**Đề bài**: Hãy tạo 1 chương trình quản lý sinh viên:

Yêu cầu:

* Tạo 1 menu chức năng, ví dụ chọn 1 để xem thông tin sinh viên, chọn 2 để thêm sinh viên, chọn 3 để xóa sinh viên, chọn 4 để xuất danh sách ra file .txt, …, có thể thêm nếu muốn. **Thêm 1 chức năng mới là tính thống kê về lớp và xuất ra graph nếu được**. Chức năng xuất mình nên thêm là chọn dạng xuất mong muốn chẳng hạn.
* Cuối cùng nhấn nút 6 để escape program.

=> **Mình có thể suy nghĩ tới function Switch**.

Mỗi sinh viên có những trường thông tin như sau: **Sử dụng Struct chỗ này**.

* Họ tên
* Năm sinh
* Giới tính (**có thể cho lựa chọn giữa 2 lựa chọn chẳng hạn, tính năng nâng cao**)
* Lớp
* Tình trạng sinh viên (**tương tự, đang học hoặc bị thôi học như cái giới tính**)
* Quê quán (**có thể thêm tính năng người nước ngoài nếu không rơi vào 63 tỉnh thành**)
* Số điện thoại

For

If

While loop

Trước tiên mình xử lý từng phần trước:

* Phần tạo các option cho menu là dễ nhất thành ra ta sẽ đi làm phần đó đầu tiên. Ở đây, ta thích sử dụng cái GUI cũ của MATLAB.
* **Để giúp chương trình thân thiện với người dùng**.
* Ta code được 1 phần của chương trình MATLAB như sau:

**disp('This is a student information management program');**

**disp('Choose 1 if you want to see student info');**

**disp('2 if you want to add new student');**

**disp('3 to delete student');**

**disp('4 to export the data to .txt file');**

**choice = menu('Please choose which options you like',1, 2, 3, 4);**

**switch choice**

**case 1**

**disp('ll');**

**case 2**

**disp('22');**

**case 3**

**disp('33');**

**otherwise**

**disp('44');**

**end**

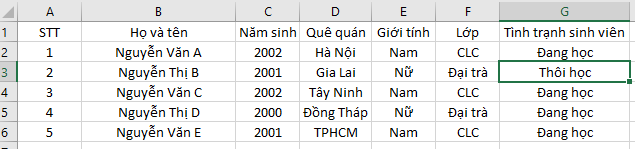
**Ở đây ta xây dựng trước mấy hướng đi để sau này đi vào code chuyên sâu hơn. Bây giờ dựng vậy cho có cái sườn thôi**.

Ở đây mục đích ta cho mấy cái disp là để cho nó đỡ dài dòng nếu ta để nó dưới dạng message trong menu thôi.

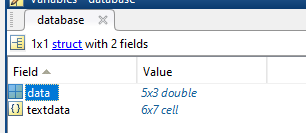
* Tuy nhiên ta thích thêm cái **while loop**.
* Để cái code dài hơn cho ngầu.
* Ngăn người dùng bấm nút x, để khiến cho người dùng phải nhập các giá trị 1, 2, 3, 4
* **Tuy nhiên như vậy ta phải thêm 1 nút chỉ để dành riêng cho việc kết thúc chương trình**.

Tiếp theo ta thấy các tính năng còn lại điều liên quan tới database nên trước tiên ta cần tạo 1 cái database.

* **Ban đầu hướng tư duy của mình là tạo 1 file excel và sau đó sẽ import nó vào và biến nó thành 1 file struct** (giống hình dưới). Vì file struct cực kì tiện xử lý trong MATLAB.



* Ngoài ra đa số hiện nay người ta sẽ nhập dữ liệu vào file excel rồi mới đưa vào những phần mềm nâng cao hơn để xử lý. Nên ta sẽ làm y chang thực tế luôn.
* Tiếp theo ta cần 2 hàm, **1 hàm để import** và **1 hàm để convert table to struct**.
* Về cái table to struct rất may là ta tìm ra 1 hàm tên **table2struct**. **Hàm này sẽ converting table to structure array**. Như vậy qua đó ta biết rằng để import data thì ta cần 1 cái function import vào dạng table.
* **Như vậy cái function importdata không xài được vì nó import vào dưới dạng struct và đó không phải dạng struct ta mong muốn**. Nó như sau:



Code sai:

**database = importdata('student database.xlsx');**

* Nó chia số với chữ ra riêng. Như vậy không được.
* Đổi lại ta sẽ dùng function **readtable**, **có tác dụng create table from file**. Và cú pháp thì ta chỉ cần nhập tên file thôi.
* Để đỡ tốn dòng code và đỡ khai báo biến thì ta kết hợp **table2struct** và **readtable** như sau:

**database = table2struct(readtable('student database.xlsx'));**

Thực ra rất dễ, readtable tạo ra cái table xong dùng chính kết quả đó bỏ vào table2struct rồi lưu vào biến là xong.

