

Assignment

MÔN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

MÃ MÔN: 503005

Đề bài: Module Quản lý dịch vụ lưu trú

Version: 1.1 – Ngày: 09/05/2024

(Sinh viên đọc kỹ tất cả hướng dẫn trước khi làm bài)

I. Giới thiệu bài toán

Một tập đoàn kinh doanh dịch vụ lưu trú cần phần mềm để quản lý tất cả dịch vụ lưu trú của họ để cung cấp cho người dùng có thể đặt dịch vụ. Tập đoàn này sở hữu rất nhiều loại hình dịch vụ lưu trú (Accommodation) khác nhau từ dịch vụ thông thường (Common Accommodation) như Homestay, Khách sạn (Hotel), Resort (Resort) đến dịch vụ cao cấp, sang trọng (Luxury Accommodation) như Villa hay du thuyền (Cruise Ship). Trong bài này, sinh viên sẽ lập trình một số chức để quản lý quá trình tìm kiếm và đặt dịch vụ lưu trú.

Các chức năng mà sinh viên phải thực hiện là:

- Cài đặt các lớp liên quan đến các loại dịch vụ phòng.
- Đọc thông tin về các loại phòng.
- Xử lý các yêu cầu tìm kiếm và đặt phòng.

II. Tài nguyên cung cấp

Source code được cung cấp sẵn bao gồm các file:

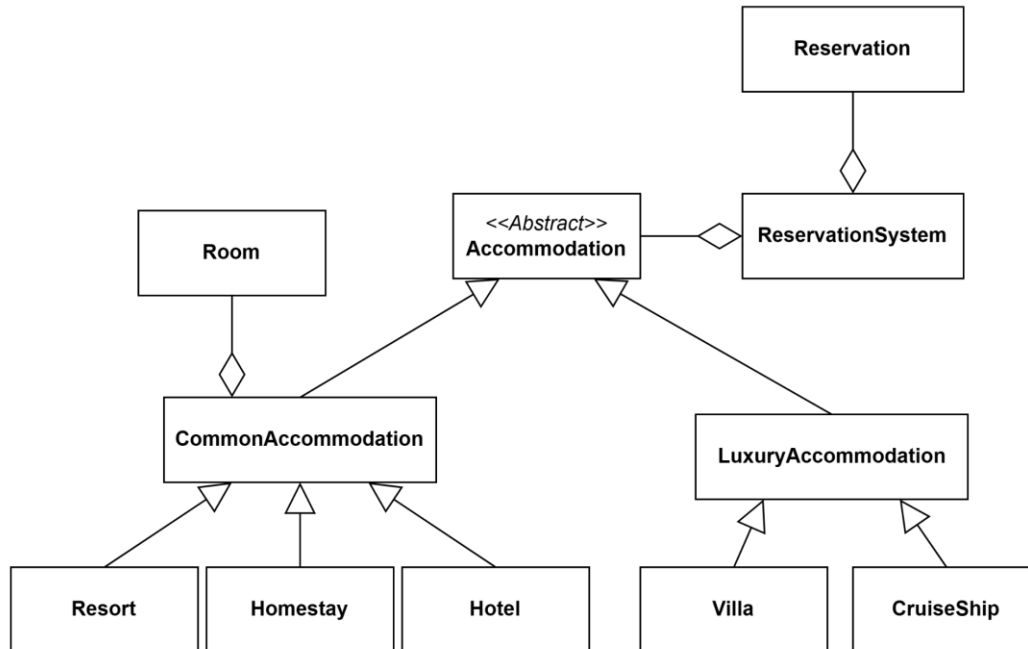
- File đầu vào và kết quả mong muốn:
 - o Folder *input* gồm 5 file:
 - *accommodation.csv*: chứa thông tin của tất cả các địa điểm lưu trú.
 - *room_type.csv*: chứa thông tin loại phòng của dịch vụ lưu trú thường (Common Accommodation).
 - *room_in_acc.csv*: chứa thông tin danh sách phòng của từng địa điểm trong dịch vụ lưu trú thường.
 - *reservation_3.csv*: chứa các yêu cầu đặt phòng phục vụ cho Yêu cầu 3.
 - *reservation_5.csv*: chứa các yêu cầu đặt phòng phục vụ cho Yêu cầu 5.
 - o Folder *expected_output* gồm: 7 file *Req1.txt*, *Req2.txt*, *Req3.txt*, *Req4_1.txt*, *Req4_2.txt*, *Req5.txt* chứa kết quả mẫu của các Yêu cầu từ 1 đến 5 trong bài và 1 file *reservation_5.csv* là kết quả mẫu sau khi chạy thử Yêu cầu 5.
 - File mã nguồn:
 - o *Main.java*: tạo đối tượng, gọi kiểm thử tương ứng Yêu cầu 1 đến Yêu cầu 5.
 - o *Reservation.java*: chứa lớp **Reservation** được định nghĩa sẵn phục vụ cho Yêu cầu 5.
- Các file này sinh viên không được chỉnh sửa.

- *Accommodation.java*, *CommonAccommodation.java*, *LuxuryAccommodation.java*, *Room.java*, *Hotel.java*, *Homestay.java*, *Resort.java*, *Villa.java*, *CruiseShip.java*: chứa các lớp được định nghĩa sẵn để tạo ra đối tượng tương ứng, sinh viên lập trình thêm vào các file này theo yêu cầu đề bài.
- *ReservationSystem.java*: chứa lớp **ReservationSystem** được định nghĩa sẵn thuộc tính *accommodations* để chứa danh sách các dịch vụ lưu trú được quản lý. Phương thức khởi tạo và phương thức *get* được cung cấp sẵn. Sinh viên sẽ hiện thực thêm vào các phương thức còn trống trong file này và không chỉnh sửa phương thức có sẵn.

