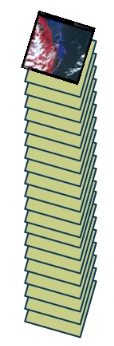
**Bài tập 3: Viết các Hàm Theo Nhu Cầu Sử dụng và Làm Việc Với Tập Ảnh**

****

**Giới Thiệu**

Code Editor cho phép người dùng thực hiện được tất cả các chức năng của Earth Engine, tuy nhiên, cần có hiểu biết cơ bản về lập trình và JavaScript. Trong bài tập này, chúng ta sẽ tiếp tục học về ngôn ngữ lập trình Java và các khái niệm mới về dữ liệu không gian trong Earth Engine. Bạn sẽ tiếp tục những gì được học ở bài tập 2 về đối tượng ảnh; tuy nhiên trong bài học này bạn sẽ tập trung nhiều hơn vào các tập hợp hoặc chồng lớp của các đối tượng ảnh có tính chất tương tự. Trong bài học này, bạn sẽ tập trung vào các khái niệm cơ bản và phương pháp liên quan đến các tập hợp ảnh trong Earth Engine. Đây là hướng dẫn giúp người sử dụng có thể viết được các sript đơn giản trong JavaScript.

**Mục Tiêu**

* Tập viết và chạy code trong Earth Engine Code Editor để truy nhập vào tập hợp dữ liệu ảnh, lọc theo thời gian (theo ngày) và không gian (ví dụ như khu vực nghiên cứu của bạn), thao tác với các đối tượng hình học, và tìm hiểu một số phương pháp xử lý ảnh

**Yêu cầu tài liệu**

* Một tài khoản Google Earth Engine
* *Gợi ý*: Cài đặt Google Chrome vào máy tính của bạn

**Mục Lục**

[Phần 1: Làm việc với tập ảnh 3](#_heading=h.gjdgxs)

[Phần 2: Lọc Tập Ảnh Theo Không Gian 4](#_heading=h.30j0zll)

[Phần 3: Lọc Tập Ảnh Theo Thời Gian 7](#_heading=h.3znysh7)

[Phần 4: Tìm hiểu các hàm xử lý ảnh 8](#_heading=h.2et92p0)

[Phần 5: Giới Thiệu về Reducers 8](#_heading=h.tyjcwt)

[Phần 6: Nhập Môn Về Hàm 9](#_heading=h.3dy6vkm)

[Phần 7: Tạo hàm lọc bỏ mây (cloud mask) 13](#_heading=h.1t3h5sf)

[Phần 8: Các Hàm Thao Tác Với Tập Hợp Ảnh 15](#_heading=h.4d34og8)

Phần 1: Làm việc với tập hợp ảnh

Một tập hợp ảnh là một nhóm các ảnh cùng loại trong Earth Engine. Ví dụ như, tất cả ảnh Landsat 8 trong GEE là một tập hợp ảnh

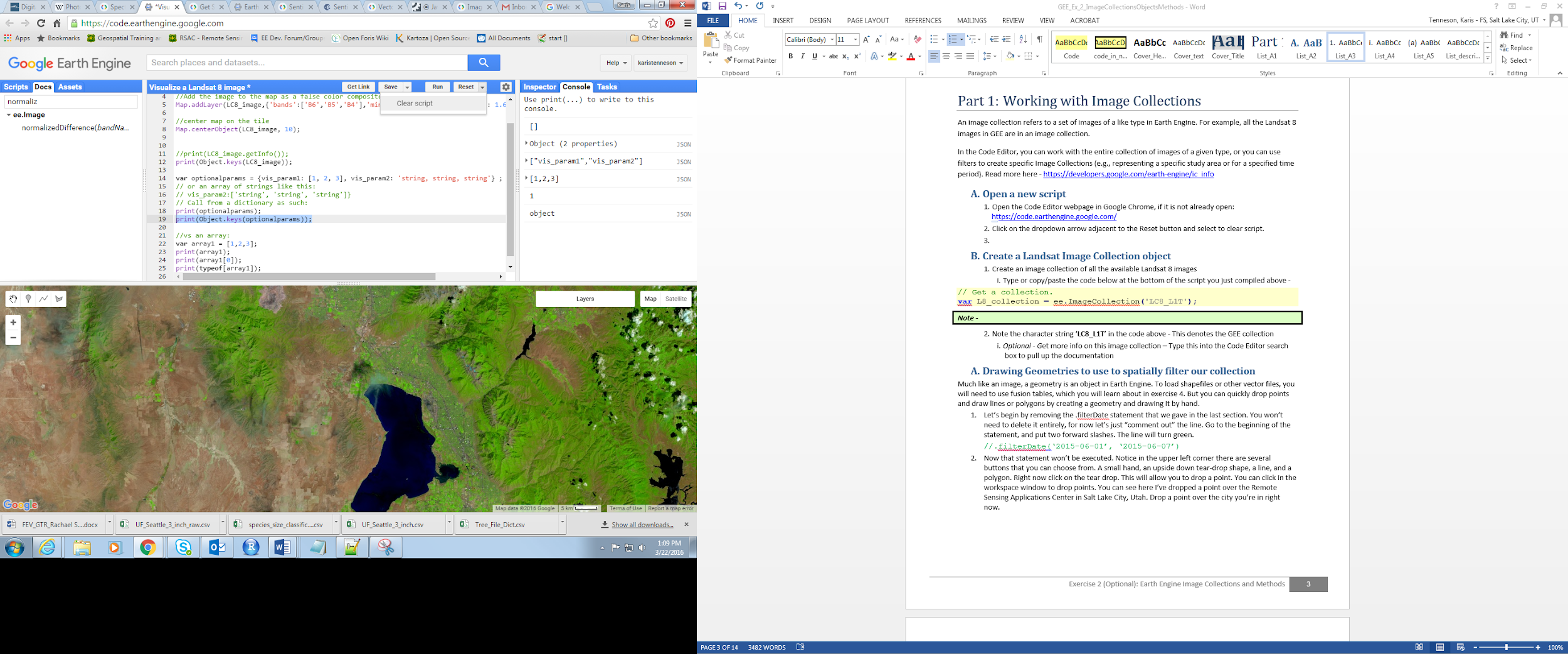
Trong Code Editor, bạn có thể làm việc với toàn bộ tập hợp ảnh ở một dạng nhất định, hoặc bạn có thể sử dụng các bộ lọc để tạo ra một tập ảnh cụ thể (ví dụ như để thể hiện khu vực nhiên cứu của bạn hoặc cho một khoảng thời gian cụ thể). Đọc thêm tại <https://developers.google.com/earth-engine/ic_info>

* 1. **Mở một Script mới**

**Mở** trang web Code Editor trong Google Chrome

<https://code.earthengine.google.com/>

**Click** vào biểu tượng đầu mũi tên cạnh nút Reset và chọn **Clear script**.



* 1. **Tạo một tập hợp đối tượng ảnh Landsat**

**Tạo** một tập hợp các ảnh Landsat 8 có sẵn bằng cách **gõ hoặc copy/paste** dòng code sau vào code editor

// Get a collection.

