



Phần 2: Giới thiệu các khái niệm lập trình

Đỗ Thành Long
dtlong@opengis.vn

OPENGIS
Discover the world, Learn with maps

 <https://opengis.vn>

Nội dung

- ✓ Một số ngôn ngữ lập trình phát triển phần mềm GIS phổ biến
- ✓ Biến, kiểu dữ liệu và toán tử
- ✓ Cấu trúc điều khiển (if-else, vòng lặp)
- ✓ Hàm (function)



1. Ngôn ngữ lập trình GIS nguồn mở



```
1 // Calculate solar potential
2
3 If($feature.SUM_ELECTR12 < 3000){
4   return "Low"
5 }
6 if($feature.SUM_ELECTR12 >= 5000){
7   return "High"
8 }
9 if(3000 <=
10  return
11 }
12
```



1. Ngôn ngữ lập trình GIS nguồn mở

GIS
programming
language

Python

Phổ biến nhất, mạnh mẽ trong lĩnh vực GIS

Nhiều **thư viện** GIS hỗ trợ: GDAL, Fiona, GeoPandas và ArcPy,..

JavaScript

Bắt buộc khi phát triển WebGIS

Các thư viện GIS như: Leaflet, OpenLayers, Mapbox, ArcGIS Maps SDK for JavaScript,..

1. Ngôn ngữ lập trình GIS nguồn mở

GIS
programming
language

R

Mạnh về thống kê

Java

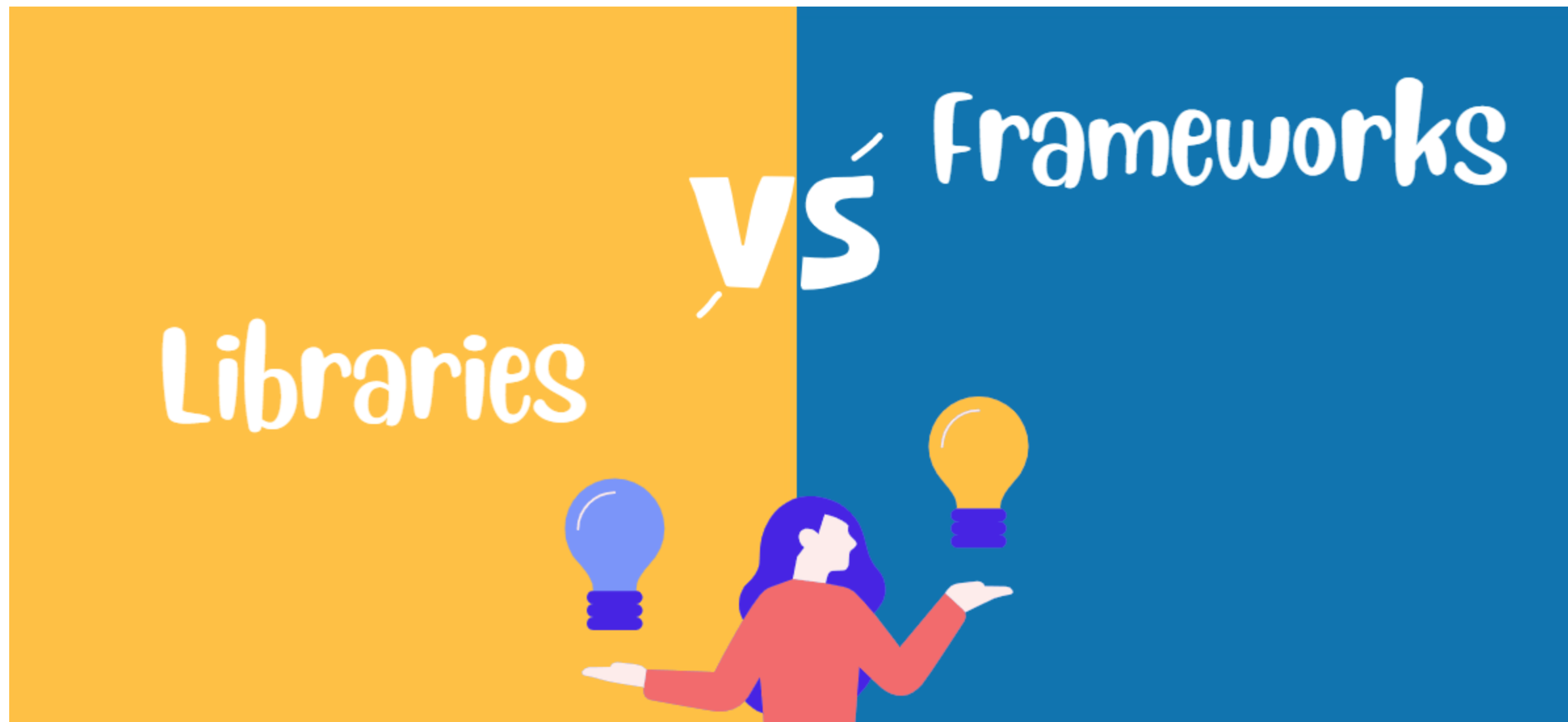
Sử dụng phổ biến trong nhiều ứng dụng GIS: GeoServer, MapStores, GeoNode, ..

