**Họ tên SV: Đỗ Minh Vũ**

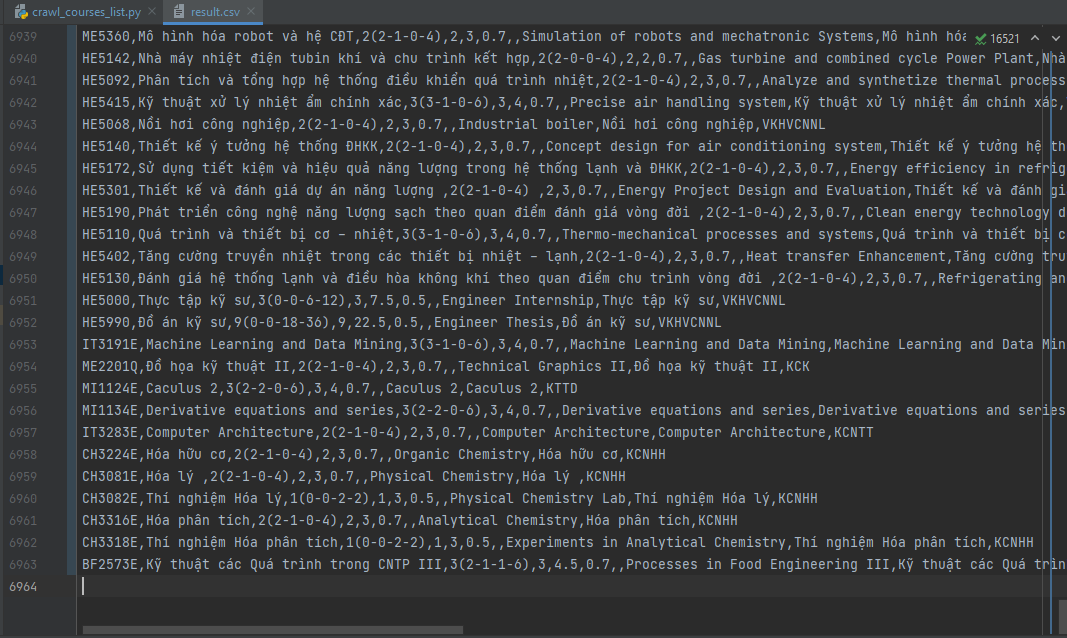
**MSSV: 20173471**

**Báo cáo hằng tuần**

**(Tuần thứ 2: 25/10/2021 – 31/10/2021)**

1. Kết quả làm việc

* Crawl tất cả dữ liệu (6962 môn học) từ 349 page, cùng với phần thông tin bổ sung của tất cả môn học.
* Lưu kết quả dưới dạng file csv như hình:



1. Các bước thực hiện

* Cài đặt thư viện Selenium để mô phỏng thao tác chuyển trang.

***pip install selenium***

* Chỉnh sửa code để có thể lấy toàn bộ dữ liệu trong nhiều trang. Source Code như trong link github:

***https://github.com/dominhvu2605/Project3***

* Xử lý: với mỗi trang, tính số lượng môn học trong danh sách hiển thị. Sử dụng vòng lặp for để lấy dữ liệu từng môn:

+ Mỗi môn sẽ thực hiện giả lập ấn nút mở rộng để xem thông tin môn học, lưu các dữ liệu chi tiết đó vào biến.

+ Cào dữ liệu trong bảng đối với mỗi môn học.

+ Ghi dữ liệu thu thập được của từng môn học vào file csv.

* Để chạy code, gõ dòng lệnh sau tại terminal của IDE (Ghi đè kết quả lên file nếu file đã tồn tại)

***scrapy crawl course -O result.csv***

* Nếu chưa cài PATH cho scrapy thì sử dụng lệnh sau:

***python -m scrapy crawl course -O result.csv***

* Kết quả thu được như mục 1.

