

MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng:

- ✓ Load dữ liệu từ Database lên JTextField, JComboBox và Jtable
- ✓ Thực hiện các chức năng thêm, xóa, sửa, update, tìm kiếm:

NỘI DUNG:

BÀI 1: JDBC – Thiết kế ứng dụng như hình và xử lý chức năng tìm kiếm và xóa dữ liệu (5 ĐIỂM)

Chuẩn bị cơ sở dữ liệu (Cơ sở dữ liệu này được tạo với MS SQL Server)

```
CREATE DATABASE Library

USE
Library
GO

CREATE TABLE Books (
    ID INT IDENTITY PRIMARY KEY
    ,title NVARCHAR(50)
    ,price FLOAT
)

GO

INSERT INTO Books (title, price) VALUES (N'Lập trình C',100)
INSERT INTO Books (title, price) VALUES (N'Lập trình Java',200)
INSERT INTO Books (title, price) VALUES (N'Lập trình C#',150)
GO
```

Cho phép người dùng xem và xoá dữ liệu của bảng Books. Ngoài ra chương trình cũng cho phép người dùng tìm kiếm một sách theo tiêu đề (title)

Book information

Filter

Title:

Search Exit

Id	Title	Price

Delete

Mô tả yêu cầu

Đối với nút “**Search**” nếu vùng filter không nhập, chương trình sẽ hiển thị tất cả dữ liệu của bảng Books. Ngược lại chương trình chỉ hiển thị đúng với thông tin mà người dùng đã nhập hoặc ứng dụng sẽ thông báo “**The book is not available!**” trong trường hợp không tìm thấy

Book information

Filter

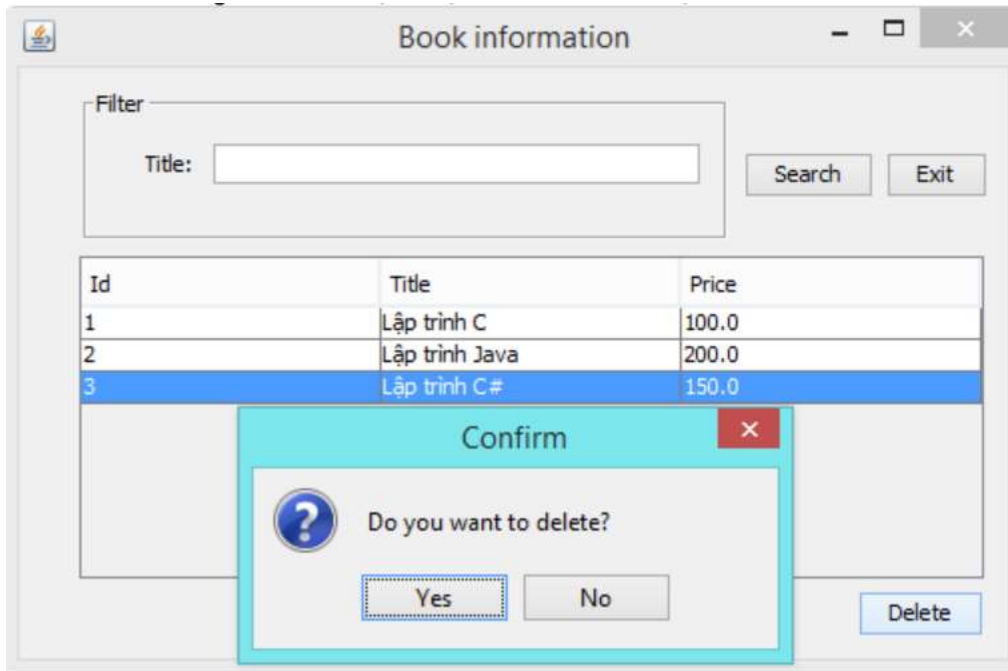
Title:

Search Exit

Id	Title	Price
1	Lập trình C	100.0
2	Lập trình Java	200.0
3	Lập trình C#	150.0

Delete

Khi người dùng chọn một dòng trên table, nút “Delete” sẽ ở trạng thái sử dụng và chương trình sẽ hiển thị thông báo xác nhận “Do you want to delete?” khi người dùng chọn nút “Delete”. Nếu chọn “Yes”, chương trình sẽ xoá thông tin sách được chọn khỏi cơ sở dữ liệu.



