

## Assignment

MÔN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

MÃ MÔN: 503005

### Đề bài: Module trong phần mềm quản lý sinh viên

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

#### I. Giới thiệu bài toán

Một trường đại học muốn xây dựng hệ thống quản lý sinh viên. Trong bài tập này, sinh viên sẽ phải hiện thực một số module (thành phần) chức năng trong hệ thống quản lý sinh viên.

Các chức năng mà sinh viên phải thực hiện là:

- Đọc file danh sách sinh viên.
- Ghi file danh sách sinh viên.
- Thêm sinh viên.
- Xóa sinh viên.
- Tìm danh sách sinh viên.
- Sắp xếp danh sách sinh viên.

#### II. Tài nguyên cung cấp

Source code được cung cấp sẵn bao gồm các file:

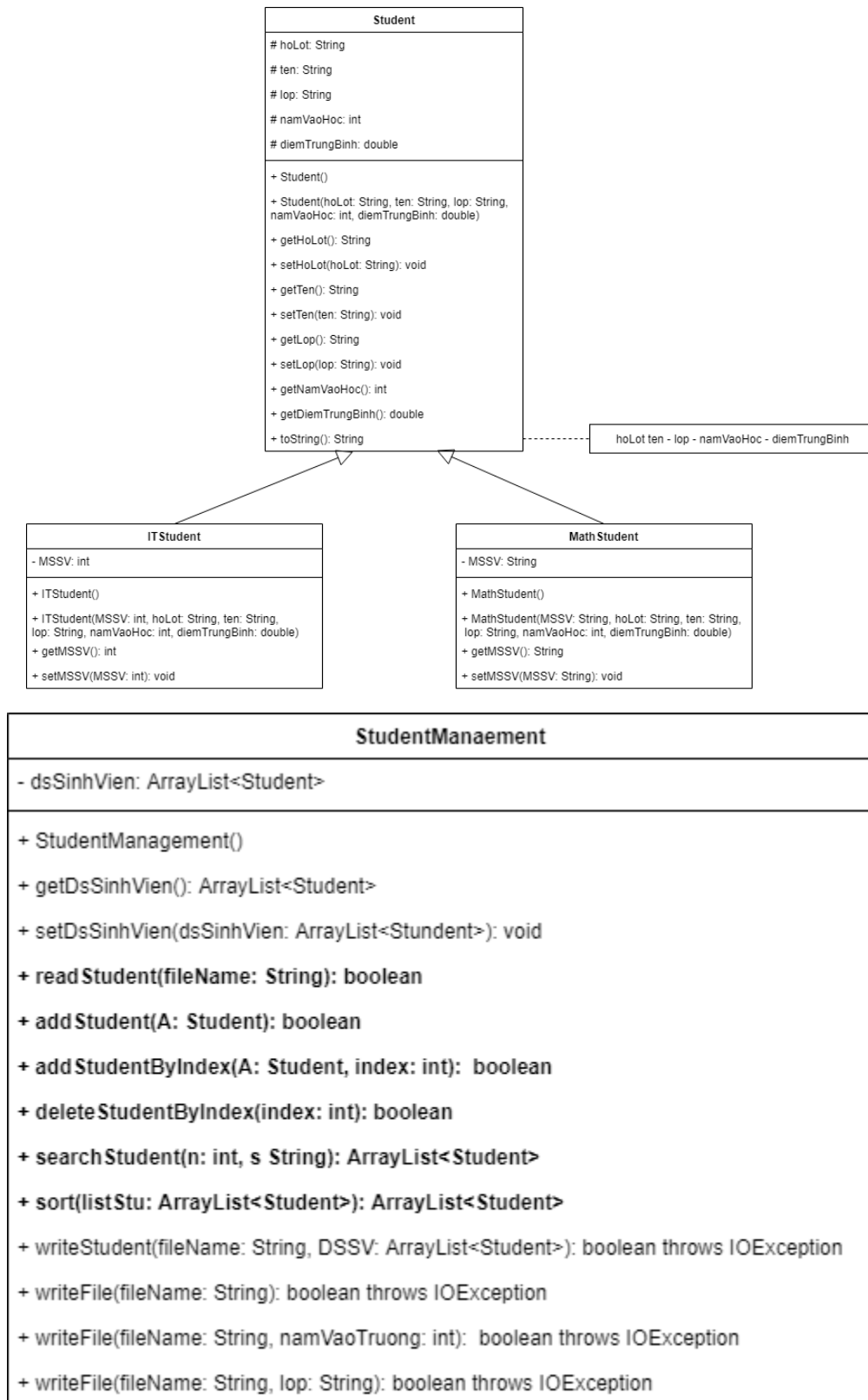
- File đầu vào và kết quả mong muốn:
  - o *DSSV.txt*: chứa danh sách sinh viên.
  - o Folder *output* gồm: 4 file *Req1.txt*, *Req2.txt*, *Req3.txt*, *Req4.txt* chứa kết quả mẫu của 4 yêu cầu trong bài.
- File sinh viên **không được chỉnh sửa**:
  - o *Main.java*: tạo đối tượng và gọi các hàm sinh viên sẽ tự định nghĩa.
  - o *Student.java*: chứa lớp **Student** được định nghĩa sẵn.