1. Các vấn đề và hướng giải quyết

* Các vấn đề
* Tuy đã crawl được tất cả dữ liệu nhưng thời gian chạy tốn nhiều thời gian. Nguyên nhân có thể do driver xử lý các response chậm, do đó cần sử dụng hàm sleep() để chờ nhận được tất cả source code khi thực hiện ấn nút.
* Vẽ ma trận quan hệ của các học phần.
* Hướng giải quyết
* Tìm hiểu thêm các hàm để xử lý ngoại lệ, bắt hết các thành phần khi thực hiện ấn nút mà không phải cài đặt thời gian cố định cho hàm sleep().
* Tìm hiểu về thư viện Graphviz để vẽ ma trận quan hệ của các môn học.

**Báo cáo hằng tuần**

**(Tuần thứ 3: 01/11/2021 – 07/11/2021)**

1. Kết quả làm việc

* Thêm comment vào trong code
* Sử dụng pydoc sinh ra file html
* Kết quả:

Graphical user interface, application, Teams

Description automatically generated

* Tìm ra hướng giải quyết cho việc thay thế việc sử dụng *time.sleep* để chờ 1 thời gian cố định: sử dụng selenium wait

+ Implicit wait: nó sẽ yêu cầu web driver tạm dừng một khoảng thời gian nhất định đã thiết lập ban đầu. Implicit wait tương tự như sleep nên không cải thiện được tốc độ.

+ Explicit Wait: khi sử dụng explicit wait nó sẽ đi kèm với một điều kiện nào đó, tức là thay vì chờ đợi một khoảng thời gian được thiết lập sẵn thì nó sẽ chờ một điều kiện cụ thể nào đó hay kiểm tra khi việc wait đã vượt qua khoảng thời gian maximum nào đó, trước khi output ra một exception.

+ Fluent wait: Sử dụng trong tường hợp thời gian load một phần tử có thể khác nhau. Fluent wait sẽ tìm kiếm đi tìm kiếm lại cho đến khi tìm được phần tử đó hoặc đến khi time out thì thôi. Thay vì set *time.sleep* cố định là 1s thì em có thể set timeout cho Fluent wait là 1s hoặc 2s và như vậy, khi đã tìm được phần tử thì sẽ thực hiện ấn nút mà không cần chờ thời gian cố định như ban đầu.

* Tuy nhiên, do không có thời gian để crawl dữ liệu lại từ đầu nên em chỉ tìm ra được phương pháp giải quyết vấn đề mà chưa triển khai vào code.
* Sử dụng graphviz để vẽ mỗi quan hệ của các học phần có học phần điều kiện

Diagram, schematic

Description automatically generated

1. Các bước thực hiện

* Về Detail Design:
* Thực hiện comment code, sủ dụng form ’’’ <comment> ’’’.
* Truy cập vào folder chứa file .py mới thêm comment ở trên.
* Thực hiện câu lệnh: ***pydoc -w crawl\_courses\_list*** để render ra file html.
* Nếu chưa cài PATH môi trường, sử dụng câu lệnh: ***python -m pydoc -w crawl\_courses\_list.py*** để gender ra file html.
* Mở file html bằng trình duyệt để có kết quả.
* Về graphviz:
* Thực hiện mở file csv chứa data crawl được.
* Đọc file csv, với mỗi dòng, kiểm tra xem có tồn tại dữ liệu tại trường “Học phần điều kiện” hay không. Nếu không có sẽ bỏ qua dòng đó. Nếu tồn tại thì ta sẽ lấy thông tin 2 trường “Mã HP” và “Học phần điều kiện” của dòng đó và tiến hành tạo node và edge.
* Cuối cùng xuất ra file .gv và sẽ ma trận quan hệ. Kết quả lưu vào file pdf

1. Các vấn đề và hướng giải quyết

* Các vấn đề:
* Em xem kết quả file html render ra có vẻ không được ổn nhưng khong biết chỉnh thế nào.

Trả lời: Trải nghiệm như vậy là ổn rồi. Em comment cũng đúng chuẩn.

Em có thể bổ sung thêm các chú thích nữa ở đầu hàm thì sẽ hiệu quả hơn. Ví dụ với hàm **def start\_requests(self)** thì thầy thêm

        Remark:

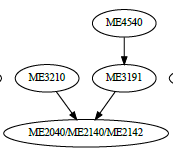
            this a callback funtion, alway run first when spider is running.