III. Trình tự thực hiện bài

- Sinh viên tải file tài nguyên được cung cấp sẵn và giải nén.
- Sinh viên hiện thực lớp **Accommodation**, **Room**, **CommonAccommodation**, **LuxuryAccommodation**, **Homestay**, **Hotel**, **Resort**, **Villa** và **CruiseShip** theo mô tả lớp trong phần tiếp theo.
- Sau khi hoàn thành các lớp trên, sinh viên lập trình thêm vào các phương thức còn trống trong lớp **ReservationSystem** theo mô tả và yêu cầu trong các phần tiếp theo.
- Sau khi hiện thực các phương thức trong lớp **ReservationSystem**, sinh viên biên dịch và chạy với phương thức **main** trong file *Main.java* đã gọi sẵn các phương thức. Sinh viên thực hiện các yêu cầu ở mục dưới và so sánh kết quả của mình với kết quả trong folder *expected_output* đã được cung cấp sẵn.
- Đối với Yêu cầu 5 sinh viên đọc kỹ lớp **Reservation** được cung cấp sẵn để kết hợp với phần định nghĩa của sinh viên để hiện thực yêu cầu này.
- Đối với các yêu cầu sinh viên không làm được vui lòng không xóa và phải đảm bảo chương trình hoạt động được với phương thức **main** trong *Main.java* được cung cấp sẵn.
- Đường link Google Drive gửi đề này cho sinh viên sẽ chứa kèm một file *version.txt*, sinh viên nên thường xuyên vào xem file này, nếu thấy có nội dung mới thì nên đọc kỹ mô tả và tải lại đề mới nhất. File này được dùng để thông báo nội dung chỉnh sửa và ngày chỉnh sửa trong trường hợp đề bị sai hoặc lỗi.

IV. Mô tả các lớp trong bài



- Lớp **Accommodation** là lớp trừu tượng chứa toàn bộ các thông tin cơ bản của một dịch vụ lưu trú trong hệ thống, các lớp **CommonAccommodation** và **LuxuryAccommodation** là hai lớp con kế thừa lớp **Accommodation** thể hiện có hai loại dịch vụ là dịch vụ thông thường và dịch vụ sang trọng. Đối với mỗi loại chỗ ở sẽ bao gồm nhiều loại khác nhau cho khách hàng lựa chọn. Các lớp **Resort**, **Homestay** và **Hotel** là các lớp con kế thừa lớp **CommonAccommodation**, các lớp **Villa** và **CruiseShip** là lớp con kế thừa lớp **LuxuryAccommodation**. Lớp **Room** là lớp chỉ liên quan đến lớp **CommonAccommodation**, chứa thông tin liên quan đến các phòng tại các chỗ ở thông thường. Lớp **ReservationSystem** là lớp thực hiện các thao tác xử lý và tương tác với lớp **Accommodation** trong hệ thống.
- Mô tả các lớp như sau:
 - Các thuộc tính được đánh số trong các lớp sẽ do sinh viên tự đặt tên. Tất cả các thuộc tính của lớp đều phải định nghĩa với **access modifier** là **private** hoặc **protected**. Trong quá trình hiện thực các lớp, sinh viên được phép định nghĩa thêm các phương thức **get()** để lấy dữ liệu thuộc tính.
 - Lớp **Accommodation** - Nơi lưu trú

<<Abstract>> Accommodation
attr1: int # attr2: String # attr3: String # attr4: String
+ Accommodation(attr1: int, attr2: String, attr3: String, attr4: String)

- Gồm 04 thuộc tính:
 - Thuộc tính 1: int - mã số định danh của nơi lưu trú
 - Thuộc tính 2: String - tên của nơi lưu trú
 - Thuộc tính 3: String - số nhà và tên đường của nơi lưu trú
 - Thuộc tính 4: String - thành phố nơi lưu trú tọa lạc
- Phương thức:
 - Phương thức khởi tạo đầy đủ tham số theo thứ tự của 04 thuộc tính trên.
 - Các phương thức phụ trợ khác như: getter, setter, ... sinh viên tự định nghĩa thêm.

○ Lớp **Room** - Phòng ở

Room
- attr1: int - attr2: String - attr3: int - attr4: String - attr5: int - attr6: int - attr7: double - attr8: double
+ Room(attr1: int, attr2: String, attr3: int, attr4: String, attr5: int, attr6: int, attr7: double, attr8: double) + toString(): String

- Gồm 08 thuộc tính:
 - Thuộc tính 1: int - Mã định danh
 - Thuộc tính 2: String - Tên phòng
 - Thuộc tính 3: int - Số lượng lầu
 - Thuộc tính 4: String - Loại phòng
 - Thuộc tính 5: int - Số lượng giường
 - Thuộc tính 6: int - Số lượng người tối đa có thể lưu trú tại phòng
 - Thuộc tính 7: double - Diện tích sàn
 - Thuộc tính 8: double - Chi phí phòng một đêm

- Phương thức:
 - Phương thức **Room(attr1: int, attr2: String, attr3: int, attr4: String, attr5: int, attr6: int, attr7: double, attr8: double)**: Phương thức khởi tạo đầy đủ tham số có thứ tự của 08 thuộc tính trên
 - Phương thức **toString(): String**: theo định dạng **Room [mã định danh, tên phòng, số lượng lầu, loại phòng, số lượng giường, số lượng người tối đa lưu trú, diện tích sàn, chi phí phòng một đêm]**
 - Các phương thức phụ trợ như: getter, setter, ... sinh viên tự định nghĩa thêm.

○ Lớp **CommonAccommodation** – Nơi lưu trú thông thường

CommonAccommodation
attr5: ArrayList<Room> # attr6: float
+ CommonAccommodation(attr1: int, attr2: String, attr3: String, attr4: String, attr5: ArrayList<Room>, attr6: float) + CommonAccommodation(attr1: int, attr2: String, attr3: String, attr4: String, attr6: float)

- Gồm 02 thuộc tính:
 - Thuộc tính 5: ArrayList<Room> Danh sách các phòng trong nơi lưu trú
 - Thuộc tính 6: float - Hệ số đánh giá nơi lưu trú
- Phương thức:
 - Phương thức **CommonAccommodation(attr1: int, attr2: String, attr3: String, attr4: String, attr5: ArrayList<Room>, attr6: float)**: Phương thức khởi tạo đầy đủ tham số với đầu vào là các tham số tương ứng theo thứ tự của lớp cha (**Accommodation**) và hai phương thức của lớp con.
 - Phương thức **CommonAccommodation(attr1: int, attr2: String, attr3: String, attr4: String, attr6: float)**: Phương thức khởi tạo không chứa tham số danh sách các phòng trong nơi lưu trú.
 - Các phương thức phụ trợ như: getter, setter, ... sinh viên tự định nghĩa thêm.

