LAB 3: COLLECTION & MAP

MỤC TIÊU:

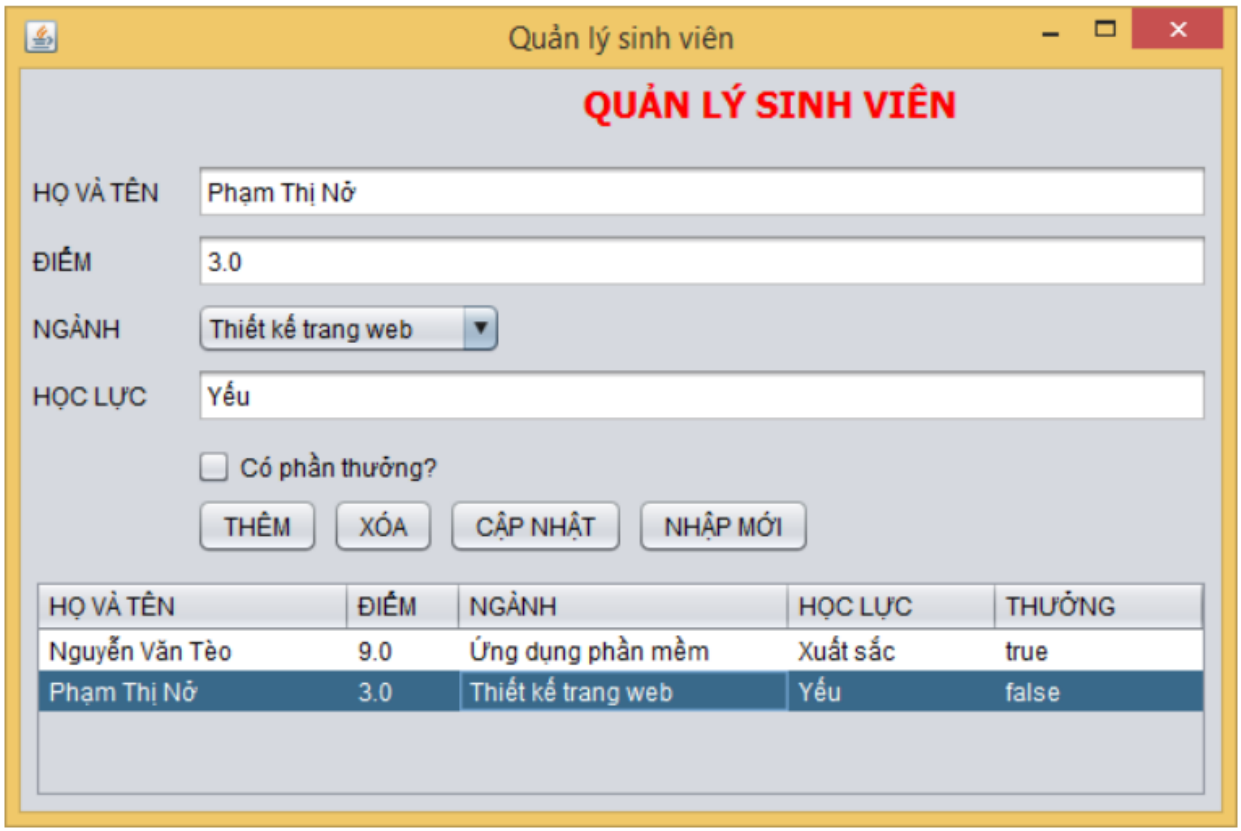
Kết thúc bài thực hành này bạn có khả năng củng cố lại kiến thức

- ✓ Sử dụng List và ArrayList
- ✓ Sử dụng JTable, JComboBox
- ✓ Sử dụng lớp tiện ích Collections
- ✓ Sử dụng Map và HashMap

PHẦN I

Bài 1 (2 điểm)

Dufng JFrame xây dựng một ứng dụng nhỏ để quản lý sinh viên với giao diện như sau



- ✓ Đặt tên theo qui ước cho các thành phần giao diện trên form
- ✓ Không cho phép nhập vào ô học lực
- ✓ Viết mã để
  - Đưa cửa sổ hiển thị giữa màn hình

```
setLocationRelativeTo(null);
```

  - Click nút [NHẬP MỚI] sẽ xóa trắng các ô nhập trên form và bỏ chọn CheckBox [Có phần thưởng]

Viết hàm makeNew() như sau:

```
//  Xóa nội dung các điều khiển để nhập thông tin SV mới
    public void makeNew() {
        txtHoTen.setText("");
        txtDiem.setText("");
        txtHocLuc.setText("");
        cboNganh.setSelectedIndex(0);
        chkPhanThuong.setSelected(false);
        txtHoTen.requestFocus();
    }
```