Các dạng đầu mục mà người ta hay comment cho hàm là

See, Remark, Example

* Khi vẽ ma trận quan hệ giữa các học phần thì em chỉ vẽ được cơ bản nhất là trên cùng 1 dòng nên bố cục chưa được đẹp. Có những môn học có nhiều học phần điều kiện hoặc chỉ thỏa mãn 1 trong các học phần thì em chưa bóc tách dữ liệu để vẽ một cách chuẩn xác được mà còn để nguyên các học phần đó thành 1 node để vẽ.

Trả lời: đây cấu hình để gọi graphviz, có thời gian tìm hiểu thêm là okay thôi. Mà thầy chưa hiểu lắm ý em . Có phải ý em như trong hình là ME 3210 có thể là môn học điều kiện của 2 môn thì chưa vẽ được, và boc tách cụm ME2040, ME2140,ME2142 thành các môn riêng biệt đung không



* Do tuần này em hơi bận nên vẫn chưa làm phần crawl duy nhất 1 học phần trong tham số dòng lệnh truyền vào.
* Hướng giải quyết:
* Em sẽ tìm hiểu thêm cách vẽ trên nhiều dòng của graphviz để bố cục được đẹp hơn.
* Em sẽ giành thời gian hơn để làm tiếp phần chưa hoàn thành được trong tuần này.

**Báo cáo hằng tuần**

**(Tuần thứ 4+5: 08/11/2021 – 21/11/2021)**

1. Kết quả làm việc

* Chỉnh sửa các đường dẫn tĩnh trong code thành đường dẫn tương đối
* Thêm file requirements để liệt kê các thư viện cần thiết cho project
* Sử dụng dòng lệnh để crawl dữ liệu của 1 môn học cần tìm.

1. Các bước thực hiện

* Thêm các file cần thiết vào folder của dự án, sửa các đường dẫn tới các file này thành đường dẫn tương đối
* Sử dụng câu lệnh *pip freeze > requirements.txt* để render các thư viện cần thiết cho dự án vào file requirements. Sau đó sẽ tìm các thư viện tự thêm vào project và xóa các thư viện còn lại. Như vậy đã có thể xuất ra được các thư viện thêm từ nguồn bên ngoài vào project.
* Trong phần sử dụng tham số dòng lệnh để crawl một môn theo mã môn học, em sẽ thêm một bước tiền xử lý để xem người dùng có nhập mã môn học hay không. Nếu không nhập mã môn học, chương trình sẽ crawl hết thông tin của tất cả các học phần. Nếu người dùng nhập mã học phần trong commandline thì sẽ tiến hành tìm kiếm ô search theo mã học phần, search theo mã học phần đó và crawl các thông tin có được theo như thuật toán cũ.

1. Các vấn đề và hướng giải quyết

* Các vấn đề:
* Trong phần tham số dòng lệnh, chương trình sẽ bị lỗi nếu người dùng không nhập mã môn học cần tìm (tức là crawl toàn bộ thông tin của tất cả các môn học). Do đó, em đang sử dụng try … except … để bỏ qua lỗi này và crawl toàn bộ khi không có mã môn học được nhập vào. Em không biết có cách nào để check xem trong tham số dòng lệnh có tồn tại tham số “code” để sử dụng hay không, vì nếu không có tham số này thì câu lệnh *if (self.code != "")* sẽ lỗi. Em không biết cách dùng try như trên có ổn không, mong thầy cho em lời khuyên ạ.
* Trả lời:

1. Chương trình này sẽ chạy ở server. Các tham số vào sẽ được kiểm tra bởi lớp vỏ web api bên ngoài. Vì thế lỗi này sẽ không gặp nữa.
2. Thầy thấy em đang sử dụng luôn bộ phân tích tham số mặc định của scrapy vì em gọi luôn biến self.code. Thầy thì phân tích luôn từ tham số sys.argv nên sẽ chủ động hơn (nhưng thầy không sử dụng cách viết spider qui củ như em đang làm).
3. Em sử dụng hàm hasattr như trong minh họa này: [How to know if an object has an attribute in Python - Stack Overflow](https://stackoverflow.com/questions/610883/how-to-know-if-an-object-has-an-attribute-in-python)