- *StudentManagement.java*: chứa lớp **StudentManagement** được định nghĩa sẵn một số thuộc tính, phương thức khởi tạo, phương thức ghi file và một số phương thức trống. Sinh viên sẽ hiện thực thêm vào các phương thức còn trống trong file này.

### III. Trình tự thực hiện bài

- Sinh viên tải file tài nguyên được cung cấp sẵn và giải nén.
- Trong cùng thư mục với 3 file cung cấp sẵn sinh viên tạo ra 2 file tương ứng với 2 lớp con là **ITStudent**, **MathStudent**.
- Sinh viên tự hiện thực lớp **ITStudent**, **MathStudent** và code thêm vào lớp **StudentManagement** theo mô tả và yêu cầu trong các phần tiếp theo.
- Khi hiện thực xong lớp **ITStudent** và **MathStudent**, nếu hiện thực đúng thì sinh viên chạy hàm **main** sẽ ra 4 file txt rỗng.
- Sau khi hiện thực các phương thức trong lớp **StudentManagement**, sinh viên biên dịch và chạy với hàm **main** trong file *Main.java* đã gọi sẵn các phương thức, kết quả sau khi chạy hàm **main** là 4 file txt. Sinh viên thực hiện các yêu cầu ở mục dưới và so sánh kết quả output của mình với kết quả output đã được cung cấp sẵn.
- **Đối với các yêu cầu sinh viên không làm được vui lòng không xóa và phải đảm bảo chương trình chạy được trước khi nộp bài.**

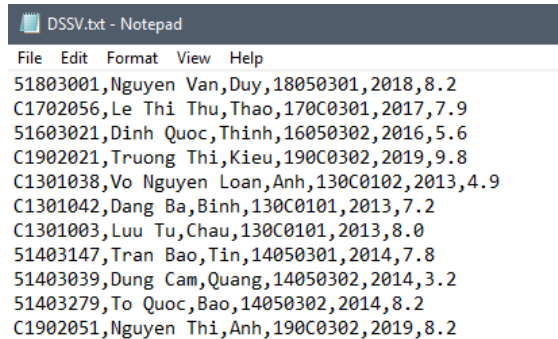
### IV. Mô tả lớp Student, ITStudent, MathStudent và StudentManagement