Như vậy về cơ bản là ta đã xong phần import data rồi. Bây giờ ta đi vào xử lý từng cái nhánh.

* Xử lý nhánh 1 luôn dễ nhất vì chỉ là view thôi.
* Để view thì view dạng table sẽ trực quan với dễ thực hiện hơn là view dạng struct.
* Như vậy câu code ở trên để import data ta cần sửa lại chút như sau:

**table\_form = readtable('student database.xlsx');**

**struct\_form = table2struct(table\_form);**

Mục đích để ta tách ra như vậy là để ta có cả 2 dạng là dạng table form và dạng struct form để dễ xử lý.

Và đây là câu code để view database:

**switch choice**

**case 1**

**disp(table\_form);**

**Để xem data thì ta đơn giản chỉ cần disp dạng table form**.

**Với case 2 thì ta sẽ sử dụng những gì ta được học trong bài tạo tài khoản với mật khẩu**.

* Cụ thể là như sau:

**z = numel(acc)+1;**

**acc\_z\_ten\_dang\_nhap = input('create new username: ');**

**acc\_z\_pass = input('create your password: ');**

**acc(z).ten\_dang\_nhap = acc\_z\_ten\_dang\_nhap;**

**acc(z).pass = acc\_z\_pass;**

**save('du\_lieu.mat','acc');**

Giải thích dòng code trên:

* Đầu tiên acc là 1 cái struct, ở đây ta đếm số element đang có ở trong cái struct đó.
* **Sau đó ta có z là 1 cái biến đếm số, ở đây z = numel + 1 nghĩa là ta sẽ tạo ra 1 account mới và cái z sẽ có tác dụng trong việc chỉ số thứ tự trong database**.
* Sau đó ta lưu tạm tài khoản và mật khẩu vào 2 biến tạm.
* **Cuối cùng gán 2 giá trị của 2 biến tạm đó vào cái struct theo thứ tự z thôi**. Khá đơn giản.

**Qua đó học được cách để kiểm tra từng dòng code xem nó chạy tốt không mà không ảnh hưởng tới chương trình thì câu code nào ta chắc đúng hãy cho nó là code, còn không thì để dạng comment hết rồi chạy từ từ**.

Thì ban đầu về cơ bản ta có sườn code như sau:

**case 2**

**n = numel(struct\_form)+ 1; Cái +1 này để nhằm mục đích là tránh việc ghi đè**

**new\_student\_fullname = input('please enter new student full name: ');**

**new\_student\_year\_of\_birth = input('please enter new student year of birth: ');**

**new\_student\_place\_of\_birth = input('please enter new student place of birth: ');**

**new\_student\_gender = input('please enter new student gender: ');**

**new\_student\_class = input('please enter new student class: ');**

**new\_student\_status = input('please enter new student status: ');**

Mấy cái này sẽ như cái bài password sẽ được lưu vào struct sau.

Tuy nhiên ở đây ta muốn nâng cấp thêm xíu, nếu giới tính mà để người dùng nhập vào thì sẽ rất dễ xảy ra tình trạng sau:

* **Cùng là male nhưng có thể có người nhập Male, có người nhập male, sau này sẽ gây ra rất nhiều khó khăn trong việc filter**.
* **Tiếp theo là để ngăn người dùng nhập giá trị không phải male hay female luôn, đỡ tạo data lỗi**.

Và ta có dòng code bên dưới

**new\_student\_gender = menu('please select gender', 'male', 'female');**

**switch new\_gender**

**case 1**

**new\_student\_gender = 'male';**

**otherwise**

**new\_student\_gender = 'female';**

**end**

Giải thích code xíu:

* **Ta tạo 1 biến tên gender và ở đó display menu để chọn giữa male và female**.
* **Nếu male hay female thì đều dùng switch case và lưu giá trị đó lại vào new gender**.

Việc cuối cùng chỉ là lưu lại vào struct thôi:

**struct\_form(n).No = n;**

**struct\_form(n).FullName = new\_student\_fullname;**

**struct\_form(n).YearOfBirth = new\_student\_year\_of\_birth;**

**struct\_form(n).PlacesOfBirth = new\_student\_place\_of\_birth;**

**struct\_form(n).Gender = new\_student\_gender;**

**struct\_form(n).Class = new\_student\_class;**

**struct\_form(n).StudentStatus = new\_student\_status;**

Cái này khá giống cái tạo tài khoản mật khẩu nên không cần giải thích nhiều.

* **Nhưng ở đây ta cũng muốn đổi cái class và cái status thành lựa chọn nhị nguyên giống gender, vì trong BK lớp với status cũng chỉ có vài option lựa chọn thôi**.

