
        yellowC = getBackground();
        greenC = getBackground();
        ButtonGroup gp = new ButtonGroup();
        gp.add(r1);
        gp.add(r2);
        gp.add(r3);
        r1.addActionListener(this);
        r2.addActionListener(this);
```

```
        r3.addActionListener(this);
        add(r1);
        add(r2);
        add(r3);
    }

    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.drawOval(50, 50, 50, 50);
        g.drawOval(50, 110, 50, 50);
        g.drawOval(50, 170, 50, 50);
        g.setColor(redC);
        g.fillOval(50, 50, 50, 50);
        g.setColor(yellowC);
        g.fillOval(50, 110, 50, 50);
        g.setColor(greenC);
        g.fillOval(50, 170, 50, 50);

    }

    public void actionPerformed(ActionEvent e) {
        if (r1.isSelected()) {
            redC = Color.red;
            yellowC = getBackground();
            greenC = getBackground();
        }

        else if (r2.isSelected()) {
            redC = getBackground();
            yellowC = Color.yellow;
            greenC = getBackground();
        }

        else if (r3.isSelected()) {
            redC = getBackground();
            yellowC = getBackground();
            greenC = Color.green;
        }

        repaint();
    }
}

public class Traffic11 {
    public static void main(String[] args) {
        JFrame frame = new JFrame("TRAFFIC LIIGHT");
```

```
        frame.setBounds(500, 100, 420, 420);
        TrafficLight11 tf = new TrafficLight11();
        frame.add(tf);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    }
}


/*
 write a program to read the first n characters in a file where n is given by the
user. The characters read from file has to be reversed and displayed on
screen and also store in another file. Built in methods can be used in the program
 */

import java.io.*;
import java.util.Scanner;

public class ReadCharacters {

    public static void main(String[] args) {
        FileReader fr = null;
        FileWriter fw = null;
        try {
            fr = new FileReader("readfile.txt");
            fw = new FileWriter("writefile.txt");
            Scanner sc = new Scanner(System.in);
            System.out.print("how many characters should read ?");
            int n = sc.nextInt();
            int i;
            int j = 0;
            String str = "";
            while (((i = fr.read()) != -1) && j < n) {
                str = str + Character.toString((char) i);
                j++;
            }
            System.out.println("n characters :" + str);
            str = new StringBuilder(str).reverse().toString();
            System.out.println("reversed charcters : " + str);
            j = 0;
            fw.write(str);
        } catch (IOException e) {
            System.out.println(e.getMessage());
        } finally {
            try {
```

```
            fw.close();
            fr.close();
         } catch (IOException e) {
            System.out.println("Error in closing files");
         }
      }
   }

}

/*
 Create a user defined exception "InvalidAgeException'. Write a Java program  that takes age as a Command Line Argument. Raise the Exception InvalidAgeException' if age is less than 18

*/

class InvalidAgeException extends Exception {
   public InvalidAgeException(String message) {
      super(message);
   }
}

public class Agevalidator {
   public static void main(String args[]) {
      try {
         int age = Integer.parseInt(args[0]);
         if (age < 18) {
            throw new InvalidAgeException("Age must be at least 18");
         }
         System.out.println("Age is valid");
      } catch (NumberFormatException e) {
         System.err.println("Invalid age: " + args[0]);
      } catch (InvalidAgeException e) {
         System.err.println("Invalid age: " + e.getMessage());
      }
   }
}
```