Mở sự kiện actonPerformed của button **Nhập mới** thêm các lên như sau:

```
    private void btnNewActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        makeNew();
        index = -1;
        tblListStudent.clearSelection();
    }
```

Chạy chương trình để test chức năng Nhập mới

Bài 2 (2 điểm)

1. Tạo lớp Student để quản lý thông tin sinh viên.
2. Mở Source của Jframe QLSV, khai báo 1 ArrayList, biến index và model toàn cục:

```
//  Khai báo các biến toàn cục
    List<Student> list = new ArrayList<>();

    int index = -1; //Luu index cua item duoc chon
    DefaultTableModel model; //Khai bao model cho table
```

3. Viết thêm 1 hàm tên duLieuMau() như sau:

```
// Khoi tao danh sach co san mot so SV
    private void duLieuMau() {
        list.add(new Student("Lê Hồng Hạnh", 8.5, "Ứng dụng phần mềm"));
        list.add(new Student("Trần Đình Ân", 5.0, "Thiết kế đồ hoa"));
        list.add(new Student("Lê Xuân Thảo", 9.5, "Thiết kế & lập trình Web"));
        list.add(new Student("Trần Đức Tuấn", 4.0, "Ứng dụng phần mềm"));
        list.add(new Student("Nguyễn Phi Hưng", 7.5, "Lập trình máy tính"));
    }
```

4. Viết thêm 1 hàm tên fillToTable() như sau:

```
//  Lấy dữ liệu từ list đưa vào model để hiển thị lên table trên Form
    public void fillToTable() {
        model = (DefaultTableModel) tblListStudent.getModel();
        model.setRowCount(0);
        for (Student st : list) {
            Vector row = new Vector();
            row.add(st.name);
            row.add(st.marks);
            row.add(st.major);
            row.add(st.getGrade());
            row.add(st.isBonus());
            model.addRow(row);
        }
    }
```

Dùng Vector thay thé mảng object để tạo dòng dữ liệu addRow vào model:

- Thêm hàng vào model bằng cách dùng for duyệt từng phần tử của ArrayList.
- Lấy từng phần tử trong ArrayList, bóc tách dữ liệu mã, tên… và thêm vào Vector, cuối cùng add vector vào model như 1 hàng dữ liệu.

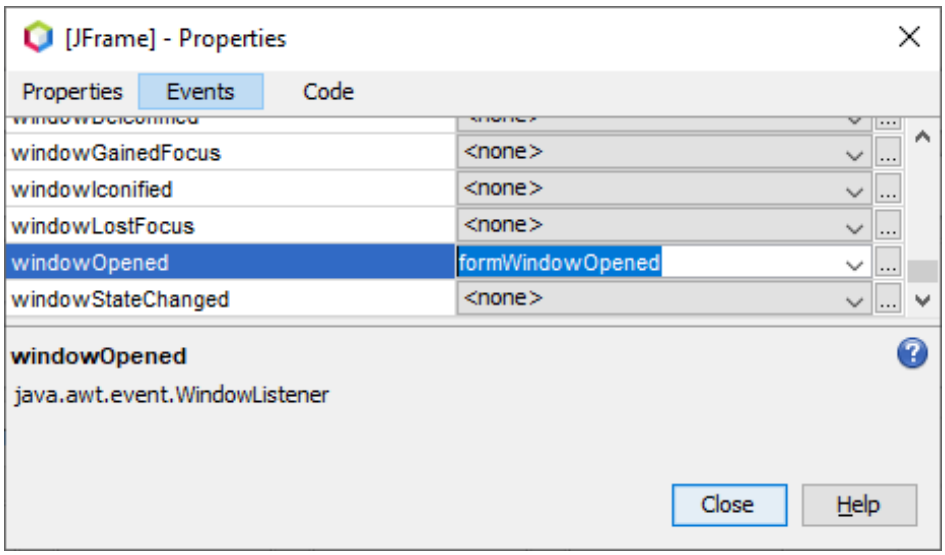
5. Viết tiếp hàm showDetail()

```
// Hiển thị dòng đang chọn trên jtable (dòng index) lên trên from
    void showDetail() {
        Student st = list.get(index);
        writeForm(st);
        tblListStudent.setRowSelectionInterval(index, index);
    }
}
```

6. Viết tiếp hàm writeForm()

```
//  Hiển thi thôn tin của SV đươc truyền vào qua tham số (Student st)
//  ra các điều khiển tren màn hình
    public void writeForm(Student st) {
        txtHoTen.setText(st.name);
        txtDiem.setText(st.marks.toString());
        txtHocLuc.setText(st.getGrade());
        cboNganh.setSelectedItem(st.major);
        if (st.isBonus()) {
            chkPhanThuong.setSelected(true);
        } else {
            chkPhanThuong.setSelected(false);
        }
    }
```