=> **Cách làm sẽ y chang gender**, chỉ là khác tên.

Ở đây ta nên suy nghĩ là làm sao sau khi nhập tên học sinh mới thì vẫn quay lại vòng lặp trừ khi bấm nút khác mà thôi.

**Tiếp theo ta xử lý tính năng xuất đi tại nó có vẻ dễ hơn việc xóa**.

* Ở đây ta muốn tăng độ khó bằng cách cho người dùng có thể chọn định dạng file mà họ có thể xuất luôn, đó là giữa excel và text file.

**data\_type = menu('please choose the data type you want', 'excel', 'txt file');**

* Về tính năng xuất thì tính năng xuất dễ nhất đó là xuất dưới dạng text file. **Cho nên giờ ta cần tìm 1 hàm có thể xuất dưới dạng text file**.
* Ban đầu mình tính xài fprintf, nhưng cái fprintf thì rất khó chịu trong việc sắp xếp số liệu các thứ nên mình quyết định tìm 1 function khác, **có thể nó có lựa chọn là xuất file ra thành nhiều dạng khác nhau chẳng hạn**.
* Thì ở đây ta tìm ra được 1 hàm khá hay tên **writetable**. **Đây là 1 function cho phép write table to file**, **điều đó khiến ta phải đặc biệt chú ý tới format đó là chỉ cho phép table mà thôi**.

Cho nên ta phải quay lại cái nhánh khi chọn switch case = 2 đó là khi thêm học sinh mới vào.

* Khi thêm học sinh mới, ta cần phải lưu database đó vào, thông thường nó sẽ lưu vào dạng **struct**. Thành ra ở đây, ta phải nghĩ cách để convert dạng này sang **table**. Sau khoảng 20p, giải pháp code của ta như sau:

**table\_form = struct2table(struct\_form);**

**writetable(table\_form, 'student database.xlsx');**

Mục đích 2 dòng code này là như sau:

* Khi ta tạo thông tin sinh viên mới, dữ liệu đó đang tạm thời được lưu dưới dạng struct.
* Sau đó ta convert nó sang dạng table form và lưu vào biến **table\_form**.
* Ở đây vì dữ liệu để được để ở dạng table, rất dễ để ta sử dụng cái function **writetable** để lưu dữ liệu mới đó vào file excel ban đầu mình có sẵn.

Sau đó ở cái xuất file thì khá dễ thôi. Ta chỉ cần viết đơn giản như sau:

**case 4**

**data\_type = menu('please choose the data type you want', 'excel', 'txt file');**

**switch data\_type**

**case 1**

**writetable(table\_form, 'student database final.xlsx', 'Sheet', 1);**

**case 2**

**writetable(table\_form,'student database final.txt','Delimiter',' ');**

**end**

Tiếp theo ta chuyển sang phần tính statistics cho file đi tại vì nó dễ hơn việc xóa.

* **Ở đây mình sẽ tận dụng kiến thức mình học trong lập trình R và subplot để vẽ graph**.

Đầu tiên mình muốn vẽ 1 cái pie chart thể hiện %nam và %nữ có trong cái database này.

* **Trước tiên liên hệ với R mình biết rằng để làm việc hiệu quả với table thì ta trước tiên phải lấy cái subset của table đó ra**. Vì ở đây ta chỉ quan tâm gender nên ta sẽ lấy subset của gender ra là xong.
* **Sau đó ta sẽ convert cái table đó sang array. Vì matlab chỉ làm việc với array.**
* **Sau đó từ cái array đó ta chỉ cần đếm cái string pattern có trong đó là xong.**

c = {'car' , 'tree' , 'car' , 'bag' , 'horse' , 'car' , 'tree'};

nnz(strcmp(c,'car'))

[**https://www.mathworks.com/help/fixedpoint/ref/embedded.fi.nnz.html#:~:text=Description,-example&text=N%20%3D%20nnz(%20X%20)%20returns,fewer%20than%20232%20elements**](https://www.mathworks.com/help/fixedpoint/ref/embedded.fi.nnz.html#:~:text=Description,-example&text=N%20%3D%20nnz(%20X%20)%20returns,fewer%20than%20232%20elements)**.**

Và sau 1 một hồi tìm kiếm và thử thì ta có dòng code như sau:

**gender\_subset = table\_form( : , 5);**

**array\_gender = table2array(gender\_subset);**

**number\_of\_male = nnz(strcmp(array\_gender, 'Male'));**

Giải thích dòng code sau:

* Đầu tiên ta lấy subset của cái table bằng cách đơn giản là specify cái vùng ta muốn lấy thôi. **( : ,5)** có nghĩa là ta lấy tất cả các hàng của cột 5.
* **Sau đó khi ta có subset rồi, việc tiếp theo ta làm chỉ là convert table2array dùng cái function có sẵn**.
* Sau khi convert xong rồi, ta đếm số ‘Male’ có trong array. **Ở đây ta dùng kết hợp 2 function**.
* Function strcmp đầu là để compare và sẽ trả ra kết quả 1 nếu trùng khớp và trả ra kết quả 0 nếu không trùng khớp.
* Sau đó ta dùng function **nnz** nghĩa là **Number of Non Zero element, để đếm số giá trị khác 0, ở đây là đếm giá trị 1 trùng khớp**.
* Sau đó việc còn lại khá đơn giản, làm thêm 1 cái cho female và vẽ piechart là xong.