○ Lớp **Resort** – Khu nghỉ mát

Resort
- attr7: int - attr8: boolean
+ Resort(attr1: int, attr2: String, attr3: String, attr4: String, attr5: ArrayList<Room>, attr6: float, attr7: int, attr8: boolean) + Resort(attr1: int, attr2: String, attr3: String, attr4: String, attr6: float, attr7: int, attr8: boolean) + toString(): String

- Gồm 02 thuộc tính:
 - Thuộc tính 7: int - Số sao thể hiện chất lượng của khu nghỉ mát
 - Thuộc tính 8: boolean - Có phục vụ bể bơi cá nhân không (true là có, false là không).
- Phương thức:
 - Phương thức **Resort(attr1: int, attr2: String, attr3: String, attr4: String, attr5: ArrayList<Room>, attr6: float, attr7: int, attr8: boolean)**: Phương thức khởi tạo đầy đủ tham số với các tham số đầu vào từ tham số attr1 - attr6 là các tham số cho phương thức khởi tạo ở lớp cha, 02 tham số còn lại theo thứ tự như trên.
 - Phương thức **Resort(attr1: int, attr2: String, attr3: String, attr4: String, attr6: float, attr7: int, attr8: boolean)**: Phương thức khởi tạo có tham số tuy nhiên không truyền vào tham số danh sách attr5 (danh sách các phòng).
 - Phương thức **toString(): String**: Theo định dạng **Resort [mã định danh, tên khu nghỉ mát, số sao, số nhà và tên đường, trạng thái có bể bơi cá nhân hay không, thành phố]**
 - Các phương thức phụ trợ khác như: getter, setter, ... sinh viên tự định nghĩa thêm.

○ Lớp **Homestay** - Lưu trú nhà dân

Homestay
+ Homestay(attr1: int, attr2: String, attr3: String, attr4: String, attr5: ArrayList<Room>, attr6: float) + Homestay(attr1: int, attr2: String, attr3: String, attr4: String, attr6: float) + toString(): String

- Phương thức:
 - Phương thức **Homestay(attr1: int, attr2: String, attr3: String, attr4: String, attr5: ArrayList<Room>, attr6: float)**: Phương thức khởi tạo

đầy đủ tham số với thứ tự truyền vào theo thứ tự của phương thức khởi tạo ở lớp cha.

- Phương thức **Homestay(attr1: int, attr2: String, attr3: String, attr4: String, attr6: float)**: Phương thức khởi tạo có tham số theo thứ tự của lớp cha, không truyền tham số attr5 (danh sách các phòng).
- Phương thức **toString()**: String: theo định dạng **Homestay [mã định danh, tên homestay, số nhà và tên đường, hệ số đánh giá, thành phố]**
- Các phương thức phụ trợ khác như: getter, setter, ... sinh viên tự định nghĩa thêm.

○ Lớp **Hotel** - Khách sạn

Hotel
- attr7: int
+ Hotel(attr1: int, attr2: String, attr3: String, attr4: String, attr5: ArrayList<Room>, attr6: float, attr7: int) + Hotel(attr1: int, attr2: String, attr3: String, attr4: String, attr6: float, attr7: int) + toString(): String

▪ Thuộc tính:

- Thuộc tính 07: int - Số sao chất lượng dịch vụ của khách sạn

▪ Phương thức:

- Phương thức **Hotel(attr1: int, attr2: String, attr3: String, attr4: String, attr5: ArrayList<Room>, attr6: float, attr7: int)**: Phương thức khởi tạo đầy đủ tham số với thứ tự truyền vào theo thứ tự của phương thức khởi tạo của lớp cha và thêm vào thuộc tính 07.
- Phương thức **Hotel(attr1: int, attr2: String, attr3: String, attr4: String, attr6: float, attr7: int)**: Phương thức khởi tạo có tham số không truyền tham số attr5 (danh sách các phòng).
- Phương thức **toString()**: String: theo định dạng **Hotel [mã định danh, số sao chất lượng dịch vụ, tên khách sạn, số nhà và tên đường, thành phố]**
- Các phương thức phụ trợ khác như: getter, setter, ... sinh viên tự định nghĩa thêm.

○ Lớp **LuxuryAccommodation** - dịch vụ lưu trú sang trọng

LuxuryAccommodation
attr5: boolean # attr6: boolean # attr7: boolean # attr8: boolean # attr9: int # attr10: int
+ LuxuryAccommodation(attr1: int, attr2: String, attr3: String, attr4: String) + LuxuryAccommodation(attr1: int, attr2: String, attr3: String, attr4: String, attr5: boolean, attr6: boolean, attr7: boolean, attr8: boolean, attr9: int, attr10: int)

- Thuộc tính:
 - Thuộc tính 5: boolean - Có phục vụ bể bơi cá nhân không (true là có, false là không)
 - Thuộc tính 6: boolean - Có phục vụ thức uống chào mừng không (true là có, false là không)
 - Thuộc tính 7: boolean - Có phục vụ miễn phí bữa sáng không (true là có, false là không)
 - Thuộc tính 8: boolean - Có phục vụ phòng thể hình không (true là có, false là không)
 - Thuộc tính 9: int - Số lượng người tối đa có thể phục vụ
 - Thuộc tính 10: int - Chi phí cho một đêm
- Phương thức:
 - Phương thức **LuxuryAccommodation(attr1: int, attr2: String, attr3: String, attr4: String)**: phương thức khởi tạo có tham số.
 - Phương thức **LuxuryAccommodation(attr1: int, attr2: String, attr3: String, attr4: String, attr5: boolean, attr6: boolean, attr7: boolean, attr8: boolean, attr9: int, attr10: int)**: Phương thức khởi tạo đầy đủ tham số với thứ tự tham số như lớp cha và thứ tự tham số như 06 thuộc tính trên.
 - Các phương thức phụ trợ khác như: getter, setter, ... sinh viên tự định nghĩa thêm.
- Lớp **Villa** - Biệt thự

Villa
+ Villa(attr1: int, attr2: String, attr3: String, attr4: String, attr5: boolean, attr6: boolean, attr7: boolean, attr8: boolean, attr9: int, attr10: int) + toString(): String

- Phương thức:
 - Phương thức Villa(attr1: int, attr2: String, attr3: String, attr4: String, attr5: boolean, attr6: boolean, attr7: boolean, attr8: boolean, attr9: int, attr10: int)
 - Phương thức toString(): String - Theo định dạng **Villa [mã định danh, tên biệt thự, số nhà và tên đường, thành phố, có thể phục vụ hồ bơi cá nhân không, có thể phục vụ thức uống chào mừng không, có thể phục vụ bữa sáng miễn phí không, có phục vụ phòng thể hình không, số lượng người tối đa có thể phục vụ, chi phí cho một đêm]**
 - Các phương thức phụ trợ khác như: getter, setter, ... sinh viên tự định nghĩa thêm.