**var** L8\_collection **=** ee**.**ImageCollection**(**'LANDSAT/LC8\_L1T\_TOA'**);**

Tiếp theo chúng ta sẽ hiển thih nó trên cửa sổ bản đồ, **copy** code dưới đây và **paste** vào code editor. Sau đó **click Run** để thực hiện toàn bộ các dòng lệnh.

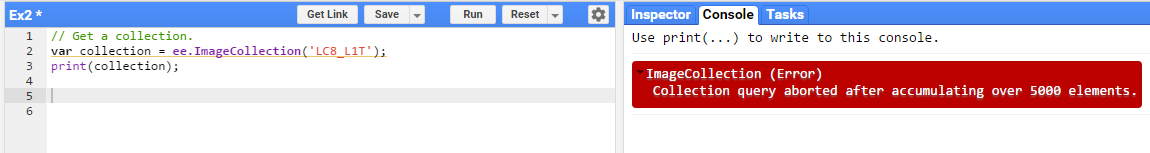
Map**.**addLayer**(**L8\_collection**,** **{** min:0.05, max: 0.8, bands: 'B6, B5, B4' **},** 'Landsat Collection'**);**

Map**.**setCenter**(**100.56**,** 13.94**,** 7**);**

* + - 1. Bạn thấy gì trên cửa sổ bản đồ? Tập ảnh gồm nhiều ảnh, nhưng không phải tất cả đều được hiện lên. Khi chúng ta sử dụng hàm **Map.addLayer** để đưa một tập ảnh vào cửa sổ bản đồ, theo mặc định/default chỉ các pixel mới nhất sẽ được hiển thịCent
    1. *(Không bắt buộc)* Nếu bạn thêm vào dòng lệnh print/in, bạn có thể xác định có bao nhiêu ảnh ở trong tập hợp. **Copy** và **paste** dòng sau vào dưới phần script của mình

print**(**L8\_collection**);**

* + - 1. Tuy nhiên, vì tập ảnh của chúng ta tương đối lớn, chúng ta nhận được báo lỗi trên Console *“Collection query aborted after accumulating the metadata for over 5,000 elements”*. Có nghĩa là yêu cầu in tập ảnh không thực hiện được khi gom toàn bộ siêu dữ liệu của hơn 5,000 đối tượng ảnh. Chúng ta sẽ lọc tập ảnh theo không gian hoặc thời gan và tìm cách khác để có số lượng ảnh phù hợp trong tập ảnh cần cho việc sử dụng.



* + - 1. **Comment out (//)** lên trướclệnh print/in**.** Chúng ta sẽ trở lại sau

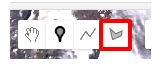
// print(L8\_collection);

Phần 2: Lọc Tập Ảnh Theo ranh giới Không Gian

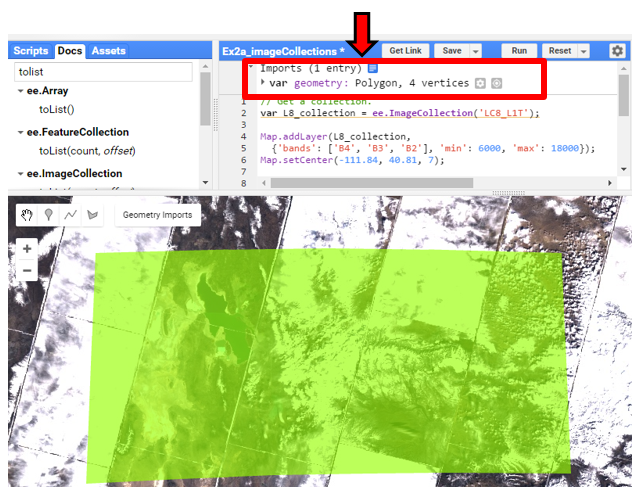
* 1. **Vẽ một đối tượng hình học tạo ranh giới không gian để lọc tập ảnh**

Đối tượng hình học (Geometry) là một dạng đối tượng không gian địa lý khác trong Earth Engine. Để tải các shapefile hoặc các file vector khác, bạn cần sử dụng Fusion Tables, sẽ được học trong bài tập 4. Bạn có thể vẽ nhanh một đường/line hoặc một vùng/polygon bằng tay hoặc các điểm/point để tạo một đối tượng hình học.

* + 1. Phía góc trên bên trái của cửa sổ bản đồ/map window, có một số nút chức năng bạn có thể sử dụng để vẽ các đối tượng hình học. Các nút này bao gồm một biểu tượng bàn tay nhỏ (để di chuyển xung quanh ảnh), một biểu tượng giọt lệ ngược, một biểu tượng đường, và một biểu tượng vùng. **Click** vào biểu tượng **vùng/ polygon**.