- Lớp **ITStudent** và lớp **MathStudent** kế thừa là lớp **Student**, thuộc tính *MSSV* là mã số sinh viên, *MSSV* của **ITStudent** là *int* và của **MathStudent** là *String*.
- Giải thích một số thuộc tính và phương thức như sau:
  - Lớp **Student**
    - *hoLot*: họ và tên lót sinh viên
    - *ten*: tên sinh viên
    - *namVaoHoc*: năm sinh viên nhập học
  - Lớp **StudentManagement**
    - *dsSinhVien*: danh sách sinh viên
    - **readStudent(String fileName)**: đọc từ file vào danh sách *dsSinhVien*
    - **addStudent(Student A)**: thêm sinh viên A vào cuối danh sách *dsSinhVien*
    - **addStudentByIndex(Student A, int index)**: thêm sinh viên A vào vị trí index của *dsSinhVien* (với index tính từ 0 là phần tử đầu tiên)
    - **deleteStudentByIndex(int index)**: xóa sinh viên khỏi danh sách *dsSinhVien* (với index tính từ 0 là phần tử đầu tiên)
    - **searchStudent(int n, String s)**: trả về danh sách sinh viên theo lớp hoặc theo năm vào học (sẽ được mô tả trong phần yêu cầu).
    - **sort(ArrayList<Student> listStu)**: trả về danh sách sinh viên *listStu* đã được sắp xếp.
    - **writeStudent(String fileName, ArrayList<Student> DSSV)**: ghi file danh sách sinh viên theo định dạng kết quả (phương thức này sinh viên không được sửa).
    - Các hàm **writeFile()** được nạp chồng (overloading) để gọi đến phương thức **writeStudent()** ghi file theo yêu cầu. (phương thức này sinh viên không được sửa)

## V. Mô tả file input và file output

- File input có tên *DSSV.txt* chứa danh sách sinh viên với mỗi dòng ứng với thuộc tính của 01 sinh viên được cách nhau bằng dấu “,” theo định dạng:  
MSSV,Họ lót,Tên,Lớp,Năm vào học,Điểm trung bình



```
DSSV.txt - Notepad
File Edit Format View Help
51803001,Nguyen Van,Duy,18050301,2018,8.2
C1702056,Le Thi Thu,Thao,170C0301,2017,7.9
51603021,Dinh Quoc,Thinh,16050302,2016,5.6
C1902021,Truong Thi,Kieu,190C0302,2019,9.8
C1301038,Vo Nguyen Loan,Anh,130C0102,2013,4.9
C1301042,Dang Ba,Binh,130C0101,2013,7.2
C1301003,Luu Tu,Chau,130C0101,2013,8.0
51403147,Tran Bao,Tin,14050301,2014,7.8
51403039,Dung Cam,Quang,14050302,2014,3.2
51403279,To Quoc,Bao,14050302,2014,8.2
C1902051,Nguyen Thi,Anh,190C0302,2019,8.2
```

- File output gồm có 4 file ứng với kết quả mẫu cho 4 yêu cầu:
  - o *Req1.txt*: kết quả danh sách sinh viên sau khi thực hiện các yêu cầu thêm, xóa sinh viên trong hàm main.
  - o *Req2.txt*: kết quả tìm kiếm theo năm vào học là 2019.
  - o *Req3.txt*: kết quả tìm kiếm theo lớp là “14050302”.
  - o *Req4.txt*: kết quả sau khi được sắp xếp.
- Mỗi dòng trong từng file output ứng với 01 sinh viên với các thông tin được ghi ra file theo định dạng trong hàm **toString()** của lớp **Student**:

**MSSV - hoLot ten - lop - namVaoHoc - diemTrungBinh**



```
Req3.txt - Notepad
File Edit Format View Help
51903001 - Nguyen Binh An - 19050301 - 2019 - 4.0
C1902021 - Truong Thi Kieu - 190C0302 - 2019 - 9.8
C1902051 - Nguyen Thi Anh - 190C0302 - 2019 - 8.2
```

## VI. Yêu cầu

### 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 hàm **main** để xác định hướng hiện thực các class các phương thức theo thức tự.

- Sinh viên có thể thêm một số thao tác vào hàm **main** để kiểm các phương thức tuy nhiên đảm bảo bài làm của sinh viên **phải chạy được trên hàm main 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 file Main.java đã được cung cấp sẵn.
- **Tuyệt đối không chỉnh sửa tên của các phương thức đã có sẵn (tuân theo đúng sơ đồ lớp phía trên).**