○ Lớp **CruiseShip** - Tàu du lịch

CruiseShip
- attr11: boolean
+ CruiseShip(attr1: int, attr2: String, attr3: String, attr4: String, attr5: boolean, attr6: boolean, attr7: boolean, attr8: boolean, attr9: int, attr10: int, attr11: boolean) + toString(): String

- Thuộc tính:
 - Thuộc tính 11: boolean - Có phục vụ quán bar riêng tư không (true là có, false là không)
- Phương thức:
 - Phương thức **CruiseShip(attr1: int, attr2: String, attr3: String, attr4: String, attr5: boolean, attr6: boolean, attr7: boolean, attr8: boolean, attr9: int, attr10: int, attr11: boolean)**: khởi tạo đầy đủ tham số với thứ tự tham số theo lớp cha sau đó là tham số của lớp con.
 - Phương thức toString(): String - theo định dạng **CruiseShip [mã định danh, tên tàu, số nhà và tên đường, thành phố, có phục vụ quán bar riêng tư không, có phục vụ hồ bơi cá nhân không, có phục vụ thức uống chào**

mừng không, có phục vụ bữa sáng miễn phí không, có phục vụ phòng thể hình không]

- Các phương thức phụ trợ khác như: getter, setter, ... sinh viên tự định nghĩa thêm.

○ Lớp **Reservation** - Đặt phòng

Reservation
- reservationId: int - accId: int - roomId: int - checkin: Date - checkout: Date
+ Reservation(reservationId: int, accId: int, roomId: int, checkin: Date, checkout: Date) + toString(): String

▪ Thuộc tính:

- Thuộc tính reservationId: int - Mã định danh đặt phòng
- Thuộc tính accId: int - Mã định danh nơi lưu trú
- Thuộc tính roomId: int - Mã định danh phòng
- Thuộc tính checkin: Date - Thời gian checkin
- Thuộc tính checkout: Date - Thời gian checkout

▪ Phương thức:

- Phương thức **Reservation(reservationId: int, accId: int, roomId: int, checkin: Date, checkout: Date)**: Phương thức khởi tạo đầy đủ tham số theo thứ tự như trên.
- Phương thức **toString(): String** - theo định dạng **Reservation [mã định danh đặt phòng, mã định danh nơi lưu trú, mã định danh phòng, thời gian checkin, thời gian checkout]**
- Các phương thức phụ trợ khác như: getter, setter, ... sinh viên tự định nghĩa thêm.

○ Lớp **ReservationSystem** - Hệ thống đặt phòng

ReservationSystem
- accommodations: ArrayList<Accommodation>
+ ReservationSystem(accPath: String, roomPath: String, roomOfAccPath: String) + loadAccommodations(accPath: String, roomPath: String, roomOfAccPath: String): ArrayList<Accommodation> + searchForRoom(city: String, numOfPeople: int): ArrayList<Accommodation> + searchForRoomByRange(reservationPath: String, priceFrom: double, priceTo: double, checkin: Date, checkout: Date, city: String, numOfPeople: int): ArrayList<Accommodation> + searchInAdvance(city: String, numOfPeople: Integer, roomType: String, privatePool: Boolean, starQuality: Double, freeBreakfast: Boolean, privateBar: Boolean): ArrayList<Accommodation> + performReservation(acc: Accommodation, room: Room, checkin: Date, checkout: Date): double

- Thuộc tính:
 - accommodations: ArrayList<Accommodation> - Chứa thông tin về các dịch vụ lưu trú có trong hệ thống.
- Phương thức:
 - Phương thức **ReservationSystem(accPath: String, roomPath: String, roomOfAccPath: String)**: Phương thức khởi tạo có tham số bao gồm đường dẫn đến file chứa thông tin các phòng, đường dẫn đến file chứa thông tin các nơi lưu trú, đường dẫn đến file chứa thông tin phòng tương ứng của từng nơi lưu trú
- Lớp **Main** – Chứa phương thức *main* kiểm thử các yêu cầu.

V. Mô tả file input và file output

- File input có tên *accommodation.csv* chứa danh sách các nơi lưu trú có trong hệ thống, mỗi dòng tương ứng với mô tả về một dịch vụ lưu trú khác nhau, mỗi thuộc tính được cách nhau bằng dấu phẩy (“,”). Mỗi dòng thể hiện cho một dịch vụ lưu trú, có số lượng thuộc tính khác nhau và tương ứng với từng dịch vụ.

- **Homestay:**

mã định danh, tên nơi lưu trú, số nhà và tên đường, thành phố, hệ số đánh giá

1,Cozy Homestay,123 Main St,City A,4.8

6,Green Garden Homestay,543 Park Ln,City F,4.0

- **Resort:**

Mã định danh, tên khu nghỉ mát, số nhà và địa chỉ, thành phố, số sao chất lượng dịch vụ, có bể bơi cá nhân không, hệ số đánh giá

3,Beach Resort,789 Ocean Ave,City C,4,yes,4.8

8,Sunset Resort,654 Beach Rd,City H,3,yes,4.5

- **Hotel:**

Mã định danh, tên khách sạn, số nhà và địa chỉ, thành phố, số sao chất lượng dịch vụ, hệ số đánh giá

2,Luxury Hotel,456 Elm St,City B,5,4.9

7,Grand Palace Hotel,789 King St,City G,4,4.9

- **Villa**

mã định danh, tên biệt thự, số nhà và tên đường, thành phố, có bể bơi cá nhân hay không, có thức uống chào mừng hay không, có bữa sáng miễn phí hay không, có thể phòng thể hình hay không, số lượng người, chi phí một đêm

4,Mountain Villa,101 Forest Rd,City D,yes,yes,yes,no,20,330

14,Cozy Corner Villa,567 Cozy St,City N,yes,yes,yes,no,10,320

- **CruiseShip**

mã định danh, tên tàu du lịch, số nhà và địa chỉ, thành phố, có bể bơi cá nhân hay không, có thức uống chào mừng hay không, có bữa sáng miễn phí hay không, có thể phòng thể hình hay không, có quầy pha chế cá nhân hay không, số lượng người, chi phí một đêm