**no\_female = nnz(strcmp(array\_gender, 'Female'));**

**pie\_chart\_data = [no\_male no\_female];**

**pie(pie\_chart\_data);**

Tiếp theo ta sẽ thêm dữ liệu vào để vẽ histogram vùng miền chẳng hạn, đại khái có nhiều câu code cho nó hay.

* Như ta đã học trong R, thì có 2 loại variables đó là **categorical variables** and **numerical variables**.
* Thông thường biến **string** sẽ là 1 categorical, mà thể hiện rõ nhất là trong histogram graph.

=> **Cho nên điều đó đòi hỏi ta phải biến đổi cái data của ta từ dạng string sang categorical variable**.

Các bước biến đổi như sau:

* Phải lấy 1 cái subset về vùng miền trong cái table form của data ta ra.
* **Tiếp theo là biến nó từ table form sang dạng array**. Ở đây ta có thử biến đổi nó từ table form sang categorical thì không được. Nói chung dạng array là dạng cơ bản MATLAB làm việc trên đó.
* **Tiếp theo từ dạng array sang dạng categorical bằng cách dùng function categorical**.
* Cuối cùng dùng biến đó để vẽ histogram là xong. Ta có code như sau:

**subplot(2, 2, 2);**

**x = linspace(0, 10, 100);**

**PlacesOfBirth\_subset = table\_form( : , 4);**

**array\_PlacesOfBirth = table2array(PlacesOfBirth\_subset);**

**categorical\_form\_of\_POB = categorical(array\_PlacesOfBirth);**

**histogram\_graph = histogram(categorical\_form\_of\_POB);**

1 phần của việc ta biết là histogram graph phải làm việc với categorical value đó là do ta vào phần documentation của MATLAB và ta thấy

<https://www.mathworks.com/help/matlab/ref/matlab.graphics.chart.primitive.histogram.html>

**Có 1 cái graph mẫu thì khi mà nó display số phiếu yes và phiếu no thì data của nó phải được để dưới dạng categorical variable mới hiển thị được**.

=> Sau khi xem cái trang types of MATLAB Plots, ta thấy có 1 plot khá thú vị tên là **geobubble**, đại khái nó sẽ **visualize data values at specific geographic locations**.

=> **Thành ra ta quyết định thay vì dùng histogram để thể hiện sự phân bố của geographic location của Places of Birth, ta quyết định dùng histogram để xài cho year of birth còn Places of Birth sẽ dùng geobubble graph**.

Để xài được 1 cái graph, thì ta đơn giản chỉ cần vào phần documentation của graph đó là xong:

* Cú pháp của code đó như sau:
* **geobubble( tbl, latvar, lonvar);**
* **Với tbl là table containing data to be plotted.**
* **latvar là variable for latitude.**
* **lonvar là variable for longitude.**