C, C++, C#

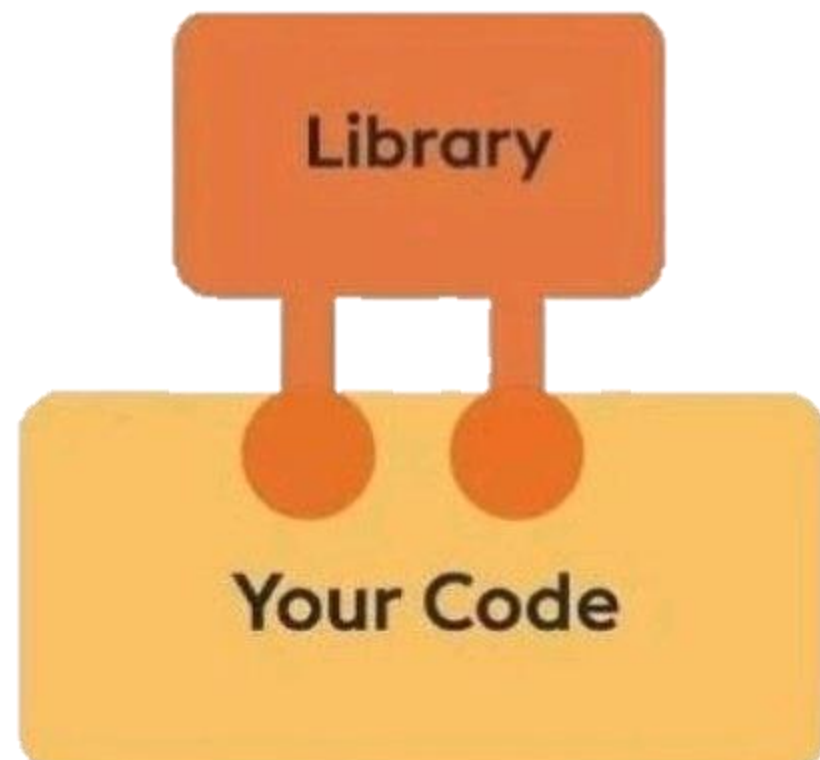
Nhanh và mạnh mẽ, phát triển các thư viện và ứng dụng GIS có hiệu suất cao



1. Ngôn ngữ lập trình GIS nguồn mở



1. Ngôn ngữ lập trình GIS nguồn mở



Library

Thư viện lập trình

- Tập hợp các mã nguồn đã được viết sẵn
- Thực hiện các tác vụ cụ thể trong ứng dụng
- Gọi các hàm hoặc lớp đã được định nghĩa trong thư viện để thực hiện các chức năng mong muốn



proj.4



1. Ngôn ngữ lập trình GIS nguồn mở



Framework

- Là một nền tảng phát triển phần mềm sẵn có
- Cung cấp các cấu trúc và tập hợp các công cụ, **thư viện**, quy tắc để giúp xây dựng ứng dụng