5,CruiseShip Villa,111 River Blvd,City E,no,yes,yes,yes,yes,30,500

20,Tropical Oasis CruiseShip,210 Palm Tree Ln,City T,yes,yes,yes,no,yes,20,700

- File input có tên *room_type.csv* chứa danh sách thông tin các phòng trong toàn bộ nơi lưu trú, mỗi dòng thể hiện cho một phòng, mỗi thuộc tính được cách nhau bằng dấu phẩy (“,”).

mã định danh, tên phòng, số lượng lầu, loại phòng, số lượng giường, số lượng người, diện tích sàn, chi phí một đêm

1,Single Room,1,Standard,1,1,20,50

2,Double Room,2,Standard,2,2,30,80

- File input có tên *room_in_accommodation.csv* chứa thông tin các phòng tương ứng của từng nơi lưu trú. Mỗi dòng thể hiện thông tin phòng nào thuộc nơi lưu trú nào và được cách nhau bằng dấu phẩy (“,”).

mã định danh nơi lưu trú, mã định danh phòng

1,1

1,2

1,3

- File input có tên *reservation_5.csv* chứa thông tin đặt phòng phục vụ cho yêu cầu 5. Mỗi dòng biểu diễn cho một sự kiện đặt phòng thành công, mỗi thuộc tính được cách nhau bằng dấu phẩy (“,”). Có hai loại thông tin đặt phòng:

- Đối với các dịch vụ lưu trú thông thường: *Mã đặt phòng, mã định danh nơi lưu trú, mã định danh phòng, thời gian checkin, thời gian checkout*

2,1,2,1654500000,1654700000

3,1,3,1654900000,1655200000

- Đối với các dịch vụ lưu trú sang trọng: *Mã đặt phòng, mã định danh nơi lưu trú, thời gian checkin, thời gian checkout*

1,12,1654018800,1654200000

7,14,1656400000,1656600000

- Hướng dẫn xử lý dữ liệu thời gian: Trong lớp **ReservationSystem** đã cung cấp phương thức **public long diffBetweenDays(long dateStart, long dateEnd)** để tính toán khoảng cách ngày giữa hai khoảng thời gian.
- Thư mục *expected_output* gồm có 07 file ứng với kết quả mẫu cho các Yêu cầu từ 1 đến 5. Các file output là danh sách Accommodation thì mỗi dòng trong file *.txt ứng với một đối tượng, có định dạng ghi ra theo theo phương thức **toString()** của lớp **tương ứng**. Riêng Yêu cầu 4 có 2 file output ứng với kết quả kiểm thử 2 testcase mẫu. File *reservation_5.csv* là trạng thái của file sau khi thực hiện Yêu cầu 5.

Lưu ý:

- Sinh viên có thể tự thêm dữ liệu vào file input để thử nhiều trường hợp khác nhau nhưng lưu ý thêm dữ liệu phải đúng định dạng đã được nêu bên trên.
- Sinh viên nên đọc kỹ nội dung trong các phương thức **main** để xác định hướng hiện thực.
- Sinh viên có thể thêm một số thao tác vào phương thức **main** để kiểm các phương thức đã định nghĩa tuy nhiên đảm bảo bài làm của sinh viên **phải chạy được trên phương thức main ban đầu đã cung cấp sẵn**.
- Sinh viên có thể thêm phương thức mới để phục vụ bài làm tuy nhiên bài làm của sinh viên phải chạy được với các file *Main.java* đã được cung cấp sẵn.
- Sinh viên tuyệt đối không đặt các đường dẫn tuyệt đối khi định nghĩa phương thức liên quan đến đọc file. Nếu sinh viên tự ý đặt đường dẫn tuyệt đối dẫn đến quá trình chấm không đọc được file để chấm thì tương đương với bài làm biên dịch lỗi.
- **Tuyệt đối tuân thủ theo tên của các phương thức được yêu cầu cụ thể trong đề bài.**

VI. Yêu cầu

Ngoại trừ thư viện đã được import trong file cung cấp sẵn, sinh viên không được thêm thư viện khác. Sinh viên thực hiện bài tập lớn trên Java 11 hoặc Java 8. Sinh viên không được dùng kiểu dữ liệu *var*. Bài làm của sinh viên sẽ được chấm trên Java 11, sinh viên tự chịu trách nhiệm tất cả các lỗi phát sinh nếu dùng các phiên bản Java khác.

Sinh viên được phép định nghĩa thêm phương thức để hỗ trợ bài làm nhưng chỉ định nghĩa thêm trong các file được yêu cầu nộp. **Sinh viên không tự ý thay đổi tên phương thức và tham số truyền vào của các phương thức có sẵn và các phương thức được yêu cầu trong đề bài.**

Các **thuộc tính được đánh số** trong phần **Mô tả các lớp trong bài**, khi sinh viên định nghĩa sẽ tự đặt tên cho thuộc tính, chỉ cần đảm bảo đúng kiểu dữ liệu theo mô tả và *access modifier* của tất cả các thuộc tính phải là *private/protected*.

Trong quá trình lập trình, sinh viên **KHÔNG** được phép dùng ký tự tiếng Việt có dấu trong bất kỳ vị trí nào của bài nào. Do bài làm chấm trên máy Windows, nếu sinh viên cố ý dùng tiếng Việt có dấu trong bài làm gây lỗi biên dịch thì bài làm 0 điểm.

Sinh viên được phép điều chỉnh code các file *Main.java* và thêm các dữ liệu mới vào các file trong thư mục *input* (tuân theo format được mô tả trong đề) để kiểm thử thêm nhiều trường hợp khác sau khi đã kiểm tra kết quả với code có sẵn trong các file này.

Sau khi hiện thực tất cả các lớp có liên quan, sinh viên bắt đầu thực hiện Yêu cầu 1 đến Yêu cầu 5.

1. YÊU CẦU 1 (2 điểm)

Từ Yêu cầu 1 đến Yêu cầu 5 sẽ liên quan đến dữ liệu đọc từ các file trong thư mục *data* cung cấp. Sinh viên sẽ định nghĩa các phương thức trong lớp **ReservationSystem** và dùng lớp **Main** để kiểm tra kết quả bài làm.

Hiện thực phương thức

public ArrayList<Accommodation> loadAccommodation(String accPath, String roomPath, String roomOfAccPath)

Phương thức này sẽ đọc file *accommodation.csv*, *room.csv*, *room_in_acc.csv* lần lượt theo đường dẫn của tham số *accPath*, *roomPath* và *roomOfAccPath* truyền vào để đọc thông tin dịch vụ lưu trú và các phòng của một số loại dịch vụ lưu trú quản lý theo phòng vào danh sách *accommodations*.