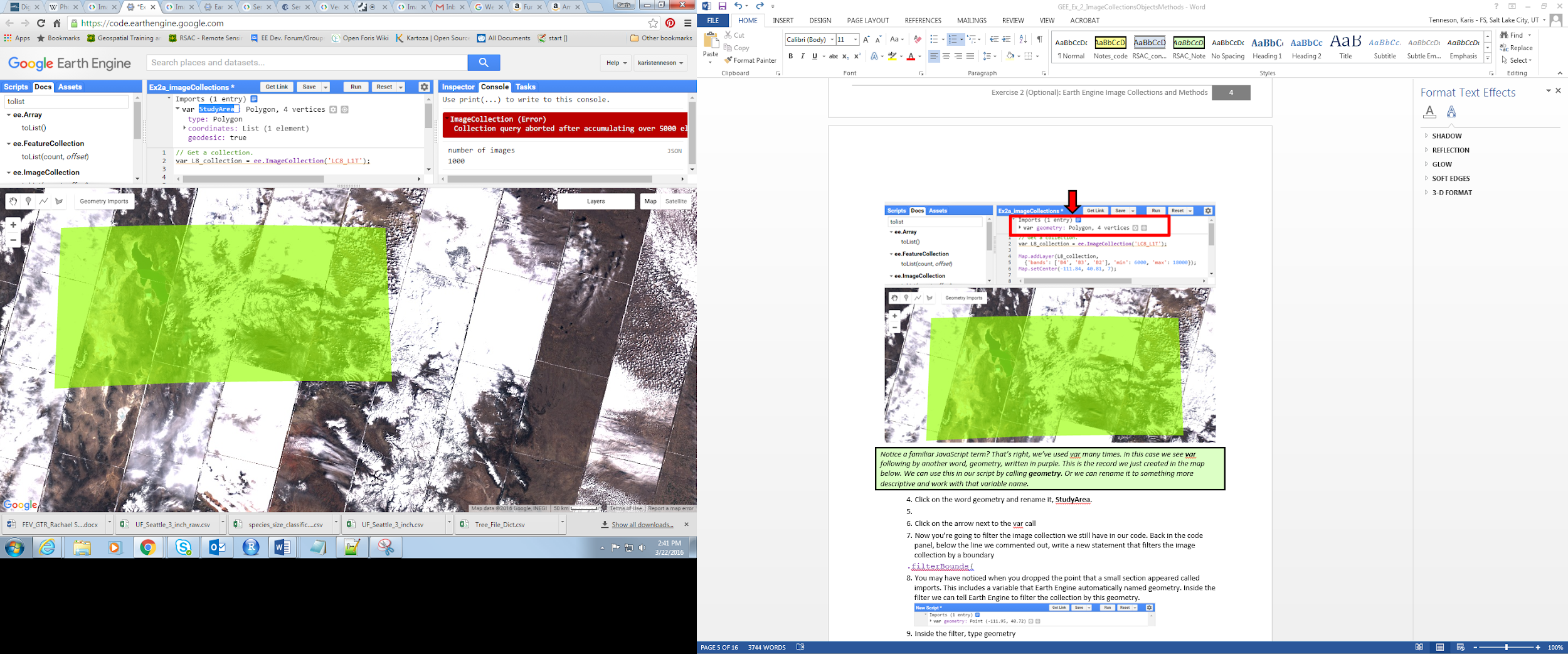


* + 1. Công cụ này cho phép bạn vẽ một đối tượng hình học thể hiện vùng nghiên cứu của mình. **Click** vào cửa sổ bản đồ để tạo một vùng/polygon xung quanh khu vực nghiên cứu của bạn (ví dụ, xung quanh Hà Nội hoặc nơi ở của bạn). Nhớ rằng, bạn có thể tắt/ mở ảnh Landsat để nhìn các lớp phía dưới. **Click đúp** để đóng kín vùng/polygon.
    2. Sau khi bạn có một vùng/polygon kín, sẽ xuất hiện một bản ghi trên code editor về vùng/polygon vừa được tạo ra. **Chú ý** mũi tên đỏ và phần khung màu đỏ trong hình dưới đây.



*Chú ý thấy một thuật ngữ JavaScript quen thuộc? Đúng vậy, chúng ta đã sử dụng biến* ***var*** *khá nhiều lần. Trong trường hợp này, chúng ta thấy* ***var*** *được theo sau bằng các từ khác, đối tượng hình học, được viết bằng màu tím. Đây chính là bản ghi (record) chúng ta vừa tạo ra trong trên cửa sổ bản đồ phía dưới. Chúng ta có thể sử dụng nó trong sript của mình bằng cách gọi* ***geometry****. Hoặc chúng ta có thể đặt lại tên cho phù hợp hơn để thuận tiện làm việc với tên biến đó*

* + 1. **Click** vào từ geometry. **Thay** geometry bằng **StudyArea.**



* + 1. **Click** vào biểu tượng đầu mũi tên cạnh dòng **var StudyArea** để xem thông tin của đối tượng hình học chúng ta vừa tạo ra (Xem hình trên)
       1. Bạn có thể làm hiện code vừa tạo bằng cách click vào box/hộp màu xanh lam cạnh dòng Import. Code này có thể đươc copy và paste vào script của bạn.
       2. Nếu bạn lướt/hover qua dòng **var StudyArea,** một biểu tượng thùng rác xuất hiện ở bên trái**.** Bạn có thể sử dụng để xóa bản ghi thao tác vừa rồi
  1. **Lọc tập ảnh bằng đối tượng hình học**

Bây giờ chúng ta đã sẵn sàng để lọc tập ảnh, L8\_collection. Trở lại với cửa sổ code, **copy** dòng dưới đây và **paste** vào code editor. **Hãy chắc chắn** bạn copy chúng xuống ngay dưới dòng lệnh tạo tập ảnh L8\_collection.

// Filter to scenes that intersect your boundary

**var** L8\_StudyArea **=** L8\_collection**.**filterBounds**(**StudyArea**);**

**Thay đổi** biến trongcâu lệnhMap.addlayer thành L8\_StudyArea (như dòng lệnh phía sau đây). Đưa nó xuống dưới dòng lệnh tạo biến L8\_StudyArea.

**Thay đổi** dòng lệnh Map.setCenter thành map.centerObject và cập nhật các tham số đầu vào. Câu lệnh này sẽ giúp dịch chuyển trung tâm của cửa sổ bản đồ về đối tượng hình học chúng ta vừa tạo (ví dụ phía dưới). Chuyển nó xuống dưới dòng lệnh tạo biến L8\_StudyArea.

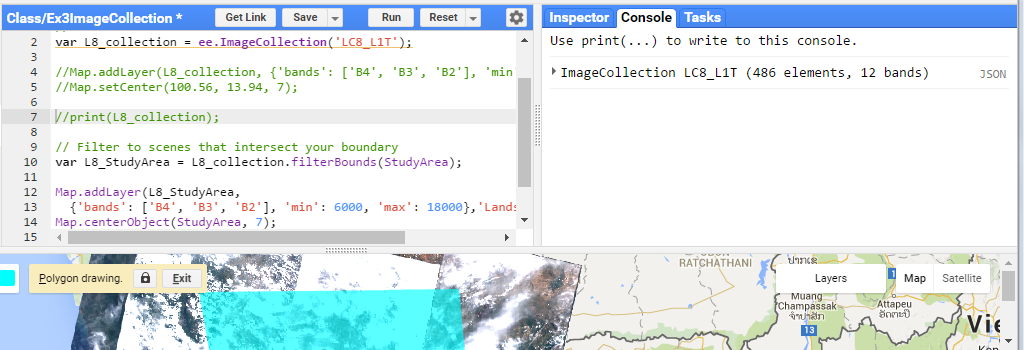
Map**.**addLayer**(**L8\_StudyArea**,**

**{** min:0.05, max: 0.8, bands: 'B6, B5, B4'**},**'Landsat 8 in study region'**);**

Map**.**centerObject**(**StudyArea**,** 7**);**

* + 1. **Click Run.** Bây giờ tập ảnh sẽ lọc vả chỉ chọn những ảnh giao với vùng/polygon mà bạn đã tạo.
    2. *((Không bắt buộc)* **Chỉnh sửa**lệnh print để in chi tiết của tập ảnh mới*,* L8\_StudyArea (đảm bảo để lệnh này ngay dưới lệnh tạo biến the L8\_StudyArea). Khi lệnh hoàn thành, kết quả có 486 thành phần/element trong tập ảnh của tôi (Số lượng ảnh của bạn có thể khác phụ thuộc vào kích thước của đối tượng hình học mà bạn đã số hóa)

print**(**L8\_StudyArea**);**



* + 1. Chúng ta cũng có thể sử dụng phương pháp định kích thước của tập ảnh để xác định bao nhiêu ảnh sẽ có trong tập ảnh của bạn. **Copy** dòng lệnh sau và **paste** chúng xuỗng dưới script của bạn và. **Run** script

// Get the number of images.