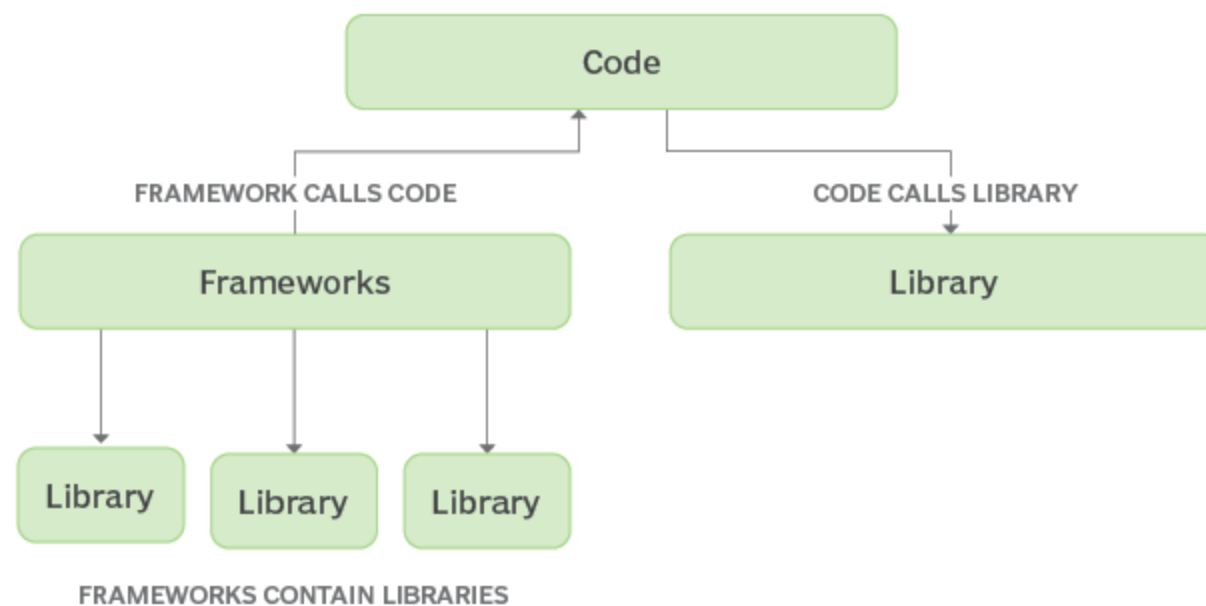


THE
SCIENCE
OF
WHERE™

ArcGIS API for JavaScript
Using Frameworks

1. Ngôn ngữ lập trình GIS nguồn mở

Libraries vs. frameworks



- Thư viện lập trình có thể được sử dụng độc lập
- Framework yêu cầu nhà phát triển tuân thủ cấu trúc và quy tắc của nó

2. Biến, kiểu dữ liệu, toán tử

- Là một vùng bộ nhớ được sử dụng để lưu trữ
- Đại diện cho một giá trị
- Có thể gán và lấy giá trị
- Lưu trữ dữ liệu tạm thời trong quá trình thực hiện chương trình

