# Bài Tập C - Sử Dụng Struct

## Bài 1: Quản lý Sinh Viên

Viết chương trình sử dụng struct để quản lý thông tin sinh viên bao gồm các thuộc tính: Tên, Tuổi, Điểm trung bình. Thực hiện:  
1. Nhập danh sách sinh viên.  
2. Hiển thị danh sách sinh viên.  
3. Tìm sinh viên có điểm trung bình cao nhất.

### Test Cases:

1. **Test Case 1:**

Input: [{'Tên': 'A', 'Tuổi': 20, 'Điểm trung bình': 8.5}, {'Tên': 'B', 'Tuổi': 21, 'Điểm trung bình': 9.0}]

Expected Output: B

1. **Test Case 2:**

Input: []

Expected Output: Danh sách rỗng.

1. **Test Case 3:**

Input: [{'Tên': 'C', 'Tuổi': 22, 'Điểm trung bình': 7.5}]

Expected Output: C

1. **Test Case 4:**

Input: [{'Tên': 'D', 'Tuổi': 20, 'Điểm trung bình': 6.5}, {'Tên': 'E', 'Tuổi': 21, 'Điểm trung bình': 6.5}]

Expected Output: D E

1. **Test Case 5:**

Input: [{'Tên': 'F', 'Tuổi': 22, 'Điểm trung bình': 10.0}]

Expected Output: F

## Bài 2: Quản lý Sản Phẩm

Viết chương trình sử dụng struct để quản lý thông tin sản phẩm bao gồm các thuộc tính: Tên sản phẩm, Giá, Số lượng. Thực hiện:  
1. Nhập danh sách sản phẩm.  
2. Hiển thị danh sách sản phẩm.  
3. Tìm sản phẩm có giá trị lớn nhất (Giá \* Số lượng).

### Test Cases:

1. **Test Case 1:**

Input: [{'Tên sản phẩm': 'SP1', 'Giá': 100, 'Số lượng': 5}, {'Tên sản phẩm': 'SP2', 'Giá': 150, 'Số lượng': 3}]

Expected Output: SP1

1. **Test Case 2:**

Input: []

Expected Output: Danh sách rỗng.

1. **Test Case 3:**

Input: [{'Tên sản phẩm': 'SP3', 'Giá': 200, 'Số lượng': 1}]

Expected Output: SP3

1. **Test Case 4:**

Input: [{'Tên sản phẩm': 'SP4', 'Giá': 50, 'Số lượng': 10}]

Expected Output: SP4

1. **Test Case 5:**

Input: [{'Tên sản phẩm': 'SP5', 'Giá': 100, 'Số lượng': 10}, {'Tên sản phẩm': 'SP6', 'Giá': 100, 'Số lượng': 10}]

Expected Output: SP5 SP6

## Bài 3: Quản lý Nhân Viên

Viết chương trình sử dụng struct để quản lý thông tin nhân viên bao gồm các thuộc tính: Tên, Chức vụ, Lương. Thực hiện:  
1. Nhập danh sách nhân viên.  
2. Hiển thị danh sách nhân viên.  
3. Tìm nhân viên có lương cao nhất.

### Test Cases:

1. **Test Case 1:**

Input: [{'Tên': 'NV1', 'Chức vụ': 'Trưởng phòng', 'Lương': 2000}, {'Tên': 'NV2', 'Chức vụ': 'Nhân viên', 'Lương': 1500}]

Expected Output: NV1

1. **Test Case 2:**

Input: []

Expected Output: Danh sách rỗng.

1. **Test Case 3:**

Input: [{'Tên': 'NV3', 'Chức vụ': 'Nhân viên', 'Lương': 1800}]

Expected Output: NV3

1. **Test Case 4:**

Input: [{'Tên': 'NV4', 'Chức vụ': 'Trưởng phòng', 'Lương': 2500}]

Expected Output: NV4

1. **Test Case 5:**

Input: [{'Tên': 'NV5', 'Chức vụ': 'Trưởng phòng', 'Lương': 3000}, {'Tên': 'NV6', 'Chức vụ': 'Nhân viên', 'Lương': 3000}]

Expected Output: NV5 NV6

## Bài 4: Quản lý Hóa Đơn

Viết chương trình sử dụng struct để quản lý thông tin hóa đơn bao gồm các thuộc tính: Mã hóa đơn, Tên khách hàng, Tổng tiền. Thực hiện:  
1. Nhập danh sách hóa đơn.  
2. Hiển thị danh sách hóa đơn.  
3. Tìm hóa đơn có tổng tiền lớn nhất.

### Test Cases:

1. **Test Case 1:**

Input: [{'Mã hóa đơn': 'HD1', 'Tên khách hàng': 'KH1', 'Tổng tiền': 5000}, {'Mã hóa đơn': 'HD2', 'Tên khách hàng': 'KH2', 'Tổng tiền': 3000}]

Expected Output: HD1

1. **Test Case 2:**

Input: []

Expected Output: Danh sách rỗng.

1. **Test Case 3:**

Input: [{'Mã hóa đơn': 'HD3', 'Tên khách hàng': 'KH3', 'Tổng tiền': 7000}]

Expected Output: HD3

1. **Test Case 4:**

Input: [{'Mã hóa đơn': 'HD4', 'Tên khách hàng': 'KH4', 'Tổng tiền': 1000}]

Expected Output: HD4

1. **Test Case 5:**

Input: [{'Mã hóa đơn': 'HD5', 'Tên khách hàng': 'KH5', 'Tổng tiền': 8000}, {'Mã hóa đơn': 'HD6', 'Tên khách hàng': 'KH6', 'Tổng tiền': 8000}]

Expected Output: HD5 HD6