### YÊU CẦU 1 (4 điểm)

- Sinh viên hiện thực phương thức **public boolean readStudent(String fileName)**
  - Trả về false khi không thể mở file, trả về true khi đọc file thành công vào danh sách *dsSinhVien*. (gợi ý: đọc String từ file, cắt chuỗi và đưa vào khởi tạo đối tượng)
- Sinh viên hiện thực phương thức **public boolean addStudent(Student A)**
  - Phương thức trước khi thêm sẽ kiểm tra *MSSV* của **Student A** thêm vào có bị trùng với đối tượng nào trong *dsSinhVien* hay không. Nếu trùng thì trả về *false* và không thêm vào danh sách, nếu không trùng thì thêm vào danh sách *dsSinhVien* và trả về *true*. (gợi ý: dùng hàm add của ArrayList)
- Sinh viên hiện thực phương thức **public boolean addStudentByIndex(Student A, int index)**
  - Phương thức trước khi thêm sẽ kiểm tra *MSSV* của **Student A** thêm vào có bị trùng với đối tượng nào trong *dsSinhVien* hay không. Nếu trùng thì trả về *false* và không thêm vào danh sách, nếu không trùng thì thêm vào danh sách *dsSinhVien* và trả về *true*. (gợi ý: dùng hàm add của ArrayList)
- Sinh viên hiện thực phương thức **public boolean deleteStudentByIndex(int index)**
  - Xóa sinh viên theo index truyền vào, nếu index lớn hơn số phần tử của *dsSinhVien* thì trả về *false*, nếu index hợp lệ thì xóa phần tử và trả về *true* (gợi ý: dùng hàm remove của ArrayList)

Sinh viên hiện thực các phương thức sao cho sau đọc file *DSSV.txt* và thực hiện các lệnh gọi phương thức trong hàm **main** sẽ cho kết quả ghi ra file “Req1.txt” như file output mẫu.

### YÊU CẦU 2 và YÊU CẦU 3 (5 điểm)

Hiện thực phương thức **public ArrayList<Student> searchStudent(int n,String str)**

Trong phương thức trên đã có sẵn cấu trúc switch ... case ..., sinh viên định nghĩa nếu:

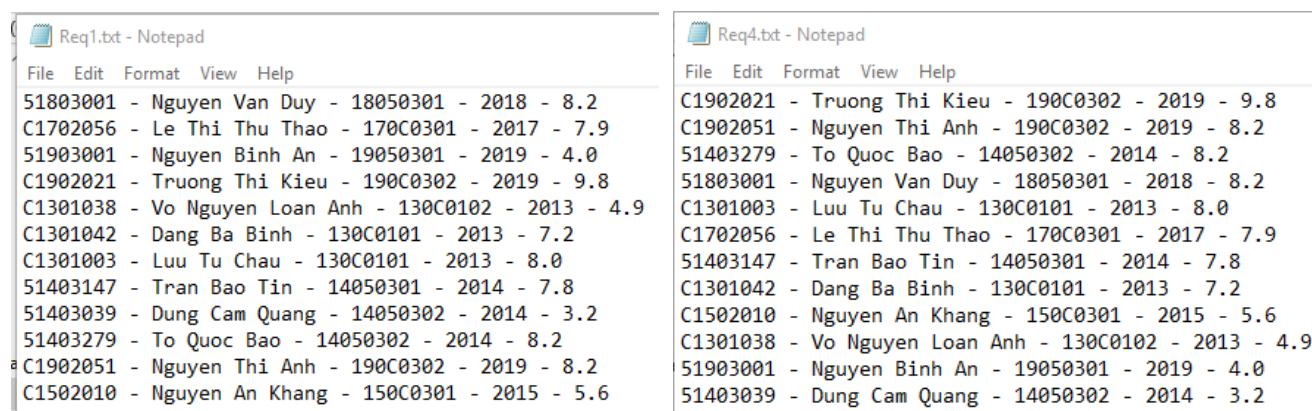
- **n = 1 (YÊU CẦU 2):** trả về danh sách các sinh viên có năm vào trường **str** từ *dsSinhVien* (gợi ý: dùng hàm parseInt để chuyển về int so sánh)
- **n = 2 (YÊU CẦU 3):** trả về danh sách các sinh viên có lớp **str** từ *dsSinhVien*.

Sinh viên hiện thực các phương thức trên sao cho khi thực hiện các lệnh gọi phương thức trong hàm **main** sẽ cho kết quả ghi ra file “Req2.txt” và “Req3.txt” như file output mẫu. (Yêu cầu 2 và yêu cầu 3 sẽ chạy với danh sách sinh viên sau khi thực hiện yêu cầu 1).