**var** count **=** L8\_StudyArea**.**size**();**

print**(**'Count of L8\_StudyArea: '**,** count**);**

* + 1. (Không bắt buộc, **tìm hiểu** các dạng dữ liệu và hàm khác để có cùng câu tr*ả* lời với phương pháp đếm/count tập ảnh): Có cách khác để có được số lượng ảnh, mặc dù phức tạp hơn một chút. Chúng ta có thể sử dụng hàm **length()** trong lists/arrays. Trước tiên, chúng ta cần chuyển tập ảnh thành một List/Danh sách. **Copy** dòng lệnh sau và **paste** chúng dưới sript của bạn. **Run** sript.
       1. **Đọc trên** Console, hàm nào tốn nhiều thời gian để thực hiện hơn

// Convert the collection to a list and get the number of images.

**var** listL8\_SA **=** L8\_StudyArea**.**toList**(**1500**);**

print**(**"number of images"**,** listL8\_SA**.**length**());**

Phần 3: Lọc Tập Ảnh Theo Thời Gian

* 1. **Tạo tập ảnh với giới hạn theo ngày**
     1. Tiếp theo **thêm** một dòng lệnh giúp lọc ảnh theo thời gian, sử dụng hàm **filterDate()** (xem ví dụ dưới đây). **filterDate()** cho phép bạn định rõ ngày bắt đầu và kết thúc như các tham số đầu vào để giảm kích thước của tập ảnh và đạt được mục tiêu dự án của mình. Dòng code mới đưa phép lọc vào tập dữ liệu, L8\_StudyArea, dựa vào ngày mà các ảnh được thu nhận
     2. Sau đó, **thêm** lệnh count/đếm và print/in để xem có bao nhiêu ảnh trong tập ảnh mới
     3. Chúng ta đã có script hoàn chỉnh phía dưới**. Chỉnh sửa** sript của bạn cho đúng và Click **Run**

// Get a collection.

**var** L8\_collection **=** ee**.**ImageCollection**(**'LANDSAT/LC8\_L1T\_TOA'**);**

// Filter to scenes that intersect your boundary

**var** L8\_StudyArea **=** L8\_collection**.**filterBounds**(**StudyArea**);**

// Filter to scenes for a given time period

**var** L8\_SA\_2015 **=** L8\_StudyArea**.**filterDate**(**'2015-01-01'**,** '2015-12-31'**);**

Map**.**addLayer**(**L8\_SA\_2015**,**

**{** min:0.05, max: 0.8, bands: 'B6, B5, B4' **});**

Map**.**centerObject**(**StudyArea**,** 7**);**

// Get the number of images.

**var** count **=** L8\_StudyArea**.**size**();**

print**(**'Count of L8\_StudyArea: '**,** count**);**

// Get the number of images.

**var** count15 **=** L8\_SA\_2015**.**size**();**

print**(**'Count of L8\_SA-2015: '**,** count15**);**

* + 1. *(Không bắt buộc)* Dưới đây là một số lệnh bổ sung giúp bạn có thể **tìm hiểu** tập ảnh của mình. Các lệnh này nằm trong phần hướng dẫn sử dụng

<https://developers.google.com/earth-engine/ic_info>

// Get statistics for a property of the images in the collection.

**var** sunStats **=** L8\_StudyArea**.**aggregate\_stats**(**'SUN\_ELEVATION'**);**

print**(**'Sun elevation statistics: '**,** sunStats**);**

// Sort by a cloud cover property, get the least cloudy image.

**var** **LoCloudimage** **=** ee**.**Image**(**L8\_StudyArea**.**sort**(**'CLOUD\_COVER'**).**first**());**

print**(**'Least cloudy image: '**,** **LoCloudimage);**

// Limit the collection to the 10 most recent images.

**var** recent **=** L8\_StudyArea**.**sort**(**'system:time\_start'**,** **false).**limit**(**10**);**

print**(**'Recent images: '**,** recent**);**

Phần 4: Tìm hiểu các hàm xử lý ảnh

Một tập hợp đầy đủ các công cụ sẵn có trong Code Editor để phân tích và xử lý tập ảnh.

Phần 5: Giới Thiệu về Reducers

* 1. **Phép thu nhỏ tập ảnh, giá trị median/trung bình của pixel**

Phần này tập trung vào một hàm dạng khác khi xử lý dữ liệu trong 1 tập ảnh, reducer/phép thu nhỏ. Các phép thu nhỏ/Reducer để giảm kích thước của tập ảnh bằng các phép toán thống kê đơn giản, ví dụ giá trị trung bình của pixel. Kết quả đầu ra sẽ là một đối tượng ảnh (một lớp raster) thể hiện được một số đặc tính của toàn bộ tập ảnh. Để biết thêm về reducer, đọc hướng dẫn sử dụng tại đây:

<https://developers.google.com/earth-engine/reducers_image_collection>.

* + 1. Trước tiên để đơn giản hóa sript/tập lệnh mà bạn đang làm việc. **sửa/Modify** script trong Code Editor cho giống với hình dưới đây (hoặc bắt đầu với cửa số script mới (empty) và copy toàn độ dòng lệnh dưới vào code editor- nhưng nhớ vẽ lại vùng nghiên cứu của mình). Sau đó **Run script/tập lệnh**

// Get a collection.

**var** L8\_collection **=** ee**.**ImageCollection**(**'LANDSAT/LC8\_L1T\_TOA'**);**

// Filter to scenes that intersect your boundary

**var** L8\_StudyArea **=** L8\_collection**.**filterBounds**(**StudyArea**);**

// Filter to scenes for a given time period

**var** L8\_SA\_2015 **=** L8\_StudyArea**.**filterDate**(**'2015-01-01'**,** '2015-12-31'**);**

print(L8\_SA\_2015, 'L8\_SA\_2015');

* + 1. Bây giờ thêm một số lệnh để tạo hoặc hiển thị một tổ hợp ảnh giá trị trung bình / median composite, sử dụng hàm thu nhỏ tập ảnh **median()**. Lệnh sẽ giúp tạo 1 layer ảnh **Image** (một ảnh đơn) biểu diễn giá trị median của tất cả các ảnh trong tập ảnh đã được lọc.

