A blue logo with a black background

Description automatically generated

BÁO CÁO THỰC HÀNH

**Bài thực hành số 02**

**Môn học:** Lập trình ứng dụng mạng

**Lớp:** NT109.O21.MMCL

**SINH VIÊN THỰC HIỆN:**

|  |  |
| --- | --- |
| **Họ và tên** | **MSSV** |
| Nguyễn Văn Huấn | 21522108 |

# BÁO CÁO CHI TIẾT

## Nội dung 1

### 1.1: Calculator bằng thư viện AWT của Java

A screenshot of a calculator

Description automatically generated

(Giao diện của Calculator bằng thư viện AWT)

A screenshot of a computer

Description automatically generated

'**logFile logger = new logFile("21522108\_NguyenVanHuan\_CalculatorAWT.txt");**: Tạo một đối tượng của lớp **logFile** với tên là **logger**, và truyền vào đường dẫn của tệp tin "21522108\_NguyenVanHuan\_CalculatorAWT.txt". Đoạn này có thể liên quan đến việc ghi log hoặc ghi ra tệp tin.

**Frame f;**: Khai báo một biến **f** kiểu **Frame**, đây có thể là cửa sổ chính của ứng dụng.

**Label l1;**: Khai báo một biến **l1** kiểu **Label**, có thể được sử dụng để hiển thị các dòng văn bản trên giao diện người dùng.

**Button b1, b2, ..., bNegative;**: Khai báo một loạt các biến kiểu **Button** để tạo các nút trên giao diện người dùng. Các biến này sẽ được sử dụng để tạo các nút số từ 0 đến 9, các nút phép toán (+, -, \*, /, %), nút dấu chấm, nút xóa (DEL), nút xóa toàn bộ (AC), và nút dấu âm.

**double xd;**: Khai báo một biến **xd** kiểu số thực.

**double num1, num2, check;**: Khai báo các biến **num1**, **num2**, và **check** kiểu số thực. Các biến này có thể được sử dụng để lưu trữ các số và kết quả của các phép tính trong máy tính.

Top of Form

A screen shot of a computer code

Description automatically generated

**f= new Frame("MY CALCULATOR");**: Tạo một đối tượng của lớp **Frame** và gán cho biến **f**. Cửa sổ **Frame** được tạo ra sẽ có tiêu đề là "MY CALCULATOR".

**f.setBackground(Color.decode("#ADD8E6"));**: Thiết lập màu nền cho cửa sổ **f** bằng cách giải mã một chuỗi hex màu **"#ADD8E6"**, màu này tương ứng với màu xanh da trời nhạt.

**l1=new Label();**: Tạo một đối tượng của lớp **Label** và gán cho biến **l1**. Đây là một vùng hiển thị văn bản trên giao diện người dùng.

**l1.setBackground(Color.WHITE);**: Thiết lập màu nền cho **Label l1** là màu trắng.

**l1.setBounds(50,50,260,60);**: Đặt vị trí và kích thước của **Label l1** trên cửa sổ. Cụ thể, **Label l1** được đặt ở tọa độ (50, 50) và có kích thước 260x60 pixel.

**l1.setAlignment(Label.RIGHT);**: Thiết lập căn chỉnh cho văn bản trong **Label l1** sang phải (RIGHT).

**b1=new Button("1");**: Tạo một nút (button) mới với chữ "1" được hiển thị trên nút, và gán cho biến **b1**.

**b1.setBounds(50,340,50,50);**: Đặt vị trí và kích thước của **Button b1** trên cửa sổ. Cụ thể, **Button b1** được đặt ở tọa độ (50, 340) và có kích thước 50x50 pixel.

Tương tự cho các button khác, cách thiết lập giống Button 1

A screen shot of a computer

Description automatically generated

**b1.addActionListener(this);**, **b2.addActionListener(this);**, ..., **bNegative.addActionListener(this);**: Gắn một trình lắng nghe sự kiện cho mỗi nút. Trình lắng nghe sự kiện này được chỉ định bằng cách truyền **this**, tức là thể hiện hiện tại của lớp chứa đoạn mã này. Điều này ngụ ý rằng lớp hiện tại phải triển khai giao diện **ActionListener**.