7. Mở sự kiện windowOpened của Jframe



thêm vào đoạn code sau để có dữ liệu mẫu trong ArrayList và hàm fillToTable để đổ dữ liệu từ ArrayList lên Jtable. Nếu danh sách có dữ liệu (size >0) thì chọn dòng đầu tiên trong list rồi iển thị thông tin chi tiết lên form qua hàm showDetail

```
private void formWindowOpened(java.awt.event.WindowEvent evt) {
    // TODO add your handling code here:
    duLieuMau();
    fillToTable();
    if (list.size() > 0) {
        index = 0;
        showDetail();
    }
}
```

8. Chạy chương trình để xem kết quả.

9. Viết tại sự kiện mouseClick của Jtable để khi chọn 1 dòng trong Table thì hiển thị thông tin dòng được chọn lên Form

```
private void tblListStudentMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    index = tblListStudent.getSelectedRow();
    Student st = list.get(index);
    writeForm(st);
}
```

10. Chạy chương trình để test kết quả.

11. Viết tiếp hàm readForm()

```
// Đọc thông tin từ các điều khiển trên from, tạo và trả về một Student
Student readForm() {
    Student st = new Student();
    st.name = txtHoTen.getText();
    st.marks = Double.parseDouble(txtDiem.getText());
    st.major = cboNganh.getSelectedItem().toString();
    return st;
}
```

12. Viết hàm addStudent()

```
//  Tạo mới sinh viên từ thông tin nhập trên form
    public void addStudent() {
        Student st = new Student();
        st.name = txtHoTen.getText();
        st.marks = Double.parseDouble(txtDiem.getText());
        st.major = (String) cboNganh.getSelectedItem();
//  Bổ sung sinh viên vào List<Student>
        list.add(st);
//  Hiển thị học lực và thưởng
        this.txtHocLuc.setText(st.getGrade());
        this.chkPhanThuong.setSelected(st.isBonus());
    }
```

13. Viết tại sự kiện actonPerformed của nút Thêm

```
    private void btnAddActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.addStudent();
        this.fillToTable();
        index = tblListStudent.getRowCount() - 1;
        tblListStudent.setRowSelectionInterval(index, index);
    }
```

14. Chạy chương trình để test chức năng Thêm

15. Viết hàm updateStudent()

```
//  Cập nhật thông tin sinh viên
    public void updateStudent() {
        if (index == -1) {
            JOptionPane.showMessageDialog(this, "Vui lòng chọn "
                    + "mẫu tin muốn cập nhật thông tin.");
        } else {
            Student st = readForm();
            list.set(index, st);
            this.fillToTable();
            tblListStudent.setRowSelectionInterval(index, index);
        }
    }
```

16. Viết tại sự kiện actonPerformed của nút Cập nhật

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

17. Chạy chương trình để test chức năng Cập nhật

18. Viết hàm removeStudent()

```
//  Xóa sinh viên đang chọn
    public void removeStudent() {
        if (index == -1) {
            JOptionPane.showMessageDialog(this, "Vui lòng chọn SV muốn xóa");
        } else {
            if (JOptionPane.showConfirmDialog(this, "Có thật sự muốn xóa "
                    + "sinh viên này không?") == JOptionPane.OK_OPTION) {
                list.remove(index);
                fillToTable();
                index = -1;
                makeNew();
                JOptionPane.showMessageDialog(this, "Xóa thành công!");
            }
        }
    }
```