Nút “Exit” sẽ đóng chương trình

Code:

Khai báo các biến toàn cục:

```
public class Book extends javax.swing.JFrame {

    /**
     * Creates new form Book
     */
    String userName = "sa";
    String password = "123";
    String url = "jdbc:sqlserver://localhost:1433;database=Library;encrypt=false";

    public Book() {
        initComponents();
    }
}
```

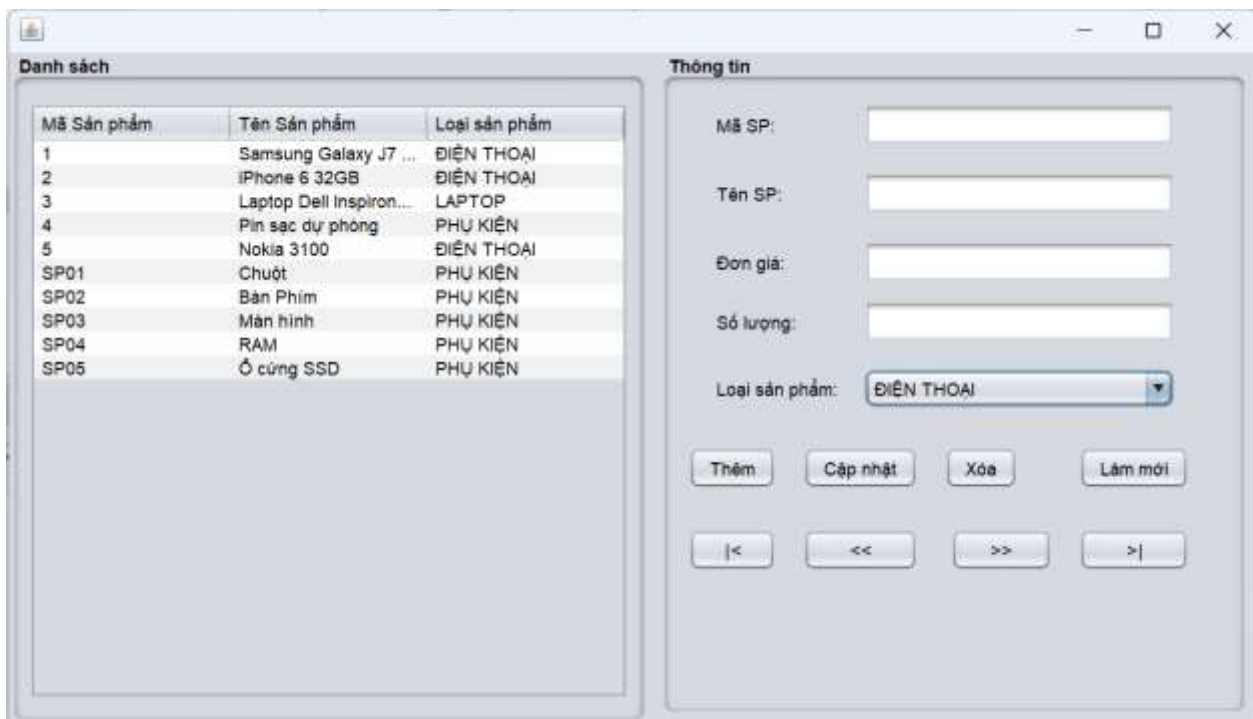
Viết xử lý cho nút "Search"

```
private void btnSearchActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String header[] = {"Id", "Title", "Price"};  
    DefaultTableModel tblModel = new DefaultTableModel(columnNames:header, rowCount:0);  
    Connection conn = null;  
    try {  
        Class.forName(className: "com.microsoft.sqlserver.jdbc.SQLServerDriver");  
        conn = DriverManager.getConnection(url, user: userName, password);  
        String sql = "select * from Books";  
        if (txtTitle.getText().length() > 0) {  
            sql = sql + " where title like '%" + txtTitle.getText() + "%'";  
        }  
        Statement stm = conn.createStatement();  
        ResultSet rs = stm.executeQuery(string: sql);  
        if (rs.isBeforeFirst() == false) {  
            JOptionPane.showMessageDialog(parentComponent:this, message:"The book is not available!");  
            return;  
        }  
        while (rs.next()) {  
            Vector data = new Vector();  
            data.add(rs.getInt(string: "id"));  
            data.add(rs.getString(string: "title"));  
            data.add(rs.getString(string: "price"));  
            tblModel.addRow(rowData:data);  
        }  
        tblBooks.setModel(dataModel: tblModel);  
    } catch (Exception e) {  
        System.out.println(v: e);  
    } finally {  
        try {  
            conn.close();  
        } catch (SQLException ex) {  
            ex.printStackTrace();  
        }  
    }  
}
```