Sinh viên hiện thực phương thức trên, biên dịch file **Main.java** và chạy lệnh *java Main 1* để ghi ra kết quả file *Req1.txt* vào thư mục *output*. Sinh viên xem kết quả file này và file *Req1.txt* trong folder *expected_output* mẫu được cung cấp sẵn để so sánh kết quả.

Lưu ý: Đây là phương thức sinh viên phải định nghĩa được mới bắt đầu tính điểm cho các Yêu cầu từ 1 đến 4, nếu không được file thành danh sách các đối tượng **Accommodation** thì các yêu cầu bên dưới không được tính điểm.

Các yêu cầu bên dưới thao tác trực tiếp trên dữ liệu đã đọc từ file.

2. YÊU CẦU 2 (2 điểm)

Hiện thực phương thức

public ArrayList<Accommodation> searchForRoom(String city, int numOfPeople)

Trả về danh sách dịch vụ lưu trú ở tại vị trí *city* truyền vào và có sức chứa của dịch vụ lưu trú phù hợp (số lượng người tối đa có thể phục vụ phải lớn hơn hoặc bằng) với số người *numOfPeople* theo yêu cầu truyền vào. Kết quả danh sách các địa điểm phù hợp cần sắp xếp lần lượt theo 02 tiêu chí:

- Các dịch vụ sang trọng (luxury accommodation) được đặt trước các dịch vụ thông thường (common accommodation).
- Sắp xếp tăng dần theo tên của địa điểm theo thứ tự bảng chữ cái alphabet.

Sinh viên hiện thực phương thức trên, biên dịch file **Main.java** và chạy lệnh *java Main 2* để ghi ra kết quả file *Req2.txt* vào thư mục *output*. Sinh viên xem kết quả file này và file *Req2.txt* trong folder *expected_output* mẫu được cung cấp sẵn để so sánh kết quả.

3. YÊU CẦU 3 (2 điểm)

Hiện thực phương thức

public ArrayList<Accommodation> searchForRoomByRange(String reservationPath, double priceFrom, double priceTo, Date checkin, Date checkout, String city, int numOfPeople)

Trả về danh sách dịch vụ lưu trú thỏa điều kiện theo các tham số truyền vào. Để hiện thực được yêu cầu này, sinh viên đọc file *reservation_3.csv* được đọc từ đường dẫn *reservationPath* của tham số để lấy thông tin các phòng đã được đặt trước đó. Phương thức **searchForRoomByRange()** trả về danh sách các dịch vụ lưu trú có giá thuê 1 đêm lớn hơn hoặc bằng *priceFrom* và bé hơn hoặc bằng *priceTo*, dịch vụ phải trống trong khoảng ngày *checkin* và *checkout*, nằm ở địa điểm *city* và số lượng người tối đa lớn hơn **không quá 2** hoặc bằng so với số lượng người *numOfPeople* truyền vào từ tham số. Danh sách trả về sắp xếp giảm dần thứ tự bảng chữ cái alphabet theo tên của địa điểm.

Sinh viên hiện thực phương thức trên, biên dịch file **Main.java** và chạy lệnh *java Main 3* để ghi ra kết quả file *Req3.txt* vào thư mục *output*. Sinh viên xem kết quả file này và file *Req3.txt* trong folder *expected_output* mẫu được cung cấp sẵn để so sánh kết quả.

4. YÊU CẦU 4 (2 điểm)

Yêu cầu này không liên quan Yêu cầu 3 nên sinh viên không cần kiểm tra điều kiện đặt trước của phòng.

Hiện thực phương thức

public ArrayList<Accommodation> searchInAdvance(String city, Integer numOfPeople, String roomType, Boolean privatePool, Integer starQuality, Boolean freeBreakfast, Boolean privateBar)

trả về danh sách dịch vụ lưu trú thỏa điều kiện:

- *city*: tại địa điểm (không null).
- *numOfPeople*: số người theo yêu cầu (không null).
- *roomType*: loại phòng (Suite, Deluxe, ...) - điều kiện này chỉ có ở các dịch vụ lưu trú thông thường (có thể null).
- *privatePool*: True/False - có phục vụ hồ bơi cá nhân, điều kiện này chỉ có ở resort hoặc các dịch vụ lưu trú sang trọng (có thể null).
- *starQuality*: số sao - chỉ có ở khách sạn và resort (có thể null).
- *freeBreakfast*: True/False - có phục vụ bữa sáng miễn phí, điều kiện này chỉ có ở các dịch vụ lưu trú sang trọng (có thể null).
- *privateBar*: True/False - có phục vụ quán bar riêng tư, điều kiện này chỉ có ở tàu du lịch (có thể null).

Người dùng sẽ điền thông tin vào các tham số, tham số *city* và *numOfPeople* sẽ luôn yêu cầu có thông tin cụ thể. Với các tham số *roomType*, *privatePool*, *starQuality*, *freeBreakfast*, *privateBar* không xét thì sẽ điền là *null*, ngược lại sẽ điền đúng thông tin muốn lọc.

Trường hợp có trên 02 thông tin khác *city* và *numOfPeople* cần lọc mà không có loại hình dịch vụ nào đáp ứng được thì trả về danh sách rỗng.

Ví dụ:

- Với lời gọi phương thức **searchInAdvance("City A", 5, "Deluxe", null, 3, null, null)** thì sẽ trả về danh sách các dịch vụ là khách sạn tại City A, sức chứa lớn hơn hoặc bằng 5 người, có loại phòng Deluxe trong danh sách phòng và khách sạn 3 sao.
- Với lời gọi phương thức **searchInAdvance("City A", 5, null, null, null, False, null)** thì sẽ trả về danh sách các dịch vụ là dịch vụ lưu trú sang trọng tại City A, sức chứa lớn hơn hoặc bằng 5 người và không phục vụ bữa sáng. Vì chỉ có dịch vụ lưu trú sang trọng mới có thuộc tính phục vụ bữa sáng.
- Với lời gọi phương thức **searchInAdvance("City A", 5, "Deluxe", null, 3, null, True)** thì sẽ trả về danh sách rỗng vì không có loại hình dịch vụ nào vừa có loại phòng, vừa có số sao và vừa có điều kiện bữa sáng.

Lưu ý, sinh viên không tự ý thay đổi tham số của phương thức này. Các giá trị không xét thì truyền vào là *null*.