//reduce to median value per pixel

**var** median\_L8\_2015 **=** L8\_SA\_2015**.**median**();**

print(median\_L8\_2015, 'median\_L8\_2015');

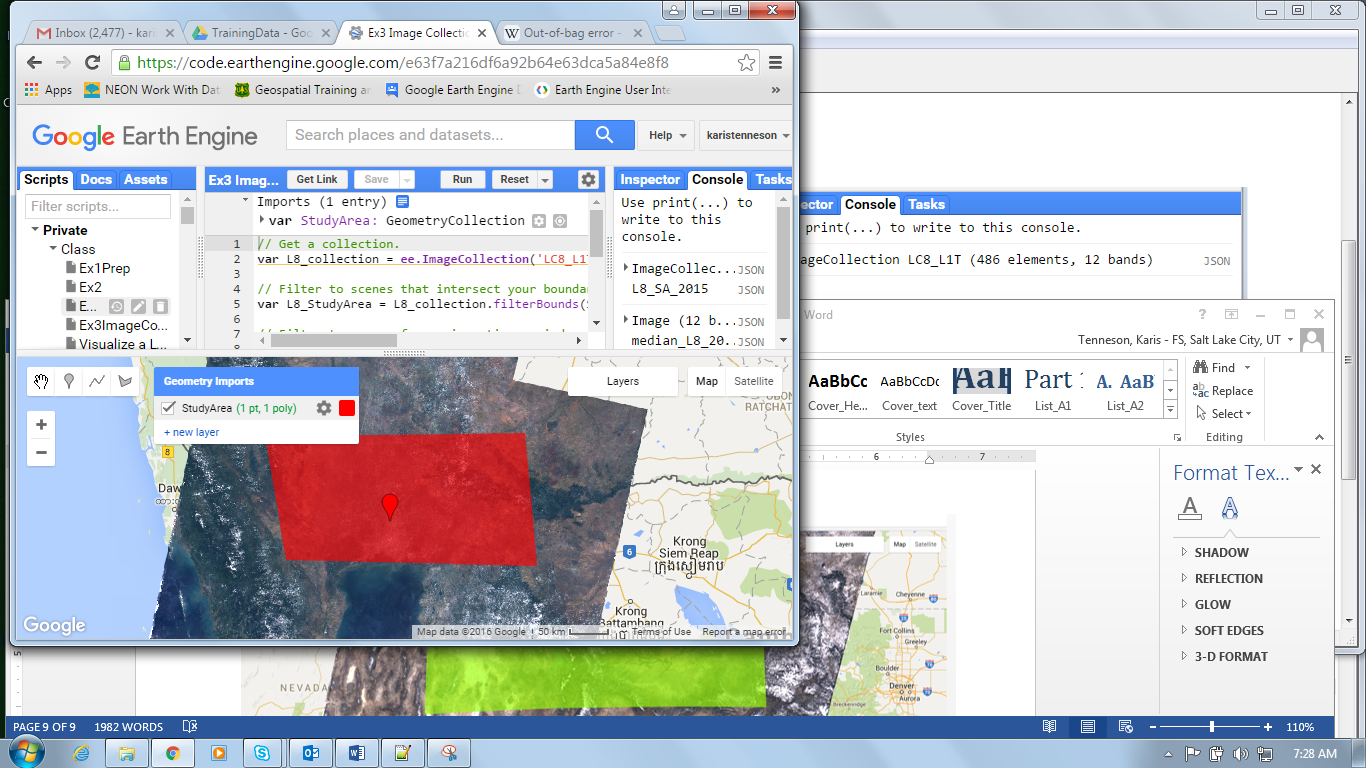
//add layers

Map**.**addLayer**(**median\_L8\_2015**,**

**{** min:0.05, max: 0.8, bands: 'B6, B5, B4'**});**

Map**.**centerObject**(**StudyArea**,** 7**);**

* + 1. Kiểm tra kết quả, **chú ý** những điểm sau**:** :
       1. **Nhìn** vào kết quả sau khi thực hiện lệnh Print/in. Bạn thấy có sự khác biệt gì giữa L8\_SA\_2015 và median\_L8\_2015? Gợi ý: một là tập hợp ảnh, còn một là đối tượng ảnh (một raster layer duy nhất)
       2. Bạn có thể **cho hiển thị** vùng nghiên cứu của mình bằng cách chọn vào ô/box vuông nhỏ trong thanh công cụ Geometry Imports phía trên bên trái của cửa sổ bản đồ (như hình sau)
       3. Tổ hợp ảnh chỉ bao gồm các hàng cột các cảnh ảnh Landsat giao với vùng nghiên cứu của bạn (phóng to để xem toàn bộ)
       4. Ảnh cũng tương đối ít mây do chúng ta đang xem giá trị trung tuyến /median pixel của toàn bài tập ảnh. Bạn còn thấy mây không?



Kể cả khi sử dụng phép thu nhỏ tập ảnh với median/median reducer, vẫn có thể còn một số pixel trông giống như mây. Tốt nhất cần loại bỏ mây ở mỗi ảnh trong toàn bộ tập ảnh trước khi chạy median reducer. Để được như vậy, chúng ta cần học thêm về các hàm để làm việc với toàn bộ tập ảnh

* + 1. Lưu sript/tập lệnh, đặt tên là ‘Ex 3 Image Collection Part 5’. Bạn sẽ quay lại bài tập này sau

Phần 6: Nhập Môn Về các Hàm

Khi bạn phát triển code, các câu lệnh có thể bắt đầu trở nên khá dài. Để code được sắp xếp trật tự và để tạo thêm nhiều code có hiệu quả cao khi sử dụng, bạn có thể tận dụng lại các code mình đã thiết lập từ trước. Bạn hãy chia nhỏ code thành các phần, tách riêng các phần này, đặt tên để có thể sử dụng khi bạn cầu sau này. Các hàm là là phần nhỏ hơn của code. Ví dụ, chúng ta có một hàm để lấy dữ liệu mà mình quan tâm, một hàm để phân tích tích dữ liệu, và một hàm để xuất dữ liệu. Chúng ta cần chia nhỏ code thành 3 phần- mỗi phần tương ứng với một hàm.

Các hàm cũng rất hữu dụng khi chúng ta có một loạt lệnh mà bạn muốn sử dụng lại nhiều lần trong tập code. Ví dụ, có thể ta muốn tính mean của dữ liệu hoặc NDVI của tập raster, chúng ta có thể tạo một hàm mà có thể sử dụng/gọi lại, hơn là phải viết mới từ đầu

* 1. **Cấu trúc**

Đây là cách các hàm được thiết lập :

1. **Code/Mã Lệnh:** Bạn lấy code mình muốn sử dụng (một dòng lệnh hoặc hàng trăm dòng lệnh), đưa chúng vào trong dấu ngoặc móc {} để biết được hàm bắt đầu và kết thúc ở đâu. Trong ví dụ dưới đây, các dòng chú giải màu xanh lá cây đang nằm trong khối mã lệnh/code block.
2. ***function*:** Bắt đầu bằng từ **function** (ngôn ngữ Javascript), in thường,trước dấu ngoặc cong đầu tiên
3. **name\_of\_function (tên hàm):** sau **function** là tên mà chúng ta muốn đặt cho hàm/function đó. Chúng ta có thể đặt tên hàm/function theo ý mình, quy luật và gợi ý đặt tên giống như khi đặt tên biến (không bắt đầu bằng chữ số, thể hiện đúng chức năng của hàm…)
4. **Thông số đầu vào (input):** Giữa tên hàm và ngoặc móc {} đầu tiên, chúng ta thêm một dấu ngoặc đơn (). Trong phần dấu ngoặc đơn, chúng ta sẽ điền các thông số đầu vào (input) mà người sử dùng muốn cung cấp cho hàm/function, nếu không có, chúng có thể được để trống. Thông số đầu vào là một phần thông tin (Nó có thể được chứa trong một biến) mà được đưa vào hàm khi được gọi. Nếu bạn có nhiều thông số muốn đưa vào hàm, đơn giản ngăn cách chúng bằng dấu phẩy. Ví dụ khi tạo ra hàm tính NDVI của ảnh, chúng ta sẽ đưa vào hàm một ảnh mà chúng ta muốn tính NDVI.
5. **Gọi hàm:** một khi bạn đã tạo một hàm, nó sẽ không được thực hiện cho tới khi bạn gọi hàm đó. Để sử dụng hàm trong code của mình, bạn viết tên hàm, tiếp theo là dấu ngoặc đơn, sau đó là điền thông số vào bên trong ngoặc đơn, kết thúc bằng dấu chấm phẩy.