Viết xử lý cho nút “Delete”

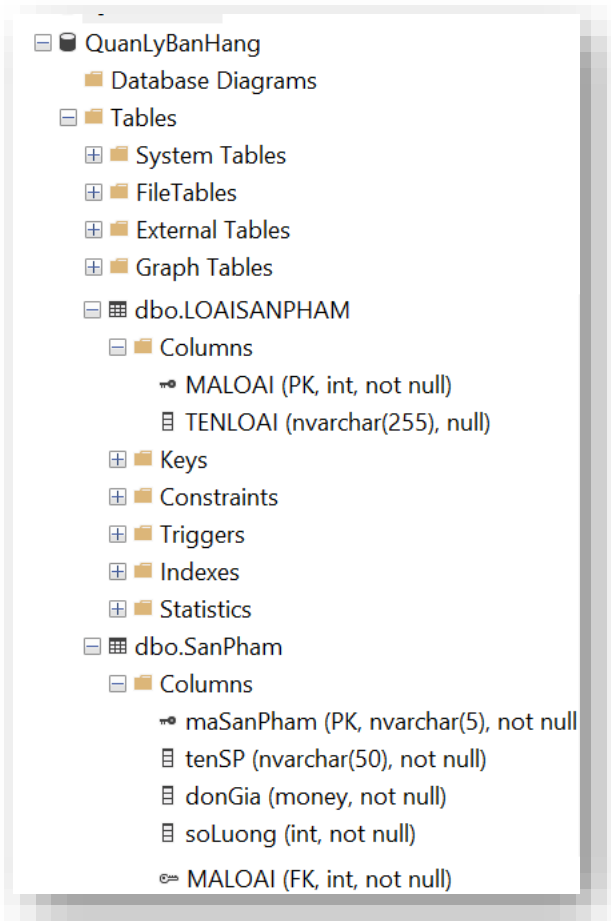
```
private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int ret = JOptionPane.showConfirmDialog(parentComponent, this, message: "Do you want to delete?", title: "Confirm",
        JOptionPane.YES_NO_OPTION);
    if (ret != JOptionPane.YES_OPTION) {
        return;
    }
    try {
        int id = (int) tblBooks.getValueAt(tblBooks.getSelectedRow(), 0); // giá trị của cột đầu tiên tại dòng được chọn trong table
        Connection conn = DriverManager.getConnection(url, user: username, password);
        String sql = "delete from Books where id = ?";
        PreparedStatement st = conn.prepareStatement(sql);
        st.setInt(1, id);
        st.executeUpdate();
        JOptionPane.showMessageDialog(parentComponent, this, message: "Delete thành công!");
        conn.close();
    } catch (Exception e) {
        System.out.println(e);
    }
}
```