Sinh viên hiện thực phương thức trên, biên dịch file **Main.java** và chạy lệnh *java Main 4* để ghi ra kết quả file *Req4_1.txt* và *Req4_2.txt* vào thư mục *output* tương ứng với **2 testcase** cho sẵn trong phương thức **main**. Sinh viên xem kết quả file này và file *Req4_1.txt* và *Req4_2.txt* trong folder *expected_output* mẫu được cung cấp sẵn để so sánh kết quả. Thứ tự kết quả của sinh viên có thể khác thứ tự trong file output mẫu nhưng phải đầy đủ các kết quả.

5. YÊU CẦU 5 (2 điểm)

Yêu cầu này là độc lập, không liên quan đến các yêu cầu trên.

Hiện thực phương thức

public double performReservation(String reservationPath, Accommodation acc, Room room, Date checkin, Date checkout) throws Exception

trả về số tiền thanh toán dịch vụ nếu đặt phòng thành công. Ngược lại, nếu đặt không thành công thì ném ra ngoại lệ theo mô tả bên dưới.

Để hiện thực được yêu cầu này, sinh viên đọc file **reservation_5.csv** được đọc từ đường dẫn *reservationPath* của tham số để lấy thông tin các phòng đã được đặt trước đó. Sinh viên sẽ kiểm tra phòng từ tham số *room* truyền, nếu phòng *room* trống thì cho phép đặt, ngược lại thời gian *checkin* hoặc *checkout* mong muốn trùng với thông tin đặt trước đang có thì đặt không thành công.

Cách tính số tiền thanh toán dịch vụ:

Tổng tiền = giá dịch vụ 1 đêm * số ngày đặt

Số tiền thanh toán dịch vụ = tổng tiền + 8% thuế trên tổng tiền

Với tham số *acc* và *room* được truyền vào sẽ luôn đảm bảo hợp lệ và tồn tại trong danh sách **Accommodation** ban đầu.




























Nếu đặt thành công thì trả về số tiền thanh toán dịch vụ và đồng thời ghi thêm thông tin đặt phòng mới vào file *reservation_5.csv* trong thư mục **data** (lưu ý sinh viên không được phép thay đổi đường dẫn khi ghi file mà phải dùng đường dẫn từ tham số *reservationPath* truyền vào để ghi thêm (append) vào file *reservation_5.csv*).

Nếu phòng đã được đặt trước thì đặt phòng không thành công và trả về một **Exception** với thông điệp (message) là “The room has already been booked during this time period”.







Sinh viên hiện thực phương thức trên, biên dịch file **Main.java** và chạy lệnh *java Main 5* để ghi ra kết quả file *Req5.txt* vào thư mục *output*. Sinh viên xem kết quả file này và file *Req5.txt* trong folder *expected_output* mẫu được cung cấp sẵn để so sánh kết quả. Đồng thời kiểm tra xem nếu ghi thành công thì file *reservation_5.csv* kiểm tra kết quả với file *reservation_5.csv* mẫu trong folder *expected_output*. Sinh viên có thể sao chép file *reservation_5.csv* ban đầu ra vị trí khác trước khi chạy Yêu cầu 5 để đảm bảo có file *reservation_5.csv* gốc dùng thực thi lại nhiều lần để kiểm thử trong quá trình code.

VII. Lưu ý kiểm tra trước khi nộp bài

- Nếu sinh viên không thực hiện được yêu cầu nào thì để nguyên phương thức của yêu cầu đó, TUYỆT ĐỐI KHÔNG XÓA PHƯƠNG THỨC CỦA YÊU CẦU sẽ dẫn đến lỗi khi chạy phương thức **main**. Trước khi nộp phải kiểm tra chạy được với phương thức **main** được cho sẵn.
- Tất cả các file ReqX.txt ($X = \{1,2,3,4,5\}$) được ghi ra trong thư mục *output*. Đối với sinh viên sử dụng IDE (Eclipse, Netbean, ...) phải đảm bảo file chạy được bằng command prompt, đảm bảo bài làm không nằm trong package. **Trường hợp bài làm đặt trong package thì sẽ bị 0 điểm cả bài.**
- File kết quả ghi đúng thư mục khi chạy chương trình sẽ có dạng như sau:

	data	06/05/2024 10:55	File folder	
	output	06/05/2024 10:55	File folder	
	Accommodation.class	06/05/2024 14:11	CLASS File	1 KB
	Accommodation.java	06/05/2024 10:55	JAVA File	1 KB
	CommonAccommodation.class	06/05/2024 14:11	CLASS File	2 KB
	CommonAccommodation.java	06/05/2024 10:55	JAVA File	1 KB
	CruiseShip.class	06/05/2024 14:11	CLASS File	2 KB
	CruiseShip.java	06/05/2024 10:55	JAVA File	1 KB
	Event.java	06/05/2024 10:55	JAVA File	2 KB
	Homestay.class	06/05/2024 14:11	CLASS File	2 KB
	Homestay.java	06/05/2024 10:55	JAVA File	1 KB
	Hotel.class	06/05/2024 14:11	CLASS File	2 KB
	Hotel.java	06/05/2024 10:55	JAVA File	1 KB
	LuxuryAccommodation.class	06/05/2024 14:11	CLASS File	2 KB
	LuxuryAccommodation.java	06/05/2024 10:55	JAVA File	3 KB
	Main.class	06/05/2024 14:11	CLASS File	4 KB
	Main.java	06/05/2024 10:55	JAVA File	4 KB
	Reservation.class	06/05/2024 14:11	CLASS File	2 KB
	Reservation.java	06/05/2024 10:55	JAVA File	2 KB
	ReservationSystem.class	06/05/2024 14:11	CLASS File	12 KB
	ReservationSystem.java	06/05/2024 10:55	JAVA File	18 KB
	Resort.class	06/05/2024 14:11	CLASS File	2 KB
	Resort.java	06/05/2024 10:55	JAVA File	2 KB
	Room.class	06/05/2024 14:11	CLASS File	3 KB
	Room.java	06/05/2024 10:55	JAVA File	3 KB
	Villa.class	06/05/2024 14:11	CLASS File	1 KB
	Villa.java	06/05/2024 10:55	JAVA File	1 KB

- Bên trong thư mục *output*:

	Req1.txt	06/05/2024 10:55	Text Document	2 KB
	Req2.txt	06/05/2024 10:55	Text Document	1 KB
	Req3.txt	06/05/2024 10:55	Text Document	1 KB
	Req4_1.txt	06/05/2024 14:13	Text Document	1 KB
	Req4_2.txt	06/05/2024 14:13	Text Document	1 KB
	Req5.txt	06/05/2024 10:55	Text Document	1 KB