#### YÊU CẦU 4 (1 điểm)

Hiện thực phương thức **public ArrayList<Student> sort(ArrayList<Student> listStu)**

Phương thức trên trả về danh sách sinh viên đã được sắp xếp theo thứ tự điểm trung bình từ lớn xuống bé. Nếu điểm trung bình bằng nhau thì sắp xếp theo chữ cái đầu tiên của tên sinh viên.



```
Req1.txt - Notepad
File Edit Format View Help
51803001 - Nguyen Van Duy - 18050301 - 2018 - 8.2
C1702056 - Le Thi Thu Thao - 170C0301 - 2017 - 7.9
51903001 - Nguyen Binh An - 19050301 - 2019 - 4.0
C1902021 - Truong Thi Kieu - 190C0302 - 2019 - 9.8
C1301038 - Vo Nguyen Loan Anh - 130C0102 - 2013 - 4.9
C1301042 - Dang Ba Binh - 130C0101 - 2013 - 7.2
C1301003 - Luu Tu Chau - 130C0101 - 2013 - 8.0
51403147 - Tran Bao Tin - 14050301 - 2014 - 7.8
51403039 - Dung Cam Quang - 14050302 - 2014 - 3.2
51403279 - To Quoc Bao - 14050302 - 2014 - 8.2
C1902051 - Nguyen Thi Anh - 190C0302 - 2019 - 8.2
C1502010 - Nguyen An Khang - 150C0301 - 2015 - 5.6

Req4.txt - Notepad
File Edit Format View Help
C1902021 - Truong Thi Kieu - 190C0302 - 2019 - 9.8
C1902051 - Nguyen Thi Anh - 190C0302 - 2019 - 8.2
51403279 - To Quoc Bao - 14050302 - 2014 - 8.2
51803001 - Nguyen Van Duy - 18050301 - 2018 - 8.2
C1301003 - Luu Tu Chau - 130C0101 - 2013 - 8.0
C1702056 - Le Thi Thu Thao - 170C0301 - 2017 - 7.9
51403147 - Tran Bao Tin - 14050301 - 2014 - 7.8
C1301042 - Dang Ba Binh - 130C0101 - 2013 - 7.2
C1502010 - Nguyen An Khang - 150C0301 - 2015 - 5.6
C1301038 - Vo Nguyen Loan Anh - 130C0102 - 2013 - 4.9
51903001 - Nguyen Binh An - 19050301 - 2019 - 4.0
51403039 - Dung Cam Quang - 14050302 - 2014 - 3.2
```

Sinh viên hiện thực các phương thức trên sao cho khi thực hiện các lệnh gọi phương thức trong hàm **main** sẽ cho kết quả ghi ra file “Req4.txt” như file output mẫu. (Yêu cầu 4 sẽ chạy với danh sách sinh viên sau khi thực hiện yêu cầu 1).

Lưu ý 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 hàm **main**. Trước khi nộp phải kiểm tra chạy được với hàm **main** được cho sẵn.

#### VII. Hướng dẫn nộp bài

- Khi nộp bài sinh viên nộp lại file *ITStudent.java*, *MathStudent.java* và file *StudentManagement.java*, **không nộp kèm bất cứ file nào khác và tuyệt đối không được sửa tên 3 file này.**
- **Sinh viên đặt 3 file bài làm vào thư mục MSSV\_HoTen** và nén lại với định dạng **.zip** nộp lên ELIT vào mục Assignment do giảng viên thực hành mở.
- 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, ...) thì bài làm của sinh viên sẽ bị **0 điểm**.

## VI. 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.
- Testcase sử dụng để chấm bài là 1 file *DSSV.txt* khác với file sinh viên đã nhận, sinh viên chỉ được điểm mỗi YÊU CẦU khi chạy ra đúng hoàn toàn kết quả của yêu cầu đó.
- Nếu bài làm của sinh viên biên dịch bị lỗi thì **0 điểm**.
- **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.**
- 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: 23h00 ngày 03/05/2020.**

-- HẾT --