Definition *Initialization*

↑ ↑

└──────────────────────────┘ └──┘

`data_type variable_name = value`

2. Biến, kiểu dữ liệu, toán tử

Variables in Python

C Programming

Variable of type int
`int x = 10;`
`x = 4.55;`
Can't store a float

Python Programming

Variable of type int
`x = 10`
`x = 4.55`
Not a problem

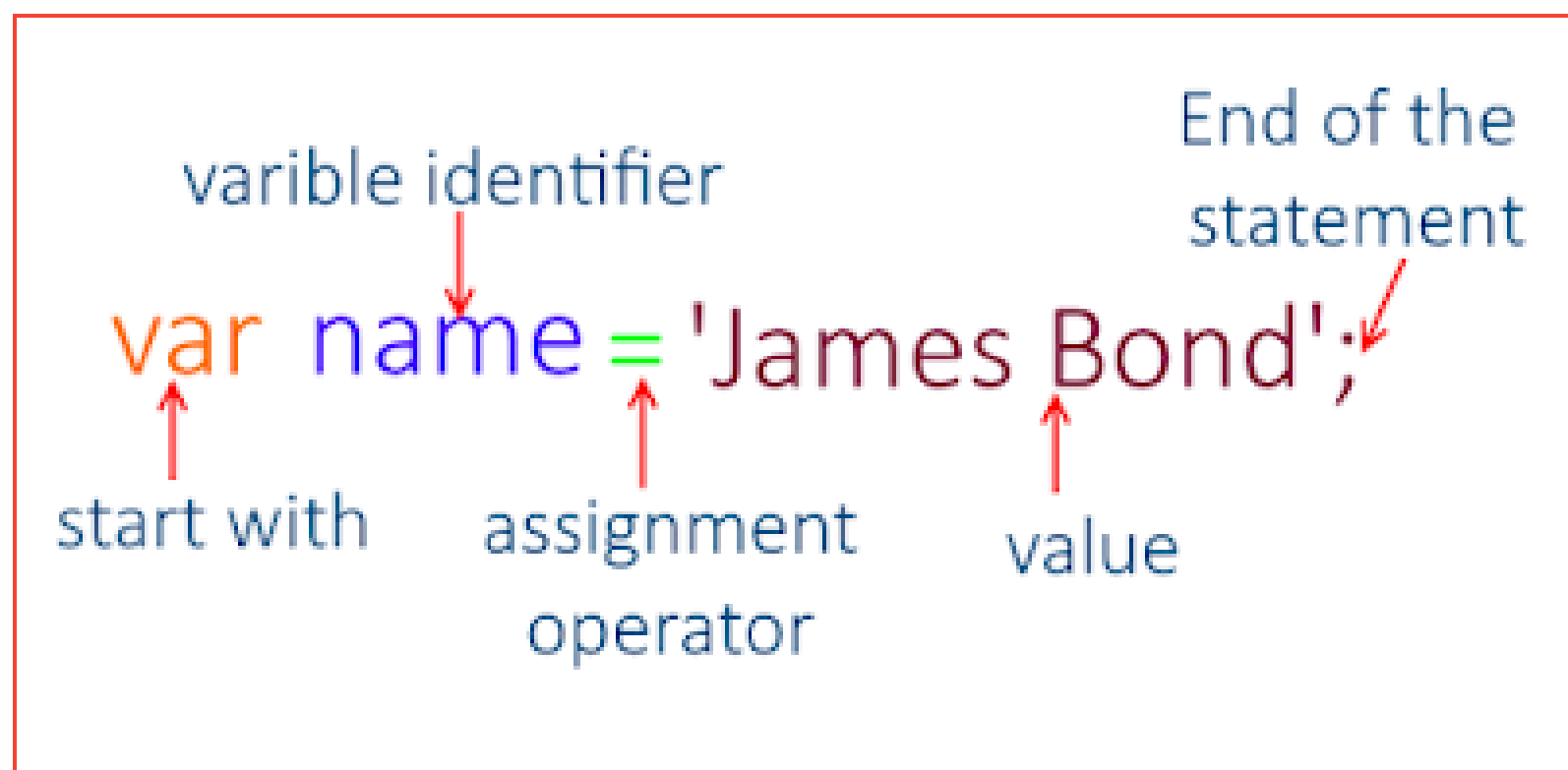
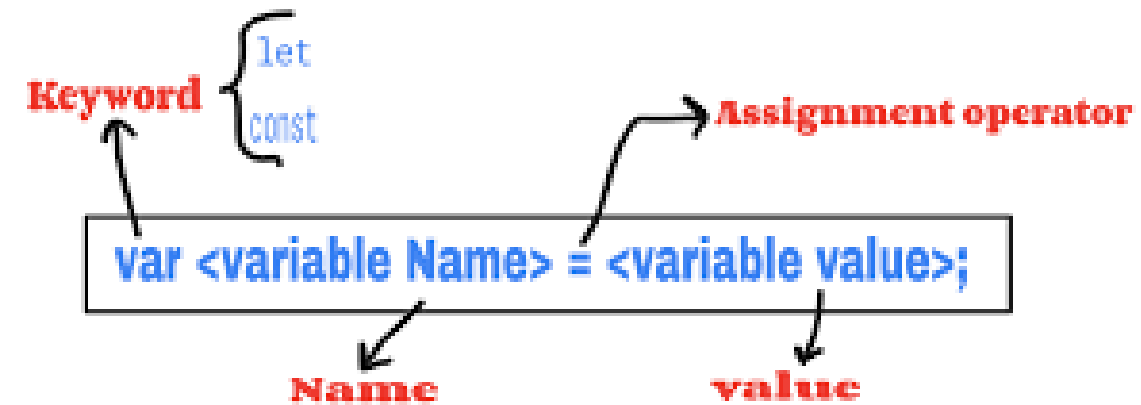
Definition Initialization