BÀI 2: Thiết kế ứng dụng như hình và xử lý chức năng theo yêu cầu (5 ĐIỂM)



Mã Sản phẩm	Tên Sản phẩm	Loại sản phẩm
1	Samsung Galaxy J7 ...	ĐIỆN THOẠI
2	iPhone 6 32GB	ĐIỆN THOẠI
3	Laptop Dell Inspiron...	LAPTOP
4	Pin sạc dự phòng	PHỤ KIỆN
5	Nokia 3100	ĐIỆN THOẠI
SP01	Chuột	PHỤ KIỆN
SP02	Bàn phím	PHỤ KIỆN
SP03	Màn hình	PHỤ KIỆN
SP04	RAM	PHỤ KIỆN
SP05	Ổ cứng SSD	PHỤ KIỆN

Database:



Xử lý sự kiện:

- Khi người dùng click vào Jtable, hiển thị các thông tin tương ứng lên các controls bên phải.
- Viết xử lý cho nút “Thêm”: Lấy các giá trị người dùng đã nhập insert vào table Student.
- Viết xử lý cho nút “Cập nhật”: Cập nhật các giá trị đã thay đổi vào table.
- Viết xử lý cho nút “Xóa”: Xóa dòng hiện tại
- Viết xử lý cho nút “Làm mới”: Clear tất cả các giá trị controls bên phải, cho người dùng nhập mới

- Viết xử lý cho nút “First(<|)” : Đọc record đầu tiên trong table hiển thị lên các controls bên phải
- Viết xử lý cho nút “Previous(<<)” : Đọc record đứng trước dòng hiện tại trong table hiển thị lên các controls bên phải
- Viết xử lý cho nút “Next(>>)” : Đọc record tiếp theo trong table hiển thị lên các controls bên phải
- Viết xử lý cho nút “Last (>|)” : Đọc record đầu tiên trong table hiển thị lên các controls bên phải

Code tham khảo:

Tạo class SanPham, LoaiSanPham như sau:

```
/**
 *
 * @author nttai
 */
public class SanPham {
    String maSanPham;
    String tenSP;
    double donGia;
    int soLuong;
    LoaiSanPham loaiSanPham;
    public SanPham() {
    }
    public SanPham(String maSanPham, String tenSP, double donGia, int soLuong, LoaiSanPham loaiSanPham) {
        this.maSanPham = maSanPham;
        this.tenSP = tenSP;
        this.donGia = donGia;
        this.soLuong = soLuong;
        this.loaiSanPham = loaiSanPham;
    }
}
```

```

/**
 *
 * @author nttai
 */
public class LoaiSanPham {
    int maLoai;
    String tenLoai;

    public LoaiSanPham() {
    }

    public LoaiSanPham(int maLoai, String tenLoai) {
        this.maLoai = maLoai;
        this.tenLoai = tenLoai;
    }
}

```

Tạo class SanPhamDAO để xử lý thao tác với cơ sở dữ liệu

✓ Khai báo biến toàn cục

```

/**
 *
 * @author nttai
 */
public class SanPhamDAO {
    String INSERT_SQL = "INSERT INTO SanPham (MASANPHAM, TENSP, DONGIA, SOLUONG, MALOAI) VALUES (?, ?, ?, ?, ?)";
    String UPDATE_SQL = "UPDATE SanPham set TENSP = ?, DONGIA = ?, SOLUONG = ?, MALOAI = ? WHERE MASANPHAM = ?";
    String DELETE_SQL = "DELETE FROM SanPham WHERE MASANPHAM=?";
    String SELECT_ALL_SQL = "SELECT * FROM SanPham";
    String SELECT_BY_ID_SQL = "SELECT * FROM SanPham where MASANPHAM=?";
    String SELECT_LOAISANPHAM_BY_ID_SQL = "SELECT * FROM LoaiSanPham where MALOAI=?";
    String SELECT_ALL_LOAISANPHAM_SQL = "SELECT * FROM LOAISANPHAM";
    String userName = "sa";
    String password = "123";
    String url = "jdbc:sqlserver://localhost:1433;database=QuanLyBanHang;encrypt=false";
}