**function** name\_of\_function **(**parameter\_1, parameter\_2**)** **{**

// code to execute

// more code to execute

// …

**}**

// call the function

name\_of\_function**(**input\_1, input\_2**);**

Đây là cấu trúc cơ bản của một hàm. Nếu bạn thấy chúng có vẻ hơi phức tạp, đừng lo lắng, bạn sẽ thấy dễ dàng hơn khi tự viết chúng. Sau đây, bạn sẽ thực tập để tạo một hàm đơn giản **calculate\_sum**

***Ghi chú****: Quy tắc chung là khai báo và định rõ tất cả các hàm trước, sau đó gọi chúng ra trong script. Điều này là không bắt buộc nhưng sẽ code dễ đọc hơn*

* 1. **Tạo hàm để tính tổng số**
     1. **Mở** một cửa sổ làm việc mới bằng cách click/nhấn vào biểu tượng đầu mũi tên cạnh Reset. Sau đó chọn **Clear Script.**
     2. **Xem kỹ** phần code bên dưới. Đây là ví dụ về hàm giúp tính tổng hai giá trị

**function** calculate\_Sum **(**in\_value1**,** in\_value2**)** **{**

**var** Sum **=** ee**.**Number**(**in\_value1**).**add**(**ee**.**Number**(**in\_value2**));** // code that calculates the sum

print**(**Sum**);**

**}**

// call the function

calculate\_Sum**(**75**,** 82**);**

***Ghi chú****: Cho tới nay, chúng ta đã thực hiện chạy các code theo thứ tự. Các hàm hơi khác nhau ở điểm chúng có thể được định nghĩa ở bất cứ đâu trong chuỗi code, nhưng chỉ thực hiện chạy khi chúng được gọi. Ví dụ dưới đây cũng giống với ví dụ ở trên. Rất nhiều lập trình viên sẽ nhóm tất cả các hàm của họ lại ngay khi bắt đầu một script, tuy nhiên hàm đó chỉ chạy khi được gọi về sau.*

* + 1. Ngay trong hàm vừa tạo, **gõ**/**type** hai giá trị khác vào calculate\_Sum () để tính tổng và chạy code. Dòng lệnh ví dụ như sau:

calculate\_Sum**(**99**,** 106**);**

// copy the call statement (above), below the creation of the function

**function** calculate\_Sum **(**in\_value1**,** in\_value2**)** **{**

**var** Sum **=** ee**.**Number**(**in\_value1**).**add**(**ee**.**Number**(**in\_value2**));** // code that calculates the sum

print**(**Sum**);**

**}**

***Ghi chú****: Nếu bạn thấy quen với lệnh gọi các hàm này, rất tốt! Chúng ta đã dùng các hàm JavaScript trong bài tập này rồi. Một ví dụ là print()*

* + 1. Một khi hàm đã được tạo, bạn có thể gọi hàm bao nhiêu lần bạn muốn. Hãy gọi gàm này một lần nữa với các tham số mới. Lần này tự **thêm** hai giá trị bạn muốn

calculate\_Sum**(**790**,** 1.555**);**

calculate\_Sum**(**133**,** 765**);**

* 1. **Biến toàn cục và biến cục bộ**

Làm thế nào khi chúng ta muốn lưu kết quả tính tổng vừa thực hiện như một biến để sử dụng trong mã code sau này, chúng ta có thể thực hiện được không? Chúng ta tạo và định nghĩa một biến gọi là Sum trong hàm. **Thử** in biến này sau khi gọi hàm để kiểm tra giá trị kết quả. **Gõ** dòng lệnh tạo hàm sau và chạy script/tập lệnh.

// call the function

calculate\_Sum**(**75**,** 82**);**

print**(**Sum**);**

* + - 1. Bạn nhận được gì? Bạn sẽ nhận được báo lỗi trên console “Sum is not defined in the global scope”, có nghĩa là Sum không được định nghĩa trong phạm vi toàn cục.
    1. Câu lệnh trên không thực hiện được. Thử lưu nó dưới một định dạng khác. Hãy xóa dòng code bạn vừa thực hiện ở bước vừa rồi. Xem kỹ dòng code dưới đây. Đưa vào script và Chạy/Run code

//call the function and save it as a variable called Sum\_test

**var** Sum\_test **=** calculate\_Sum**(**75**,** 82**);**

// Try to call the variable we created in the function,

// outside of the function call

print**(**Sum\_test**);**

* + - 1. Chuyện gì xảy ra? Chúng ta không còn thấy báo lỗi nữa, tuy nhiên lưu nó dưới dạng một biến lại không thực hiện được. Tôi nhận được một thông báo rằng “a property cannot be read” - Thuộc tính không được xác định
    1. Cho tới bây giờ, chúng ta vẫn chưa thành công để truy nhập vào kết quả của hàm. Đó là bởi vì biến **Sum** là một biến cục bộ, một biến chỉ tồn tại và khi được nằm trong hàm của nó. Trong JavaScript, bất kỳ **var** nào được dùng trong một hàm, biến kết quả là một biến cục bộ chỉ có thể truy cập được trong hàm đó.

***Thế nào là biến cục bộ? Thế nào là biến toàn cục?***

*Biến cục bộ là một biến được khai báo trong một hàm. Không giống như biến toàn cục, biến cục bộ chỉ có thể sử dụng dưới hàm mà nó được khai báo.*

*Các biến được khai báo bên ngoài hàm được gọi là biến toàn cục và có thể đọc được ở bất kì vị trí nào trong script, kể cả trong hay ngoài các hàm. Cho tới nay, tất cả những gì chúng ta được học là về biến toàn cục, mặc dù chúng ta vẫn chưa gọi chúng là biến toàn cục. Một biến toàn cục là biến mà được khai báo bên ngoài của bất kỳ hàm nào. Các biến này có thể sử dụng bất kỳ lúc nào trong code và có thể sử dụng cả bên trong các hàm.*

* 1. **Sử dụng “return” để lưu kết quả của hàm dưới dạng một biến**

Làm thế nào để chúng ta có thể chuyển kết quả cục bộ của một hàm thành một biến toàn cục? JavaScript sử dụng câu lệnh “return” để chuyển một giá trị cục bộ về chương trình chính. Bằng cách khai báo một biến toàn cục và định nó tương đương với một phép gọi hàm, chúng ta gán kết quả của hàm cho một biến mà có thể sử dụng ở bất cứ đâu trong script

* + 1. **Đọc kỹ** những dòng lệnh sau. **Đưa** các dòng lệnh này vào code editor script của bạn và **click   
       Run**

**function** calculate\_Sum **(**in\_value1**,** in\_value2**)** **{**

**var** Sum **=** ee**.**Number**(**in\_value1**).**add**(**ee**.**Number**(**in\_value2**));** // code that calculates the sum

print**(**"Local variable, Sum, = "**,** Sum**);**

**return** Sum**;** // here we use return

**}**

// Now call and save the output of the function.