**f.addWindowListener(this);**: Gắn một trình lắng nghe sự kiện cho cửa sổ. Trình lắng nghe sự kiện được chỉ định là **this**, nghĩa là thể hiện hiện tại của lớp chứa đoạn mã này phải triển khai giao diện **WindowListener**.

A computer code on a black background

Description automatically generated

**f.add(l1);**: Thêm **Label l1** vào cửa sổ **f**, để hiển thị vùng hiển thị văn bản.

**f.add(b1); f.add(b2); ... f.add(b0);**: Thêm các nút số từ 0 đến 9 vào cửa sổ **f**.

**f.add(bPlus); f.add(bMinus); ... f.add(bCalc);**: Thêm các nút phép toán (+, -, \*, /, %) và nút tính toán vào cửa sổ **f**.

**f.add(bDot); f.add(bDEL); f.add(bAC); f.add(bNegative);**: Thêm các nút dấu chấm, xóa (DEL), xóa toàn bộ (AC) và nút dấu âm vào cửa sổ **f**.

**f.setSize(360,500);**: Đặt kích thước của cửa sổ **f** là 360x500 pixel.

**f.setLayout(null);**: Đặt kiểu layout của cửa sổ **f** là null, điều này ngụ ý rằng các thành phần giao diện sẽ được đặt bằng tọa độ tuyệt đối.

**f.setVisible(true);**: Hiển thị cửa sổ **f**, làm cho cửa sổ trở nên có thể nhìn thấy trên màn hình.

A screen shot of a computer program

Description automatically generated

**String z, zt;**: Khai báo hai biến chuỗi **z** và **zt** để lưu trữ văn bản hiện tại trong **Label l1**.

**if(e.getSource() == b1) { ... }**: Kiểm tra xem sự kiện được kích hoạt có phải là từ nút **b1** không. Nếu điều kiện này đúng, có nghĩa là người dùng đã nhấn vào nút số 1.

**zt = l1.getText();**: Lấy văn bản hiện tại đã hiển thị trên **Label l1** và lưu vào biến **zt**.

**z = zt + "1";**: Tạo chuỗi mới **z** bằng cách nối chuỗi hiện tại (**zt**) với chuỗi "1", để thêm số 1 vào cuối chuỗi hiện tại.

**l1.setText(z);**: Thiết lập văn bản mới **z** cho **Label l1**, để hiển thị văn bản mới sau khi thêm số 1.

**logger.log("1");**: Ghi log về hành động này. Có thể là để ghi lại lịch sử các thao tác người dùng thực hiện trên ứng dụng.

Tương tự cho các nút khác trong Calculator

A computer screen shot of a code

Description automatically generated

(Xử lý sự kiện nút DEL)

**if(e.getSource()==bDEL){ ... }**: Kiểm tra xem sự kiện được kích hoạt có phải là từ nút **bDEL** không.

**zt=l1.getText();**: Lấy văn bản hiện tại đã hiển thị trên **Label l1** và lưu vào biến **zt**.

A computer screen shot of a program code

Description automatically generated

(Xử lý sự kiện nút "+")

**if(e.getSource()==bPlus) { ... }**: Kiểm tra xem sự kiện được kích hoạt có phải là từ nút **bPlus** không.

**try { ... } catch(NumberFormatException f) { ... }**: Thử thực hiện một khối lệnh có thể ném ra ngoại lệ **NumberFormatException**, đây là ngoại lệ xảy ra khi không thể chuyển đổi chuỗi sang số. Trong trường hợp này, nếu chuỗi không phải là một số hợp lệ, ngoại lệ sẽ xảy ra.

**num1=Double.parseDouble(l1.getText());**: Chuyển đổi chuỗi số trong **Label l1** thành một số thực (**double**) và lưu vào biến **num1**.