```


- ✓ **Hàm insert:** thêm đối tượng SanPham vào cơ sở dữ liệu

```
public int insert(SanPham sp) {  
    int rs = 0;  
    try {  
        Connection conn = DriverManager.getConnection(url, user: userName, password);  
        PreparedStatement st = conn.prepareStatement(string: INSERT_SQL);  
        st.setString(1, string: sp.getMaSanPham());  
        st.setString(2, string: sp.getTenSP());  
        st.setDouble(3, d: sp.getDonGia());  
        st.setInt(4, i: sp.getSoLuong());  
        st.setInt(5, i: sp.getLoaiSanPham().getMaLoai());  
        rs = st.executeUpdate();  
        conn.close();  
    } catch (Exception e) {  
        System.out.println(x: e);  
    }  
    return rs;  
}
```

- ✓ **Hàm update:** cập nhật đối tượng SanPham vào cơ sở dữ liệu

```
public int update(SanPham sp) {  
    int rs = 0;  
    try {  
        Connection conn = DriverManager.getConnection(url, user: userName, password);  
        PreparedStatement st = conn.prepareStatement(string: UPDATE_SQL);  
        st.setString(1, string: sp.getTenSP());  
        st.setDouble(2, d: sp.getDonGia());  
        st.setInt(3, i: sp.getSoLuong());  
        st.setInt(4, i: sp.getLoaiSanPham().getMaLoai());  
        st.setString(5, string: sp.getMaSanPham());  
        rs = st.executeUpdate();  
        conn.close();  
    } catch (Exception e) {  
        System.out.println(x: e);  
    }  
    return rs;  
}
```

- ✓ **Hàm delete:** xóa đối tượng SanPham trong cơ sở dữ liệu

```
public int delete(String maSP) {  
    int rs = 0;  
    try {  
        Connection conn = DriverManager.getConnection(url, user: userName, password);  
        PreparedStatement st = conn.prepareStatement(string: DELETE_SQL);  
        st.setString(1, string: maSP);  
        rs = st.executeUpdate();  
        conn.close();  
    } catch (Exception e) {  
        System.out.println(x: e);  
    }  
    return rs;  
}
```

- ✓ **getAllSanPham:** Lấy tất cả danh sách sản phẩm trong cơ sở dữ liệu

```
public List<SanPham> getAllSanPham() {  
    List<SanPham> list = new ArrayList<>();  
    try {  
        Connection conn = DriverManager.getConnection(url, user: userName, password);  
        String sql = SELECT_ALL_SQL;  
        Statement stm = conn.createStatement();  
        ResultSet rs = stm.executeQuery(string: sql);  
        while (rs.next()) {  
            SanPham sp = new SanPham();  
            sp.setMaSanPham(maSanPham: rs.getString(string: "maSanPham"));  
            sp.setTenSP(tenSP: rs.getString(string: "tenSP"));  
            sp.setSoLuong(soLuong: rs.getInt(string: "soLuong"));  
            sp.setDonGia(donGia: rs.getDouble(string: "donGia"));  
            int maLoai = rs.getInt(string: "maLoai");  
            sp.setLoaiSanPham(loaiSanPham: this.getLoaiSanPhamByMaLoai(maLoai));  
            list.add(e: sp);  
        }  
    } catch (Exception e) {  
        System.out.println(x: e);  
    }  
    return list;  
}
```