// Save it as a variable, Sum\_test.

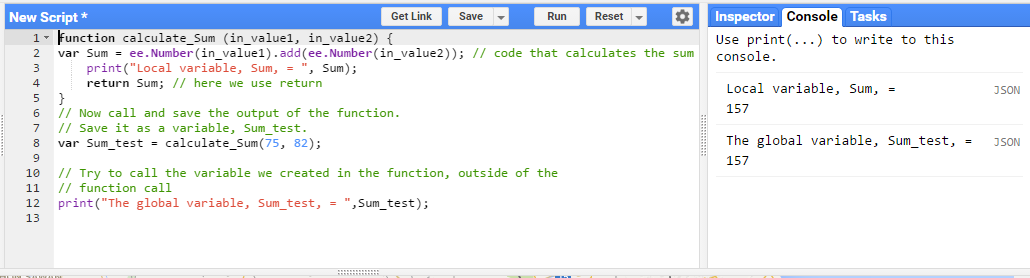
**var** Sum\_test **=** calculate\_Sum**(**75**,** 82**);**

// Try to call the variable we created in the function, outside of the

// function call

print**(**"The global variable, Sum\_test, = "**,**Sum\_test**);**

* + 1. Bạn thấy gì xảy ra? Bây giờ, dòng lệnh đã thực hiện thành công! Tôi thấy giá trị 157 được in ra trong cửa dổ Console của mình 2 lần:
       1. Lần một là một biến cục bộ sau khi kết quả được in ra từ hàm
       2. Lần thứ hai là sau khi gọi hàm print biến toàn bộ mà tôi vừa tạo, **Sum\_test.**



Phần 7: Tạo hàm lọc bỏ mây (cloud mask)

Nhớ lại bài tập 2, bạn đã học về mặt nạ lọc mây (cloud mask). Bây giờ bạn sẽ viết lại thành hàm mà có thể gọi ra bất cứ lúc nào trong script của mình.

* 1. **Đưa vào câu lệnh lọc bỏ mây (cloud mask) từ bài tập 2**
     1. Copy và paste các dòng lệnh sau vào một cửa sổ Code Editor trống. Trông chúng khá quen vì bạn đã làm việc với chúng từ bài tập 2. Chạy script và đảm bảo nó được thực hiện.

// Get an image and display in map window.

**var** LC8\_image **=** ee**.**Image**(**'LANDSAT/LC8\_L1T\_TOA/LC81290502013110LGN01'**);**

//Specify the cloud likelihood threshold -

**var** cloud\_thresh **=** 40**;**

//use add the cloud likelihood band to the image

**var** CloudScore **=** ee**.**Algorithms**.**Landsat**.**simpleCloudScore**(**LC8\_image**);**

//isolate the cloud likelihood band

**var** quality **=** CloudScore**.select(**'cloud'**);**

//get pixels above the threshold

**var** cloud01 **=** quality**.**gt**(**cloud\_thresh**);**

//create a mask from high likelihood pixels

**var** cloudmask **=** LC8\_image**.**mask**().**and**(**cloud01**.**not**());**

//mask those pixels from the image

**var** LC8\_imageNoClouds **=** LC8\_image**.**mask**(**cloudmask**);**

//Review the result

addToMap**(**LC8\_imageNoClouds**,** **{**bands:**[**'B6'**,**'B5'**,**'B4'**],**min:0.1**,** max:0.5**},**'Landsat8scene\_cloudmasked'**);**

* + 1. Bây giờ chỉnh sửa lại để chuyển nó một thành hàm. Đầu tiên, phía trước “use add the cloud likelihood band to the image”, thêm một dòng như sau

**var** cloudfunction **=** **function(){**

* + 1. Phía trước dòng ghi chú ‘Review the result’, chỉ thêm duy nhất một dấu ngoặc móc }. Phần code của bạn sẽ trông thế này:

// Get an image and display in map window.

**var** LC8\_image **=** ee**.**Image**(**'LANDSAT/LC8\_L1T\_TOA/LC81290502013110LGN01'**);**

//Specify the cloud likelihood threshold -

**var** cloud\_thresh **=** 40**;**

**var** cloudfunction **=** **function(){**

//use add the cloud likelihood band to the image

**var** CloudScore **=** ee**.**Algorithms**.**Landsat**.**simpleCloudScore**(**LC8\_image**);**

//isolate the cloud likelihood band

**var** quality **=** CloudScore**.select(**'cloud'**);**

//get pixels above the threshold

**var** cloud01 **=** quality**.**gt**(**cloud\_thresh**);**

//create a mask from high likelihood pixels

**var** cloudmask **=** LC8\_image**.**mask**().**and**(**cloud01**.**not**());**

//mask those pixels from the image

**var** LC8\_imageNoClouds **=** LC8\_image**.**mask**(**cloudmask**);**

**}**

//Review the result

addToMap**(**LC8\_imageNoClouds**,** **{**bands:**[**'B6'**,**'B5'**,**'B4'**],**min:0.1**,** max:0.5**},**'Landsat8scene\_cloudmasked'**);**

* + 1. Tiếp theo bạn cần định rõ tham số đầu vào. Trong ngoặc đơn ở dòng thứ 7 (dòng khai báo hàm: var cloudfunction = function()), thêm từ *image*
    2. Sau đó thay biến *LC8\_image* bằng *image*. Như vậy là có 3 vị trí *LC8\_image* cần được thay thế, xem ví dụ code dưới đây

// Get an image and display in map window.