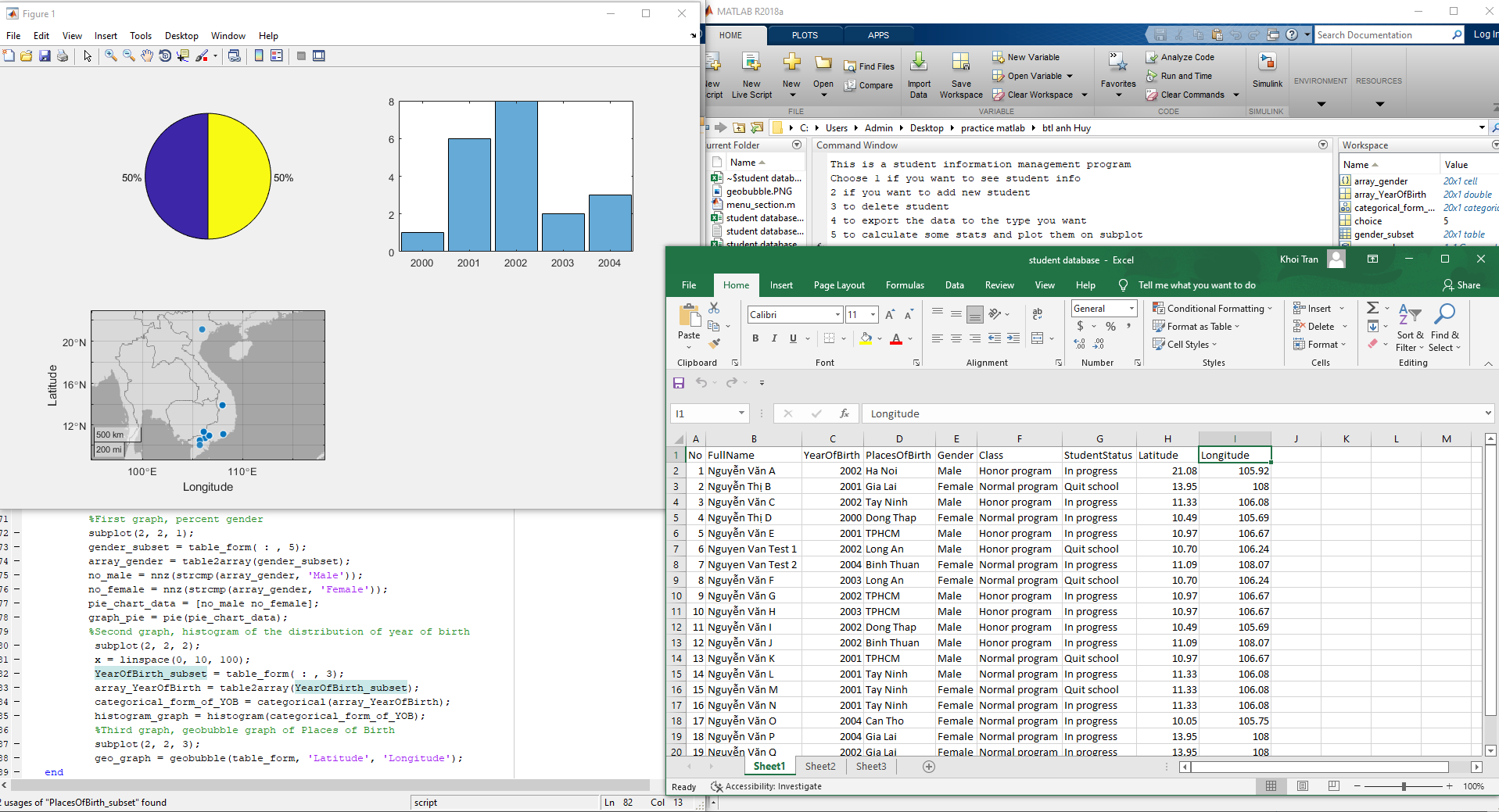
**=> Như vậy qua đó ta tư duy rằng graph này có thể sử dụng data dưới dạng table.**

**=> Như vậy ta không cần thiết phải thực hiện các bước biến đổi data phức tạp.**

Trong trang web của Matlab chỉ cách sử dụng geobubble graph thì ta thấy người ta có 1 ví dụ về cách vẽ graph. Bên cạnh có try this example nên ta nhấn vào. Cụ thể các bước như sau:

* Import data của ta vào dưới dạng table form, cái này ta có thể sử dụng function readtable.
* **Sau đó thì dùng function geobubble thôi, specify cái table form của data ta, và 2 cái variable chứa giá trị Latitude và Longitude là xong**.
* Code như sau:

**geo\_graph = geobubble(table\_form, 'Latitude', 'Longitude');**



Tuy ta đã vẽ ra được 1 cái graph rất đẹp như sau, nhưng ta muốn đi xa hơn nữa. **Ta muốn là độ to nhỏ của mấy cái bubble này tỷ lệ thuận với lại cái frequency của từng tỉnh ở trong cái table data này**.

* **Cho nên bước tiếp theo ta làm là tìm trên mạng xem MATLAB có hỗ trợ việc tạo frequency table không**.
* Và ta tìm ra được 1 hàm khá hay, gọi là **tabulate**. Function tabulate có cú pháp như sau:

**tabulate( X );**

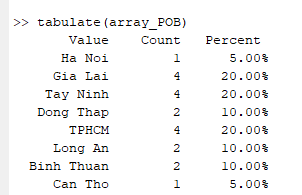
**Với x là 1 vector. Như vậy ta phải thêm bước chuyển đổi data từ table form sang dạng vector**.

**POB\_subset = table\_form( : , 4);**

**array\_POB = table2array(POB\_subset);**

**tabulate(array\_POB)**

Và nó hiện ra kết quả như sau:



* Ta dự định dùng cái percent như 1 cách để specify cái size.
* Tuy nhiên ta nhớ ra 1 điều, là nếu dùng cái bảng tabulate này thì sẽ không được vì nó thiếu cái latitude và longitude. Trong khi nếu dùng bảng table\_form sẽ thiếu cái percent. Như vậy ta phải nghĩ cách add cột vào bảng với nhau:

Code: Bên dưới là 1 đoạn code để ta add thử. Ta định add cái frequency của các tỉnh vào giữa cột student status và latitude.