- ✓ **getSanPhamByMaSanPham:** Lấy sản phẩm trong cơ sở dữ liệu theo mã sản phẩm

```
public SanPham getSanPhamByMaSanPham(String maSP) {
    SanPham sp = new SanPham();
    try {
        Connection conn = DriverManager.getConnection(url, user: userName, password);
        String sql = SELECT_BY_ID_SQL;

        PreparedStatement stm = conn.prepareStatement(string: sql);
        stm.setString(1, string: maSP);
        ResultSet rs = stm.executeQuery();
        while (rs.next()) {
            sp.setMaSanPham(maSanPham: rs.getString(string: "maSanPham"));
            sp.setTenSP(tenSP: rs.getString(string: "tenSP"));
            sp.setSoLuong(soLuong: rs.getInt(string: "soLuong"));
            sp.setDonGia(donGia: rs.getDouble(string: "donGia"));
            int maLoai = rs.getInt(string: "maLoai");
            sp.setLoaiSanPham(loaiSanPham: this.getLoaiSanPhamByMaLoai(maLoai));
        }
    } catch (Exception e) {
        System.out.println(e);
    }
    return sp;
}
```

- ✓ **getAllLoaiSanPham:** Lấy tất cả danh sách loại sản phẩm trong cơ sở dữ liệu

```
public List<LoaiSanPham> getAllLoaiSanPham() {
    List<LoaiSanPham> list = new ArrayList<>();
    try {
        Connection conn = DriverManager.getConnection(url, user: userName, password);
        String sql = SELECT_ALL_LOAISANPHAM_SQL;
        Statement stm = conn.createStatement();
        ResultSet rs = stm.executeQuery(string: sql);
        while (rs.next()) {
            LoaiSanPham lsp = new LoaiSanPham();
            lsp.setMaLoai(maLoai: rs.getInt(string: "maLoai"));
            lsp.setTenLoai(tenLoai: rs.getString(string: "tenLoai"));
            list.add(lsp);
        }
    } catch (Exception e) {
        System.out.println(e);
    }
    return list;
}
```

✓ **getLoaiSanPhamByMaLoai:** Lấy loại sản phẩm trong cơ sở dữ liệu theo mã loại

```
public LoaiSanPham getLoaiSanPhamByMaLoai(int maLoai){
    LoaiSanPham loaiSanPham = new LoaiSanPham();
    try {
        Connection conn = DriverManager.getConnection(url, user, password);
        String sql = SELECT_LOAISANPHAM_BY_ID_SQL;
        PreparedStatement stm = conn.prepareStatement(sql);
        stm.setInt(1, maLoai);
        ResultSet rs = stm.executeQuery();
        while (rs.next()) {
            loaiSanPham.setMaLoai(rs.getInt("maLoai"));
            loaiSanPham.setTenLoai(rs.getString("tenLoai"));
        }
    } catch (Exception e) {
        System.out.println(e);
    }
    return loaiSanPham;
}
```

Viết code xử lý cho SanPhamFrame

Khai báo biến toàn cục

```
/**
 *
 * @author nttai
 */
public class SanPhamFrame extends javax.swing.JFrame {

    /**
     * Creates new form SanPhamFrame
     */
    SanPhamDAO spdao = new SanPhamDAO();
    List<SanPham> listSanPhams = new ArrayList<>();
    int current = 0;
    public SanPhamFrame() {
        initComponents();
        loadDataToTable();
        loadDataToCombobox();
    }
}
```

Hàm loadDataToTable(): đọc tất cả dữ liệu bảng “SanPham” trong cơ sở dữ liệu hiển thị lên table “tblSanPhams”.

```
public void loadDataToTable() {
    listSanPhams = spdao.getAllSanPham();
    String header[] = {"Mã sản phẩm", "Tên sản phẩm", "Loại sản phẩm"};
    DefaultTableModel tblModel = new DefaultTableModel(listSanPhams.header, listSanPhams.data);
    for (SanPham sp : listSanPhams) {
        tblModel.addRow(new Object[] {sp.getMaSanPham(), sp.getTenSP(), sp.getLoaiSanPham().getTenLoai()});
    }
    tblSanPhams.setModel(tblModel);
}
```