↑ ↑


`data_type variable_name = value`

2. Biến, kiểu dữ liệu, toán tử

SYNTAX of Variable Declaration



2. Biến, kiểu dữ liệu, toán tử



VARIABLE NAME VARIABLE VALUE

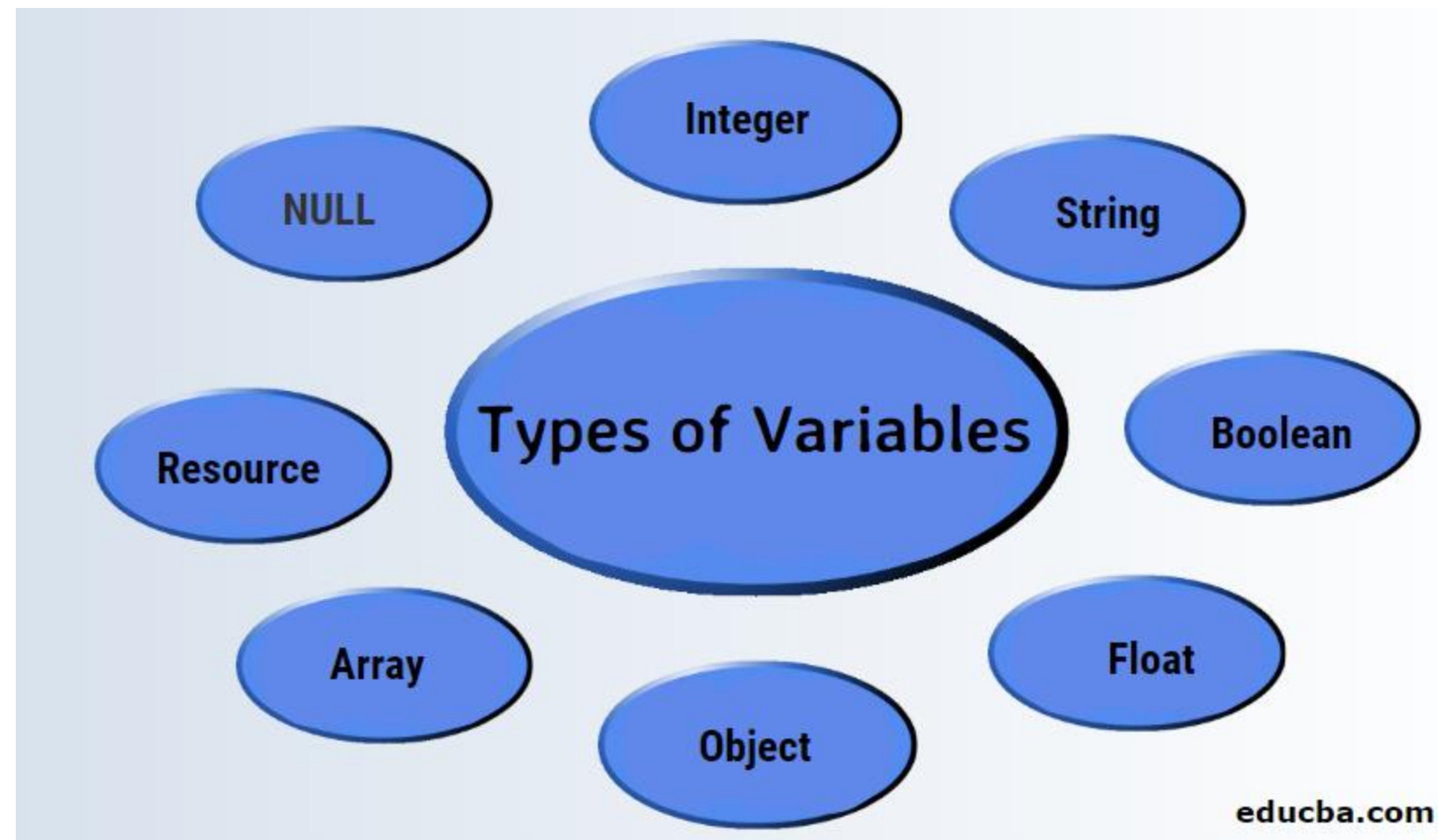
`$name` = `value` ;