* Trong phần vẽ đồ thị sử dụng tham số để vẽ cho 1 môn, em mong muốn có thể tìm hết tất cả các học phần điều kiện (HPDK) của môn học sâu nhất có thể. Ví dụ: môn A có HPDK là B, môn B có HPDK là C, môn C có HPDK là D thì trong sơ đồ sẽ vẽ là A ➩ B ➩ C ➩ D. Nhưng hiện tại em vẫn chưa tìm được giải pháp ạ.

Trả lời: chính xác

Giải pháp toàn diện sẽ thế này:

* + **Phần mềm A.1:** scrapy và thu thập toàn bộ các mã học phần ra file csv. Là chương trình với tham số -O mà em đã viết.
  + **Phần mềm A.2:** Viết một phần mềm khác độc lập, cũng bằng python, để đọc file csv đó, đồng thời phân tích sâu hơn về môn phụ thuộc và bị phụ thuộc
    - Sau đó, em hãy đưa cái kết quả phân tích và toàn bộ file csv vào một cơ sở dữ liệu. Có lẽ trường hợp này sử dụng một NoSQL nào đó như MongoDB là tiện nhất.
  + **Phần mềm A.3**: chỉ là môt đoạn khai báo để chạy định kỳ bằng cronjob (nếu dùng Linux) hoặc TaskSchedule (nếu dùng Windows) để định kỳ một lần, vào cuối tuần, sẽ chạy Phần mềm A1 và A2
  + **Phần mềm B.1:** Viết môt phần mềm khác chịu trách nhiệm đọc học phần từ tham số dòng lệnh (tương tự như tham số code= hiện thời). Sau đó truy vấn db và sinh ra file .dot (Tham khảo [DOT Language | Graphviz](http://www.graphviz.org/doc/info/lang.html))
  + **Phần mềm B.2:**  thực ra chỉ là một lệnh command line, viết vào file .ps hoặc là file .sh tùy theo OS là Windows hay Linux. Lệnh này sẽ yêu cầu graphviz đọc dot file và sinh ra ảnh ([windows - Graphviz: How to go from .dot to a graph? - Stack Overflow](https://stackoverflow.com/questions/1494492/graphviz-how-to-go-from-dot-to-a-graph))
  + **Phần mềm B.3**: là một webAPI, nên viết bằng Python Fast, vì dự án này sử dụng python nhiều. Hoặc NodeJS swagger tùy theo thạo cái gì.
  + Thầy giải thích thêm tại sao nên viết tách thành phần mềm riêng: các em thường viết tất cả tính năng vào một phần mềm 🡪 to, cồng kềnh, dễ bị mất toàn bộ mã nguồn khi giao cho ai đó, khó kiếm người thạo tất cả các mảng để kiểm soát đống code đó. Vậy hãy tách chức năng crawl, và chức năng phân tích cú pháp riêng.
  + To tát là thế, nhưng thực ra viết không lâu đâu.
* Trong vẽ đồ thị, em vẫn chưa phân tích được các nhóm môn học điều kiện.

Trả lời: Theo qui tắc trình bày của sis thầy đoán thế này.

* + - Với môn IT3030: học phần điều kiện viết thế này **IT1110/IT1140**

Đây là phép hoặc, có nghĩa là chỉ cần học môn IT1110, hoặc là môn IT1140 là được

* + - Với môn ME3130: học phần điều kiện viết thê này **((ME3191/ME3050),(ME3090/ME3220))/ME3191/ME3061** tức là chỉ câ
      * Học môn ME3061 là okay
      * Hoặc là học môn ME3191 là okay
      * Hoặc là phải học cả 2 môn **(ME3191 hoặc ME3050) và (ME3090 hoặc ME3220)**
    - Vậy túm lại, em phiên dịch dấu “/” thành phép OR, và dấu “,” thành phép AND.
* Hướng giải quyết
* Em sẽ tìm hiểu thêm để phân tích cấu trúc và xử lý file danh sách môn học để có thể xuất được sơ đồ như mong muốn.