**l1.setText("Invalid Format");**: Thiết lập văn bản trong **Label l1** thành "Invalid Format" nếu không thể chuyển đổi chuỗi thành số, để thông báo cho người dùng về lỗi này.

**return;**: Kết thúc phương thức và không làm gì nữa nếu xảy ra ngoại lệ **NumberFormatException**.

**z="";**: Gán chuỗi rỗng cho biến **z**.

**l1.setText(z);**: Thiết lập văn bản trong **Label l1** thành chuỗi rỗng, để xóa bất kỳ nội dung hiện tại nào trong **Label l1**.

**check=1;**: Gán giá trị 1 cho biến **check**. Trong ngữ cảnh này, biến **check** có thể được sử dụng để đánh dấu phép tính cần thực hiện sau khi người dùng chọn một phép toán.

**logger.log("+");**: Ghi log về hành động này. Có thể là để ghi lại lịch sử các thao tác người dùng thực hiện trên ứng dụng. Trong trường hợp này, nó ghi lại phép toán cộng được thực hiện.

Tương tự cho các nút phép toán khác.

A screen shot of a computer code

Description automatically generated

A screen shot of a computer program

Description automatically generated

(Xử lý sự kiện nút "=")

Tương tự với cách xử lý nút "+", nếu chưa có số nào được nhập vào thì sẽ trả về câu báo ENTER NUMBER FIRST. Dựa vào giá trị check để biết thực hiện phép toán nào.

A computer screen shot of a code

Description automatically generated

(Xử lý sự kiện nút "AC")

Nếu nhấn nút "AC", mọi giá trị sẽ trả về 0 và xoá hết phép tính trên label l1.

A screenshot of a computer

Description automatically generated

(file log khi thực hiện phép tính 5\*5)

A screenshot of a computer

Description automatically generated

(file log sau khi tiếp tục thực hiện các phép tính +, /, -)

### 1.2: Calculator bằng Java Swing

A screenshot of a calculator with Gallery Arcturus in the background

Description automatically generated

(Giao diện của Calculator bằng Java Swing)

A computer screen shot of a program

Description automatically generated

**logFile logger = new logFile("21522108\_NguyenVanHuan\_SwingCalculator.txt");**: Khởi tạo một đối tượng **logger** từ lớp **logFile** để ghi log các hoạt động của ứng dụng vào tệp tin "21522108\_NguyenVanHuan\_SwingCalculator.txt".

**private DecimalFormat df = new DecimalFormat(("#, ###.0"));**: Khởi tạo một đối tượng **DecimalFormat** để định dạng số trong phạm vi hàng ngàn, với một số thập phân.

**private String[] symbols = { ... };**: Khai báo một mảng các chuỗi **symbols**, chứa các ký tự và toán tử cho các nút trên máy tính.

**private int operator = 0;**: Biến **operator** để lưu trữ loại toán tử đang được sử dụng.

**private JPanel panel = new JPanel(new BorderLayout(5,5));**: Khai báo một **JPanel** để chứa giao diện của máy tính, sử dụng bố cục **BorderLayout**.

**private JPanel btnPanel = new JPanel(new GridLayout(5,3,2,2));**: Khai báo một **JPanel** để chứa các nút của máy tính, sử dụng bố cục **GridLayout**.

**private JButton[] btns = new JButton[20];**: Một mảng các nút **JButton** để chứa các nút của máy tính.

**private JTextArea screen = new JTextArea(5,40);**: Khai báo một **JTextArea** để hiển thị dữ liệu đầu vào và kết quả tính toán.

**private double firstNum = 0, secondNum = 0;**: Biến **firstNum** và **secondNum** để lưu trữ số hạng đầu tiên và thứ hai trong các phép tính.

**private JTextField calculatingTf = new JTextField(40);**: Một **JTextField** để hiển thị phép tính đang được thực hiện.

A computer screen shot of a program code

Description automatically generated

**screen.setFont(new Font("Times New Roman", Font.BOLD,18));**: Đặt font chữ cho **JTextArea** **screen** thành "Times New Roman" có độ in đậm, kích thước 18.