**h = tabulate(array\_POB)**

**o = h( : , 3)**

**new\_table = addvars(table\_form, o, 'StudentStatus', 'Latitude')**

**Nó disp lỗi là không cùng number of rows**. Điều đó là vì khi ta đếm frequency, thì số row về cơ bản là đã giảm rồi. Do những ô có cùng tỉnh thì được gom lại để đếm. Nên lỗi không cùng row là đương nhiên.

=> Như vậy ta phải nghĩ ra 1 cách khác. Cách đó chính là:

* **Đầu tiên tính frequency của từng tỉnh, convert cái đó thành dạng table form.**
* **Tiếp theo từ cái biến table\_form ban đầu, tách cái cột latitude và longitude ra riêng, tiếp tục cho vào frequency table. Mục đích là để gom nó lại cho cùng number of rows.**
* **Cuối cùng lấy 2 cái cột đó add vào cái table frequency của tỉnh là xong.**
* **Cuối cùng dùng table đó để vẽ geobubble là xong.**

**POB\_subset = table\_form( : , 4);**

**array\_POB = table2array(POB\_subset);**

**fre\_of\_POB = tabulate(array\_POB);**

**table\_new = cell2table(fre\_of\_POB);**

**lat\_subset = table\_form( : ,8);**

**long\_subset = table\_form( : ,9);**

**array\_lat = table2array(lat\_subset);**

**array\_long = table2array(long\_subset);**

**fre\_lat = tabulate(array\_lat);**

**fre\_long = tabulate(array\_long);**

**lat\_value\_fre = fre\_lat( : ,1);**

**long\_value\_fre = fre\_long( : ,1);**

**new\_table = addvars(table\_new, lat\_value\_fre, long\_value\_fre);**

**new\_geo = geobubble(new\_table, 'lat\_value\_fre', 'long\_value\_fre', 'SizeVariable', 'fre\_of\_POB2');**

**Về mặt cơ bản là disp size được rồi đó nhưng mà nó đang lệch latitude với longitude thôi.** Tức là latitude, longitude không có khớp với tỉnh. Cho nên ở đây ta phải sắp xếp lại xíu.

* **Sắp xếp theo từ té tới lớn, theo frequency**.
* Mục đích là vì frequency của từng tỉnh sẽ ứng với frequency của cặp tọa độ luôn. Tức là Ví dụ TPHCM có frequency cao nhất thì frequency tọa độ của TPHCM cũng cao nhất. Như vậy ta chỉ cần sắp xếp frequency của các tỉnh từ bé tới lớn, sắp xếp frequency của tọa độ từ bé tới lớn.
* **Cuối cùng trong frequency của tọa độ, tách cột tọa độ ra quăng vào bảng frequency tỉnh là ta sẽ có 1 bảng trọn vẹn**.

**POB\_subset = table\_form( : , 4);**

**array\_POB = table2array(POB\_subset);**

**fre\_of\_POB = tabulate(array\_POB);**

**fre\_of\_POB = sortrows(fre\_of\_POB, 2);**

**table\_new = cell2table(fre\_of\_POB);**

**lat\_subset = table\_form( : ,8);**

**long\_subset = table\_form( : ,9);**

**array\_lat = table2array(lat\_subset);**

**array\_long = table2array(long\_subset);**

**fre\_lat = tabulate(array\_lat);**

**fre\_lat = sortrows(fre\_lat, 2);**

**fre\_long = tabulate(array\_long);**

**fre\_long = sortrows(fre\_long, 2);**

**lat\_value\_fre = fre\_lat( : ,1);**

**long\_value\_fre = fre\_long( : ,1);**

**new\_table = addvars(table\_new, lat\_value\_fre, long\_value\_fre);**

**new\_geo = geobubble(new\_table, 'lat\_value\_fre', 'long\_value\_fre', 'SizeVariable', 'fre\_of\_POB2');**

* **Sort thì cũng có nhiều loại sort**. Function **sort** bình thường ta tìm là sort array. Tức là ví dụ ta có 1 array sau:

[ 2 5 8 1 4 3 ]

* Khi sử dụng sort array nó sẽ sắp xếp được cho mình.
* Tuy nhiên do cái ta đang làm việc là 1 cái cell array. Nên nó không dùng được cái sort bình thường. Đó là lý do ta sử dụng function **sortrows**. Function này cho phép ta **sort rows of matrix or table**.