**Báo cáo hằng tuần**

**(Tuần thứ 6: 22/11/2021 – 28/11/2021)**

1. Kết quả làm việc:

* Tiếp tục tìm hiểu cách xử lý dữ liệu môn học.
* Tìm hiểu các thuật toán để có thể vẽ đồ thị của một môn nhập từ commandline. Đồ thị sẽ vẽ ra tất cả các môn học điều kiện theo chiều sâu. Từ đó sẽ cho ta biết được tất cả các môn điều kiện cần học để có thể học một học phần nào đó

1. Các vấn đề và hướng giải quyết

* Các vấn đề:
* Việc phân tách các nhóm học phần điều kiện phải rõ ràng, dễ hiều và có thể tiếp tục tìm kiếm các học phần điều kiện cho nhóm học phần này (có thể trong nhóm này chỉ có một vài môn có học phần điều kiện, còn lại thì không? Cách phân chia như thế nào?)

Trả lời:  
Đúng rồi. Nếu học phần điều kiện bỏ trống, tức là không bị phụ thuộc 🡪 sinh viên có thể học môn này bất cứ lúc nào.

Thông tin trên trang sis là chính xác. Phòng đào tạo vẫn cập nhật ở đó. Vậy ta chỉ cần crawl ở đó thôi.

* Việc truy xuất các học phần điều kiện theo chiều sâu khá khó khăn. Nếu sử dụng mỗi vòng lặp cho một tầng môn điều kiện thì sẽ dẫn dến thời gian thực hiện sẽ rất lâu nếu độ sâu của đồ thị có thể là ba, bốn hoặc lớn hơn (mỗi vòng lặp qua 6900 môn, độ phức tập thuật toán sẽ tăng rất nhanh với n = 3,4, …).
* Trả lời:  
  Đúng rồi. Giải pháp đã ghi ở phần trả lời tuần 5. Đó là chỉ crawl 1 lần và đưa vào database. Sau đó ta chỉ cần tìm kiểm trên database

Yên tâm là không có vòng lặp vô hạn, vì nếu tồn tại thì SV sẽ không ra trường được 😊

* Hướng giải quyết:
* Hiện tại em vẫn chưa tìm ra được giải quyết cho bài toán.
* Em đang tìm hiểu để có thể cố tìm ra một cách có độ phức tạp thuật toán không quá lớn.

**Báo cáo hằng tuần**

**(Tuần thứ 7: 06/12/2021 – 12/12/2021)**

1. Kết quả làm việc:

* Đã có thể tìm các học phần điều kiện theo chiều sâu.



* Tìm hiểu cách để có thể lấy hết tất cả các môn học điều kiện theo chiều sâu.

1. Các bước thực hiện:

* Sử dụng mảng để lưu tất cả dữ liệu các môn học.
* Tìm ra dữ liệu chứa học phần nhập từ commandline.
* Tìm học phần điều kiện của môn học đó. Sau đó xem học phần điều kiện mới tìm được làm học phần cần tìm học phần điều kiện.
* Cứ tìm như vậy cho đến khi học phần điều kiện là “” (không còn tồn tại). Như vậy đã lấy hết được dữ liệu để vẽ cây học phần điều kiện.

1. Các vấn đề và hướng giải quyết

* Các vấn đề:
* Hiện tại em chỉ mới thực hiện được với các học phần chỉ có duy nhất 1 học phần điều kiện.
* Với các học phần có nhiều học phần điều kiện, em vẫn đang test các thuật toán có thể nghĩ ra để có thể lấy toàn bộ dữ liệu các môn học đó. Tuy nhiên, em thấy nếu xây dựng được cây đầy đủ học phần điều kiện thì có lẽ sẽ phải sử dụng thuật toán đệ quy. Em vẫn đang tìm cách tối ưu để thực hiện được yêu cầu đặt ra ạ.
* Hướng giải quyết:
* Test các thuật toán khác nhau.
* Em mong thầy có thể cho em lời khuyên ạ.