└──────────┬──────────┘
EXPRESSION

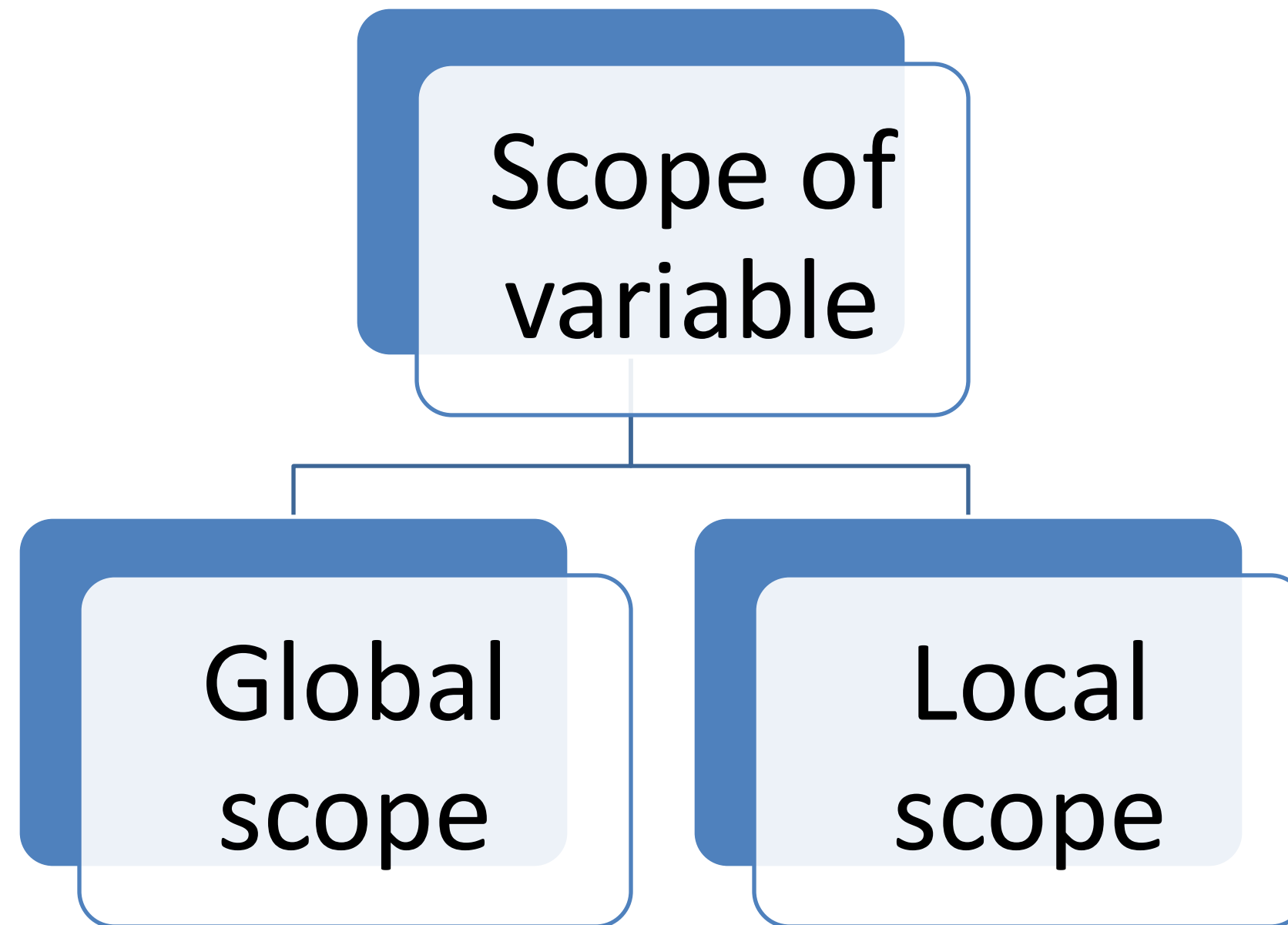
└──────────────────────────┘
STATEMENT

How to Get Started with
PHP Variables

2. Biến, kiểu dữ liệu, toán tử



2. Biến, kiểu dữ liệu, toán tử



2. Biến, kiểu dữ liệu, toán tử



Global scope

Biến toàn cục

- Phạm vi toàn cục
- Truy cập từ bất kỳ đâu trong chương trình
- Được khai báo bên ngoài các khối mã hoặc hàm cụ thể và có thể được truy cập từ bất kỳ đâu trong chương trình