**setTitle("Calculator");**: Đặt tiêu đề của cửa sổ chính thành "Calculator".

**screen.setBackground(Color.BLACK);**: Đặt màu nền của **JTextArea** **screen** thành màu đen.

**screen.setForeground(Color.WHITE);**: Đặt màu chữ của **JTextArea** **screen** thành màu trắng.

**panel.setBackground(Color.BLACK);**: Đặt màu nền của **JPanel** **panel** thành màu đen.

**btnPanel.setBackground(Color.BLACK);**: Đặt màu nền của **JPanel** **btnPanel** thành màu đen.

**for (int i=0;i<btns.length;i++){ ... }**: Duyệt qua mảng **btns**, tạo và cấu hình các nút theo các ký tự trong mảng **symbols**, sau đó thêm chúng vào **btnPanel**.

**calculatingTf.setForeground(Color.WHITE);**: Đặt màu chữ của **JTextField** **calculatingTf** thành màu trắng.

**calculatingTf.setBackground(Color.BLACK);**: Đặt màu nền của **JTextField** **calculatingTf** thành màu đen.

**panel.add(calculatingTf, BorderLayout.SOUTH);**: Thêm **calculatingTf** vào **panel** ở phía dưới cùng (SOUTH) của cửa sổ.

**panel.add(btnPanel, BorderLayout.CENTER);**: Thêm **btnPanel** vào **panel** ở giữa (CENTER) của cửa sổ.

**panel.add(screen, BorderLayout.NORTH);**: Thêm **screen** vào **panel** ở phía trên cùng (NORTH) của cửa sổ.

**add(panel);**: Thêm **panel** (bao gồm tất cả các thành phần của giao diện) vào cửa sổ chính.

**setSize(340,400);**: Đặt kích thước của cửa sổ chính là 340x400 pixel.

**setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);**: Đặt phương thức đóng của cửa sổ là **EXIT\_ON\_CLOSE**, nghĩa là ứng dụng sẽ thoát khi cửa sổ chính được đóng.

**setVisible(true);**: Hiển thị cửa sổ chính của ứng dụng.

A screen shot of a computer program

Description automatically generated

Phương thức **actionPerformed(ActionEvent e)** được gọi khi một hành động xảy ra trên một thành phần giao diện, như việc nhấn một nút. Dưới đây là giải thích từng dòng trong phương thức này:

**String cmd = e.getActionCommand().toString();**: Lấy lệnh được thực thi từ sự kiện **ActionEvent** và chuyển đổi thành chuỗi, gán vào biến **cmd**. Lệnh này thường tương ứng với văn bản hiển thị trên nút mà người dùng đã nhấn.

**switch (cmd) { ... }**: Sử dụng cấu trúc **switch** để kiểm tra giá trị của **cmd**.

**case ".":**: Trường hợp nếu lệnh là dấu chấm (".").

**logger.log(".");**: Ghi log về việc nhấn nút dấu chấm.

**if (!screen.getText().contains(".")) { ... }**: Kiểm tra xem văn bản hiện tại trong **screen** đã chứa dấu chấm hay chưa. Nếu không chứa, tiếp tục thực hiện các hành động bên trong.

**screen.setText(screen.getText() + ".");**: Thêm dấu chấm vào cuối văn bản hiện tại trong **screen**. Điều này giúp tránh việc người dùng nhập nhiều dấu chấm liên tiếp.

Tương tự cho các button khác trong Calculator

A screen shot of a computer program

Description automatically generated

(Xử lý sự kiện cho các nút phép toán)

**case "+":**: Trường hợp nếu người dùng nhấn nút biểu thị phép cộng.

* + **logger.log("+");**: Ghi log về việc nhấn nút cộng.
  + **if (!screen.getText().isEmpty()) { ... }**: Kiểm tra xem văn bản hiện tại trong **screen** có trống không. Nếu không, tiếp tục thực hiện các hành động bên trong.
  + **firstNum = Double.parseDouble(screen.getText().toString());**: Chuyển đổi văn bản hiện tại trong **screen** thành số thực và gán vào biến **firstNum**.
  + **operator = 1;**: Gán giá trị 1 cho biến **operator** để đánh dấu rằng phép tính là phép cộng.
  + **screen.setText("");**: Xóa văn bản hiện tại trong **screen**.