VIII. Hướng dẫn nộp bài

- Khi nộp bài sinh viên nộp lại các file *Accommodation.java*, *Room.java*, *CommonAccommodation.java*, *LuxuryAccommodation.java*, *Homestay.java*, *Hotel.java*,

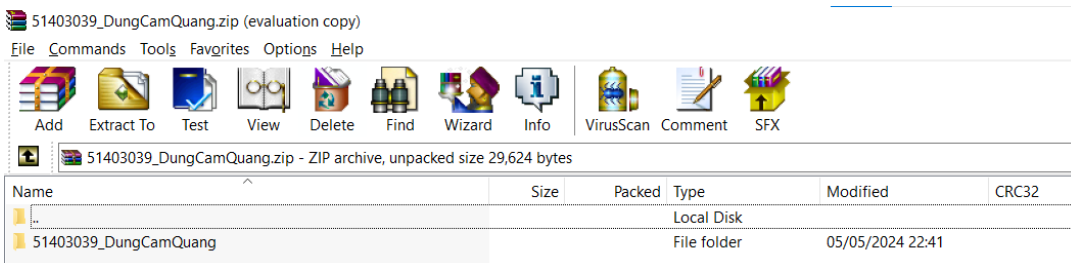
Resort.java, Villa.java, CruiseShip.java và file *RevervationSystem.java*, **không nộp kèm bất cứ file nào khác và tuyệt đối không được sửa tên 10 file này.**

- **Sinh viên đặt 10 file bài làm vào thư mục MSSV_HoTen** (HoTen viết liền, không dấu) và nén lại với định dạng **.zip** và nộp vào mục nộp tương ứng trên hệ thống ELIT (elit.tdtu.edu.vn).
- Trường hợp làm sai yêu cầu nộp bài (đặt tên thư mục sai, không để bài làm vào thư mục khi nộp, nộp dư file, nộp thiếu file, ...) thì bài làm của sinh viên sẽ bị **0 điểm**.
- File nộp đúng sẽ như sau:

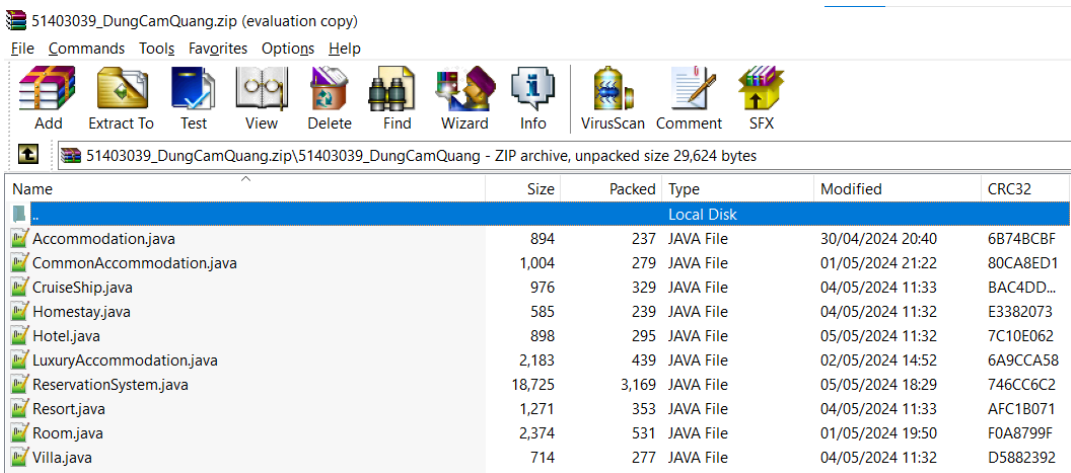
- o File nén nộp bài:

 51403039_DungCamQuang.zip 06/05/2024 14:16 WinRAR ZIP archive 9 KB

- o Bên trong file nén:



- o Bên trong thư mục:



IX. Đánh giá và quy định

- Bài làm sẽ được chấm tự động thông qua testcase (file input và output có định dạng như mẫu đã gửi kèm) do đó sinh viên tự chịu trách nhiệm nếu không thực hiện đúng theo Hướng dẫn nộp bài hoặc tự ý sửa tên các phương thức đã có sẵn dẫn đến bài làm không biên dịch được khi chấm.
- **Tất cả các trường hợp sinh viên gán cứng đường dẫn trong quá trình đọc file đều sẽ bị 0 điểm cả bài.**
- Testcase sử dụng để chấm bài là file khác với file sinh viên đã nhận, sinh viên chỉ được điểm mỗi YÊU CẦU khi chạy ra đúng hoàn toàn kết quả theo testcase chấm của yêu cầu đó.

- Nếu bài làm của sinh viên biên dịch bị lỗi trên máy chấm thì **0 điểm cả bài**.
- **Sinh viên tuyệt đối KHÔNG ĐƯỢC dùng các kỹ thuật trong Java nâng cao hơn so với kiến thức hiểu biết và khả năng lập trình của sinh viên.**
- **Tất cả code sẽ được kiểm tra sao chép. Mọi hành vi sao chép code trên mạng, chép bài bạn hoặc cho bạn chép bài nếu bị phát hiện đều sẽ bị điểm 0 vào điểm Quá trình 2 hoặc cấm thi cuối kì.**
- **Đặc biệt, tất cả các trường hợp code sinh từ chatbot (ChatGPT, Gemini, ...) đều đã được lưu trữ vào kho dữ liệu sao chép, nếu sinh viên sử dụng chatbot trong bài tập lớn này thì sinh viên sẽ bị xử lý tương tự hành vi sao chép bài.**
- **Các trường hợp phát hiện không phải tự code, nhờ người khác làm hộ, mua bán bài tập lớn, hoặc không giải thích được nguồn gốc code sẽ đưa về Khoa để xử lý kỷ luật theo Quy chế Công tác sinh viên hiện hành.**
- Nếu bài làm của sinh viên có dấu hiệu sao chép trên mạng hoặc sao chép nhau, sinh viên sẽ được gọi lên phòng vấn code để chứng minh bài làm là của mình.
- **Hạn chót nộp bài: 23h59 ngày 26/05/2024.**
- **Sinh viên nộp đúng vào mục bài tập của Bài tập lớn trên hệ thống ELIT.**

-- HẾT --