**var** LC8\_image **=** ee**.**Image**(**'LANDSAT/LC8\_L1T\_TOA/LC81290502013110LGN01'**);**

//Specify the cloud likelihood threshold -

**var** cloud\_thresh **=** 40**;**

**var** cloudfunction **=** **function(image){**

//use add the cloud likelihood band to the image

**var** CloudScore **=** ee**.**Algorithms**.**Landsat**.**simpleCloudScore**(image);**

//isolate the cloud likelihood band

**var** quality **=** CloudScore**.select(**'cloud'**);**

//get pixels above the threshold

**var** cloud01 **=** quality**.**gt**(**cloud\_thresh**);**

//create a mask from high likelihood pixels

**var** cloudmask **=** image**.**mask**().**and**(**cloud01**.**not**());**

//mask those pixels from the image

**var** LC8\_imageNoClouds **=** image**.**mask**(**cloudmask**);**

**}**

//Review the result

addToMap**(**LC8\_imageNoClouds**,** **{**bands:**[**'B6'**,**'B5'**,**'B4'**],**min:0.1**,** max:0.5**},**'Landsat8scene\_cloudmasked'**);**

* + 1. Cuối cùng, chúng ta cần thêm một lệnh return để kết quả, LC8\_imageNoClouds là một biến toàn cục mà chúng ta có thể sử dụng bên ngoài hàm. Thay dòng code dưới đây

//mask those pixels from the image

**var** LC8\_imageNoClouds **=** LC8\_image**.**mask**(**cloudmask**);**

Bằng lệnh return như sau:

**return** LC8\_image**.**mask**(**cloudmask**);**

* + 1. Bây giờ chúng ra đã sẵn sàng để gọi hàm. Sau khi đóng ngoặc móc } tại vị trí kết thúc cuối hàm, tạo một biến mới tên là LC8\_imageNoClouds. Sau đó chạy hàm trên ảnh chúng ta vừa lưu là LC8\_image. Xem ví dụ dòng lệnh sau:

**var** LC8\_imageNoClouds **=** cloudfunction**(**LC8\_image**);**

* + 1. Click Run để xem hàm của chúng ta có làm việc không. Tham khảo toàn bộ script dưới đây để đảm bảo những gì bạn viết là đúng

// Get an image and display in map window.

**var** LC8\_image **=** ee**.**Image**(**'LANDSAT/LC8\_L1T\_TOA/LC81290502013110LGN01'**);**

//Specify the cloud likelihood threshold -

**var** cloud\_thresh **=** 40**;**

**var** cloudfunction **=** **function(image){**

//use add the cloud likelihood band to the image

**var** CloudScore **=** ee**.**Algorithms**.**Landsat**.**simpleCloudScore**(image);**

//isolate the cloud likelihood band

**var** quality **=** CloudScore**.select(**'cloud'**);**

//get pixels above the threshold

**var** cloud01 **=** quality**.**gt**(**cloud\_thresh**);**

//create a mask from high likelihood pixels

**var** cloudmask **=** image**.**mask**().**and**(**cloud01**.**not**());**

//mask those pixels from the image

**return** image**.**mask**(**cloudmask**);**

**}**

**var** LC8\_imageNoClouds **=** cloudfunction**(**LC8\_image**);**

//Review the result

addToMap**(**LC8\_imageNoClouds**,** **{**bands:**[**'B6'**,**'B5'**,**'B4'**],**min:0.1**,** max:0.5**},**'Landsat8scene\_cloudmasked'**);**

* + 1. Lưu scrip của bạn với tên gọi “Ex3\_CloudMaskFunction”.

Phần 8: Các Hàm sử dụng Với Tập Hợp Ảnh

Bây giờ chúng ta biết các hàm của mình đã hoạt động, cùng áp dụng chúng cho tập ảnh mà bạn đã tạo trong nửa đầu bài tập này

Copy hàm lọc mây (cloud mask) như dưới đây.

**var** cloudfunction **=** **function(image){**

//use add the cloud likelihood band to the image

**var** CloudScore **=** ee**.**Algorithms**.**Landsat**.**simpleCloudScore**(image);**

//isolate the cloud likelihood band

**var** quality **=** CloudScore**.select(**'cloud'**);**

//get pixels above the threshold

**var** cloud01 **=** quality**.**gt**(**cloud\_thresh**);**

//create a mask from high likelihood pixels

**var** cloudmask **=** image**.**mask**().**and**(**cloud01**.**not**());**

//mask those pixels from the image

**return** image**.**mask**(**cloudmask**);**

**}**

Sau đó mở script đã được đặt tên là ‘Ex 3 Image Collection Part 5’.

Paste hàm lọc mây và biến cloud\_thresh vào script (sau những dòng lệnh tạo tập hợp ảnh lọc mây, nhưng phía trước median reducer). Xem code hoàn chỉnh phía dưới đây

// Get a collection.

**var** L8\_collection **=** ee**.**ImageCollection**(**'LANDSAT/LC8\_L1T\_TOA'**);**

// Filter to scenes that intersect your boundary

**var** L8\_StudyArea **=** L8\_collection**.**filterBounds**(**StudyArea**);**

// Filter to scenes for a given time period

**var** L8\_SA\_2015 **=** L8\_StudyArea**.**filterDate**(**'2015-01-01'**,** '2015-12-31'**);**

print**(**L8\_SA\_2015**,** 'L8\_SA\_2015'**);**

**var** cloud\_thresh **=** 40;

**var** cloudfunction **=** **function(image){**

//use add the cloud likelihood band to the image

**var** CloudScore **=** ee**.**Algorithms**.**Landsat**.**simpleCloudScore**(image);**

//isolate the cloud likelihood band

**var** quality **=** CloudScore**.select(**'cloud'**);**

//get pixels above the threshold

**var** cloud01 **=** quality**.**gt**(**cloud\_thresh**);**

//create a mask from high likelihood pixels

**var** cloudmask **=** image**.**mask**().**and**(**cloud01**.**not**());**

//mask those pixels from the image

**return** image**.**mask**(**cloudmask**);**

**};**

//reduce to median value per pixel

**var** median\_L8\_2015 **=** L8\_SA\_2015**.**median**();**

print**(**median\_L8\_2015**,** 'median\_L8\_2015'**);**

//add layers

Map**.**addLayer**(**median\_L8\_2015**,**

**{** min:0.05, max: 0.8, bands: 'B6, B5, B4'**},** 'median\_L8\_2015 with clouds'**);**

Map**.**centerObject**(**StudyArea**,** 7**);**

* + 1. Bây giờ thêm một lệnh giúp đưa kết quả thực hiện với toàn bộ tập ảnh lên cửa sổ xuất kết quả bản đồ. Sau đó đưa tập ảnh với mây được lọc lên cửa sổ hiển thị. Xem ví dụ về lệnh map dưới đây

**var** L8\_SA\_2015NoClouds **=** L8\_SA\_2015**.**map**(**cloudfunction**);**

Map**.**addLayer**(**ee**.**Image**(**L8\_SA\_2015NoClouds**.**first**()),**

**{**min**:**0.05**,** max**:** 0.8**,** bands**:** 'B6, B5, B4'**},** 'first image no clouds'**);**

Map**.**centerObject**(**StudyArea**,** 7**);**

* + 1. Thay đổi lệnh median reducer để chạy trên tập ảnh L8\_SA\_2015NoClouds image mà bạn vừa tạo. Xem ví dụ các code dưới đây.

//reduce to median value per pixel

**var** median\_L8\_2015 **=** L8\_SA\_2015NoClouds**.**median**();**

print**(**median\_L8\_2015**,** 'median\_L8\_2015'**);**

//add layers

Map**.**addLayer**(**median\_L8\_2015**,**

**{**min**:**0.05**,** max**:** 0.8**,** bands**:** 'B6, B5, B4'**},** 'median composite of cloud free images'**);**