**case "-":**: Tương tự như trường hợp của phép trừ.

**case "x":**: Tương tự như trường hợp của phép nhân.

**case "÷":**: Tương tự như trường hợp của phép chia.

**case "%":**: Trường hợp nếu người dùng nhấn nút biểu thị tính phần trăm.

* + **logger.log("%");**: Ghi log về việc nhấn nút tính phần trăm.
  + **double num = Double.parseDouble(screen.getText().toString());**: Chuyển đổi văn bản hiện tại trong **screen** thành số thực và gán vào biến **num**.
  + **screen.setText(String.valueOf(num / 100.0));**: Hiển thị kết quả phần trăm trong **screen**.

A screen shot of a computer code

Description automatically generated

**case "+/-":**: Trường hợp nếu người dùng nhấn nút biểu thị việc đổi dấu (+/-).

* + **logger.log("-");**: Ghi log về việc nhấn nút đổi dấu.
  + **double neg = Double.parseDouble(screen.getText().toString());**: Chuyển đổi văn bản hiện tại trong **screen** thành số thực và gán vào biến **neg**.
  + **neg \*= -1;**: Đổi dấu của biến **neg**.
  + **screen.setText(String.valueOf(neg));**: Hiển thị số đã đổi dấu trong **screen**.

**case "AC":**: Trường hợp nếu người dùng nhấn nút biểu thị việc xóa toàn bộ nội dung trên màn hình (All Clear).

* + **screen.setText("");**: Xóa toàn bộ văn bản hiện tại trong **screen**, đưa nó về trạng thái rỗng.

A screen shot of a computer program

Description automatically generated

(Xử lý sự kiện cho nút "=")

**if (cmd.equalsIgnoreCase("=")) { ... }**: Kiểm tra nếu lệnh (command) là "=" (bằng nhau), bằng cách sử dụng phương thức **equalsIgnoreCase()** để so sánh không phân biệt hoa thường.

**if (!screen.getText().isEmpty()) { ... }**: Kiểm tra xem văn bản hiện tại trong **screen** có trống không. Nếu không, tiếp tục thực hiện các hành động bên trong.

**secondNum = Double.parseDouble(screen.getText().toString());**: Chuyển đổi văn bản hiện tại trong **screen** thành số thực và gán vào biến **secondNum**, đại diện cho số thứ hai trong phép tính.

**switch (operator) { ... }**: Sử dụng cấu trúc **switch** để kiểm tra giá trị của biến **operator**, đại diện cho loại phép tính.

**case 1: // +**: Trường hợp nếu biến **operator** có giá trị là 1, tương ứng với phép cộng.

* + **screen.setText(String.valueOf(firstNum + secondNum));**: Hiển thị kết quả của phép tính cộng trong **screen**.
  + **calculatingTf.setText(String.valueOf(firstNum + "+" + secondNum + " = " + (df.format(firstNum + secondNum))));**: Hiển thị chuỗi biểu thức tính toán cùng với kết quả trong **calculatingTf**.
  + **logger.log(df.format(firstNum + secondNum));**: Ghi log kết quả của phép tính cộng với định dạng số được xác định bởi **DecimalFormat**.

Top of Form

Tương tự cho các phép toán -, \*, /A screenshot of a computer

Description automatically generated

(file log sau khi thực hiện các phép toán +,-,\*,/)

## Khó khăn

- Sử dụng Java Swing chưa thành thục nhưng sau bài Lab02 này, có thể đã cải thiện được đôi chút.

- Chưa thực sự có một giao diện đẹp, thân thiện với người dung.

- Còn nhiều test case vẫn chưa hoàn thiện, ứng dụng vẫn chưa tối ưu.

# TÀI LIỆU THAM KHẢO