Hàm loadDataToCombobox()

```
public void loadDataToCombobox() {
    List<LoaiSanPham> loaiSanPhams = spdao.getAllLoaiSanPham();
    DefaultComboBoxModel model = (DefaultComboBoxModel) cboLoaiSP.getModel();
    model.removeAllElements();
    for (LoaiSanPham loaiSanPham : loaiSanPhams) {
        model.addElement(loaiSanPham);
    }
}
```

Hàm setForm(int i): dùng để hiển thị sản phẩm thứ i trong tblSanPhams lên các control bên phải

```
public void setForm(int i) {
    SanPham sp = listSanPhams.get(index: i);
    txtMaSP.setText(t: sp.getMaSanPham());
    txtTenSP.setText(t: sp.getTenSP());
    txtDonGia.setText(t: String.format(format: "%,.0f", args: sp.getDonGia()));
    txtSoLuong.setText(t: String.valueOf(i: sp.getSoLuong()));
    cboLoaiSP.getModel().setSelectedItem(anItem: sp.getLoaiSanPham());
    tblSanPhams.setRowSelectionInterval(index0: i, index1: i);
}
```

Hàm `getForm()`: lấy thông tin từ các control bên phải

```
public SanPham getForm() {
    SanPham sp = new SanPham();
    sp.setMaSanPham(maSanPham: txtMaSP.getText());
    sp.setTenSP(tenSP: txtTenSP.getText());
    sp.setDonGia(donGia: Double.valueOf(txtDonGia.getText().replaceAll(regex: ",", replacement: "")));
    sp.setSoLuong(soLuong: Integer.valueOf(txtSoLuong.getText()));
    sp.setLoaiSanPham((LoaiSanPham) cboLoaiSP.getSelectedItem());
    return sp;
}
```

Hàm `clearForm()`: reset thông tin các control bên phải

```
public void clearForm() {
    txtMaSP.setText(t: "");
    txtTenSP.setText(t: "");
    txtDonGia.setText(t: "");
    txtSoLuong.setText(t: "");
    cboLoaiSP.getModel().setSelectedItem(anItem: "");
}
```

Viết xử lý cho các nút: First, Next, Previous, Last

```
private void btnFirstActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    current = 0;
    setForm(i: current);
}
```

```
private void btnPreviousActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(current == 0){
        JOptionPane.showMessageDialog(parentComponent:this, message:"Đang ở đầu danh sách!");
        return;
    }
    current--;
    setForm(1, current);
}
```

```
private void btnNextActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(current == listSanPhams.size() - 1){
        JOptionPane.showMessageDialog(parentComponent:this, message:"Đang ở cuối danh sách!");
        return;
    }
    current++;
    setForm(1, current);
}
```

```
private void btnLastActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    current = listSanPhams.size() - 1;
    setForm(1, current);
}
```

Viết xử lý cho nút “Thêm”:

```
private void btnThemActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    SanPham sp = getForm();
    int rs = spdao.insert(sp);
    if(rs > 0){
        JOptionPane.showMessageDialog(parentComponent:this, message:"Thêm sản phẩm thành công!");
        loadDataToTable();
    }
}
```


Viết xử lý cho nút “Cập nhật”:

```
private void btnCapNhatActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    SanPham sp = getForm();
    int rs = spdao.update(sp);
    if(rs > 0){
        JOptionPane.showMessageDialog(parentComponent, this, message, "Cập nhật sản phẩm thành công!");
        loadDataToTable();
    }
}
```

Viết xử lý cho nút “Xóa”:

```
private void btnXoaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    int rs = spdao.delete(masf, txtMaSP.getText());
    if(rs > 0){
        JOptionPane.showMessageDialog(parentComponent, this, message, "Xóa sản phẩm thành công!");
        loadDataToTable();
    }
}
```

Viết xử lý cho nút “Làm mới”:

```
private void btnLamMoiActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    clearForm();
}
```

*** YÊU CẦU NỘP BÀI:

Sv nén file bao gồm các yêu cầu đã thực hiện trên, nộp lms đúng thời gian quy định của giảng viên. Không nộp bài coi như không có điểm.

--- Hết ---