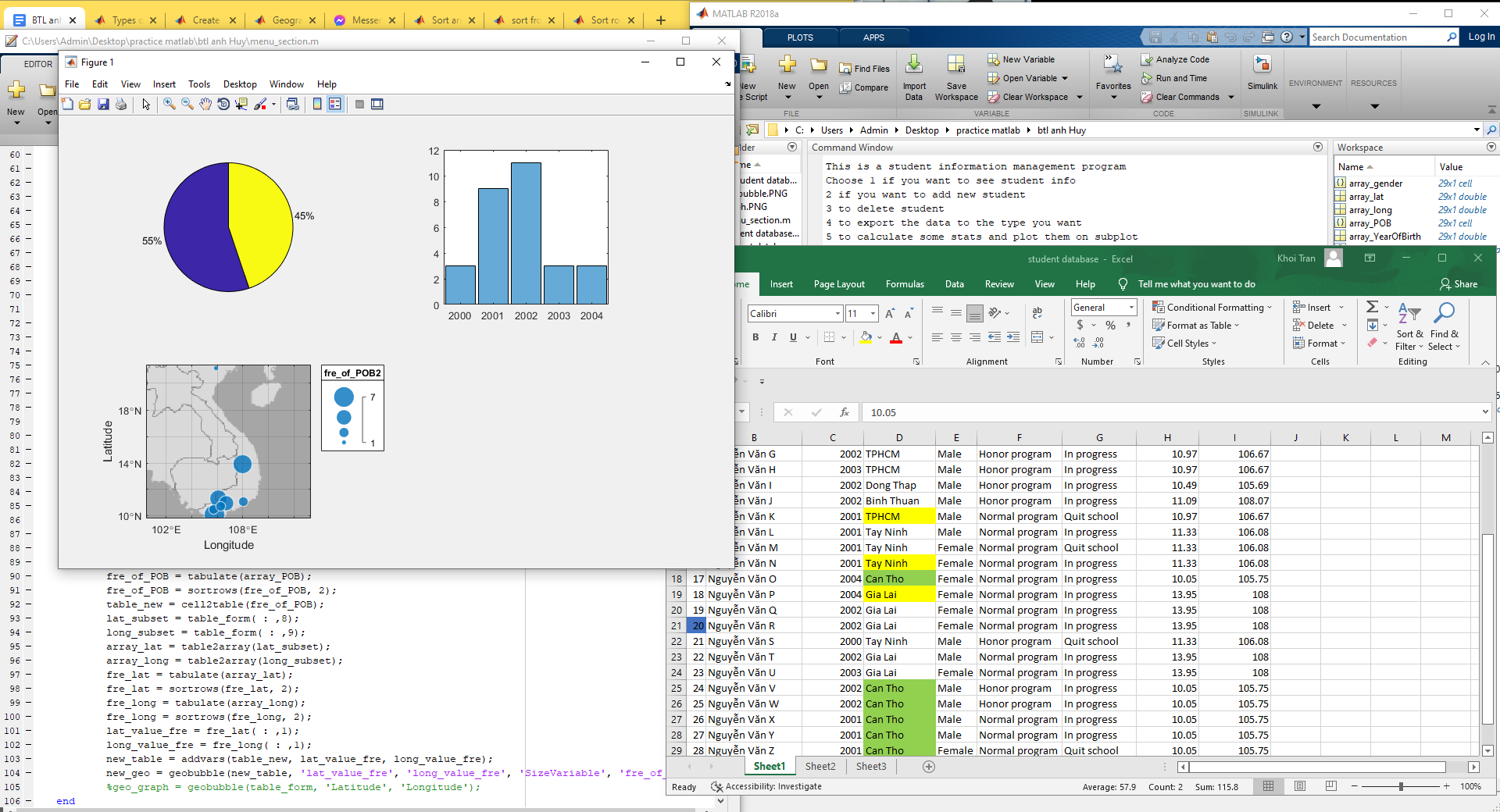
<https://www.mathworks.com/help/matlab/ref/double.sortrows.html>

* Về mặt cú pháp nó sẽ như sau:

**sortrows( A, column )**

* Với A là matrix hoặc table của mình.
* **Column chính là cái cột mình muốn sort. Nó sẽ luôn sort theo giá trị từ bé tới lớn. Cái này rất giống với Excel**.

Như vậy ta đã xong phần code bubble và nó ra kết quả như sau:



**Bây giờ ta sẽ xử lý sang phần xóa dữ liệu sinh viên**.

Ta biết cách tốt nhất để remove a row or column from a matrix **is to set that row or column equal to a pair of empty square brackets [ ]**.

* Như vậy ta tư duy có thể table nó cũng xài cách y chang, và ta search trên mạng và ra được cách sau:

<https://www.mathworks.com/help/matlab/matlab_prog/add-and-delete-table-rows.html>

* **Về cơ bản nó y chang**. Thì đây là cách ta tư duy để cho người dùng có thể xóa hàng mà họ muốn.