19. Viết tại sự kiện actonPerformed của nút Xóa

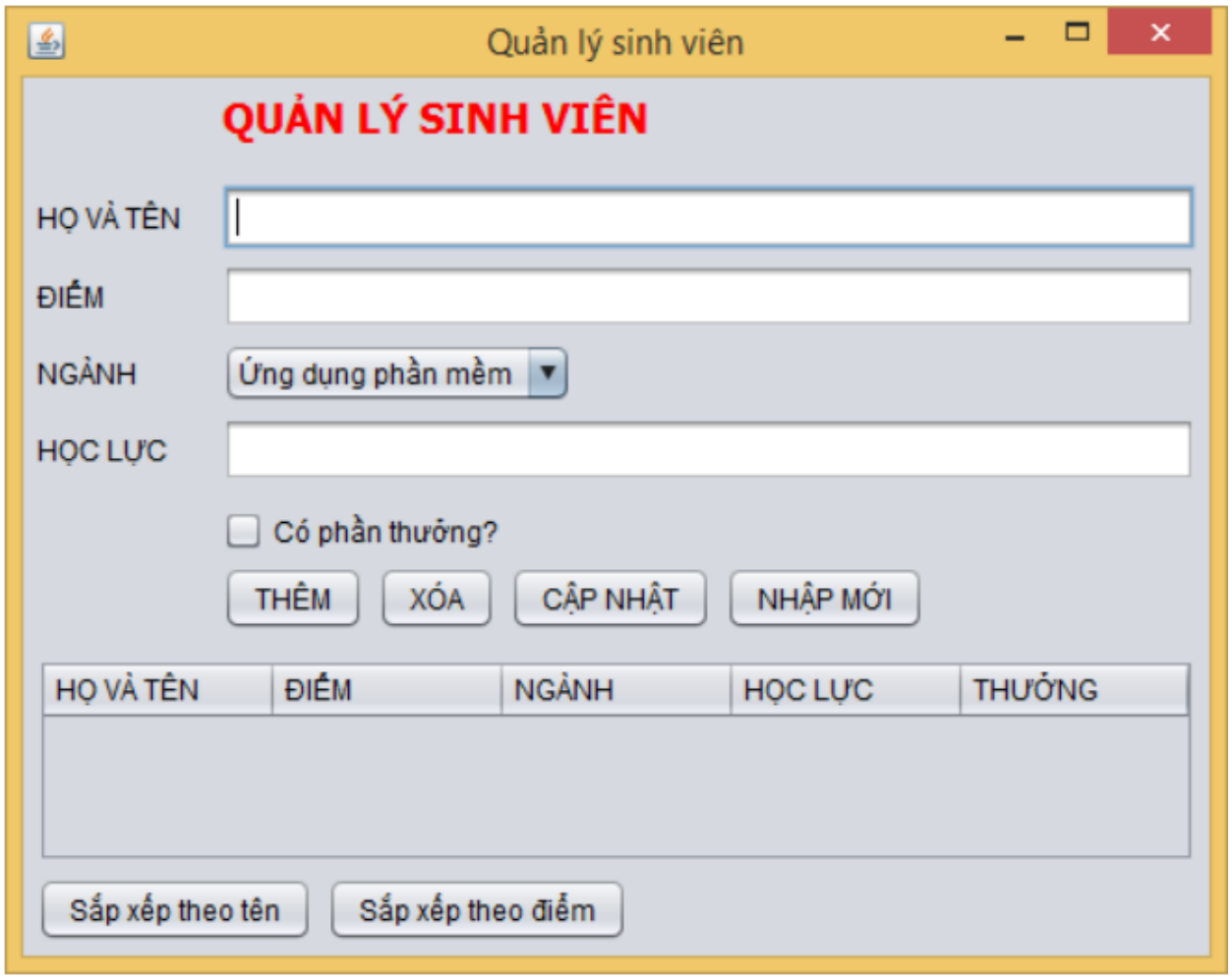
```
    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        removeStudent();
    }
```

20. Chạy chương trình để test chức năng Xóa

PHẦN II

Bài 3 (2 điểm)

Bổ sung 2 nút để sắp xếp danh sách sinh viên có giao diện như sau



21. Viết hàm orderByName()

```
//  Sắp xếp danh sách SV theo Họ tên tăng dần
    private void orderByName() {
        Comparator<Student> comName = new Comparator<Student>() {
            @Override
            public int compare(Student o1, Student o2) {
                return o1.name.compareTo(o2.name);
            }
        };
        Collections.sort(list, comName);
    }
```

22. Viết tại sự kiện actonPerformed của nút Sắp xếp theo tên

```
private void btnSXTheoTenActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    orderByName();
    fillToTable();
    index = 0;
    showDetail();
}
```

23. Chạy chương trình để test chức năng Sắp xếp theo tên

24. Viết hàm orderByMarks()

```
//  Sắp xếp danh sách SV theo điểm tăng dần
private void orderByMarks() {
    Comparator<Student> comMark = new Comparator<Student>() {
        @Override
        public int compare(Student o1, Student o2) {
            return o1.marks.compareTo(o2.marks);
        }
    };
    Collections.sort(list, comMark);
}
```

25. Viết tại sự kiện actonPerformed của nút Sắp xếp theo điểm

```
private void btnSXTheoDiemActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    orderByMarks();
    fillToTable();
    index = 0;
    showDetail();
}
```

26. Chạy chương trình để test chức năng Sắp xếp theo điểm

Bài 4 (2 điểm): Sẽ nộp cùng Lab4:

Bài 5 (2 điểm) Giảng viên cho thêm.

Xây dựng hệ thống điều hướng để có thể di chuyển đến các vị trí khác nhau trong danh sách.

Hết