**i = input('please choose row number you want to delete: ');**

**table\_form(i, : ) = [];**

Giải thích code:

* **Về cơ bản ta đặt i là cái row number mà người dùng muốn delete.**
* **Sau đó ta chỉ cần cho i là 1 biến chỉ vị trí hàng thôi, xong cột chọn hết và gán giá trị hàng i đó cho [ ] tập hợp rỗng là xong chuyện**.

Tuy nhiên ta muốn nâng mức độ lên 1 mức cao hơn:

**delete\_row\_options = menu('Please choose number of rows you want to delete', 1, 2, 3);**

* **Tức là ở đây ta cho phép người dùng chọn số dòng họ muốn delete luôn**.

Nếu họ chọn 1 dòng thì quá dễ.

**disp(table\_form);**

**r = input('please choose row number you want to delete: ');**

**table\_form(r, : ) = [];**

**disp(table\_form);**

Code này giống cái trên ta làm mẫu thôi.

Còn với xóa 2 và 3 dòng thì ta làm code như sau:

**disp(table\_form);**

**r1 = input('please enter the first row you want to delete: ');**

**r2 = input('please enter the second row you want to delete: ');**

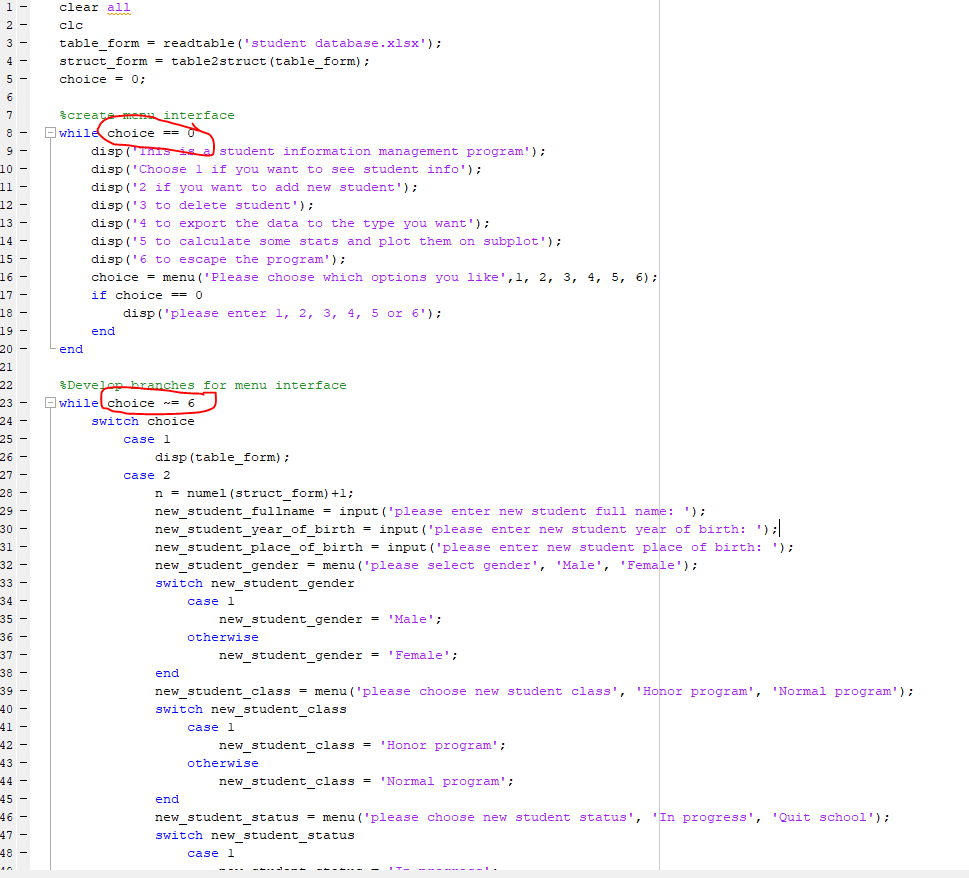
**table\_form([r1 r2], : ) = [];**

**disp(table\_form);**

Càng muốn xóa nhiều dòng thì đặt nhiều biến như r1, r2, r3,... rồi bỏ vào matrix dưới là xong.

* **Ở đây, r1 và r2 sẽ đại diện cho các dòng mà ta muốn xóa, sau đó nó sẽ đóng vai trò trong dòng code bên dưới như 1 số chỉ dòng số mấy mà ta muốn xóa (gán giá trị rỗng) vào thôi**.

**Cuối cùng ta đặt chương trình bấm nút 6 để escape**.



* Lý do ta có 2 cái while loop là vì 1 cái while loop == 0 là để ngăn người dùng nhấn dấu X.
* Còn cái while loop ~= 6 là để ép người dùng chỉ có 1 cách kết thúc chương trình đó là nhấn số 6.