2. Biến, kiểu dữ liệu, toán tử



Local scope

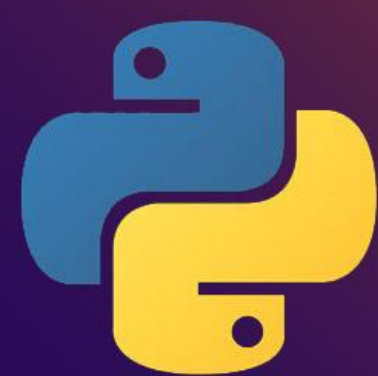
Biến cục bộ

- Phạm vi cục bộ
- Biến được khai báo trong một khối mã hoặc hàm cụ thể
- Chỉ có thể truy cập và sử dụng trong phạm vi khối mã, hàm



3. Cấu trúc điều kiện

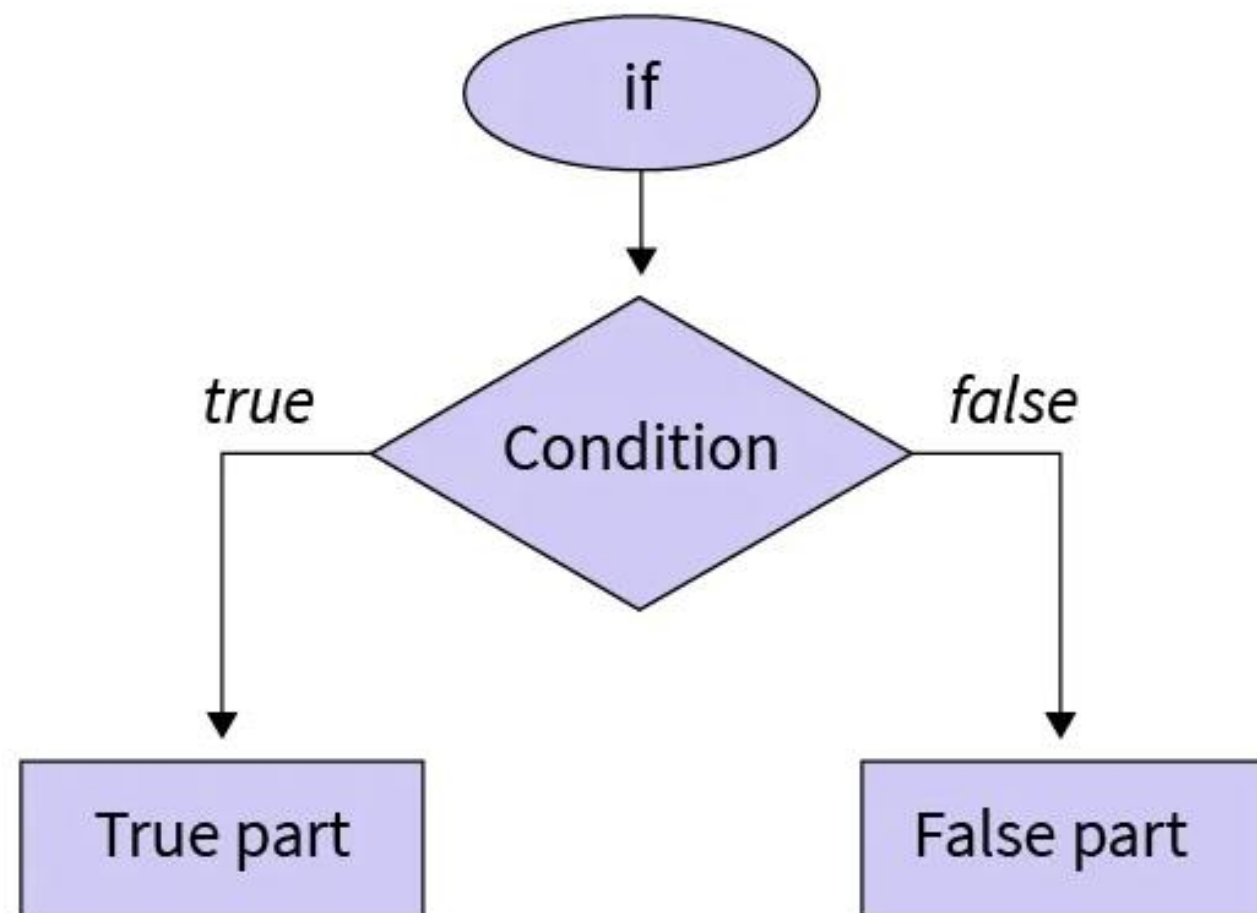
CONTROL STRUCTURES



python



3. Cấu trúc điều kiện

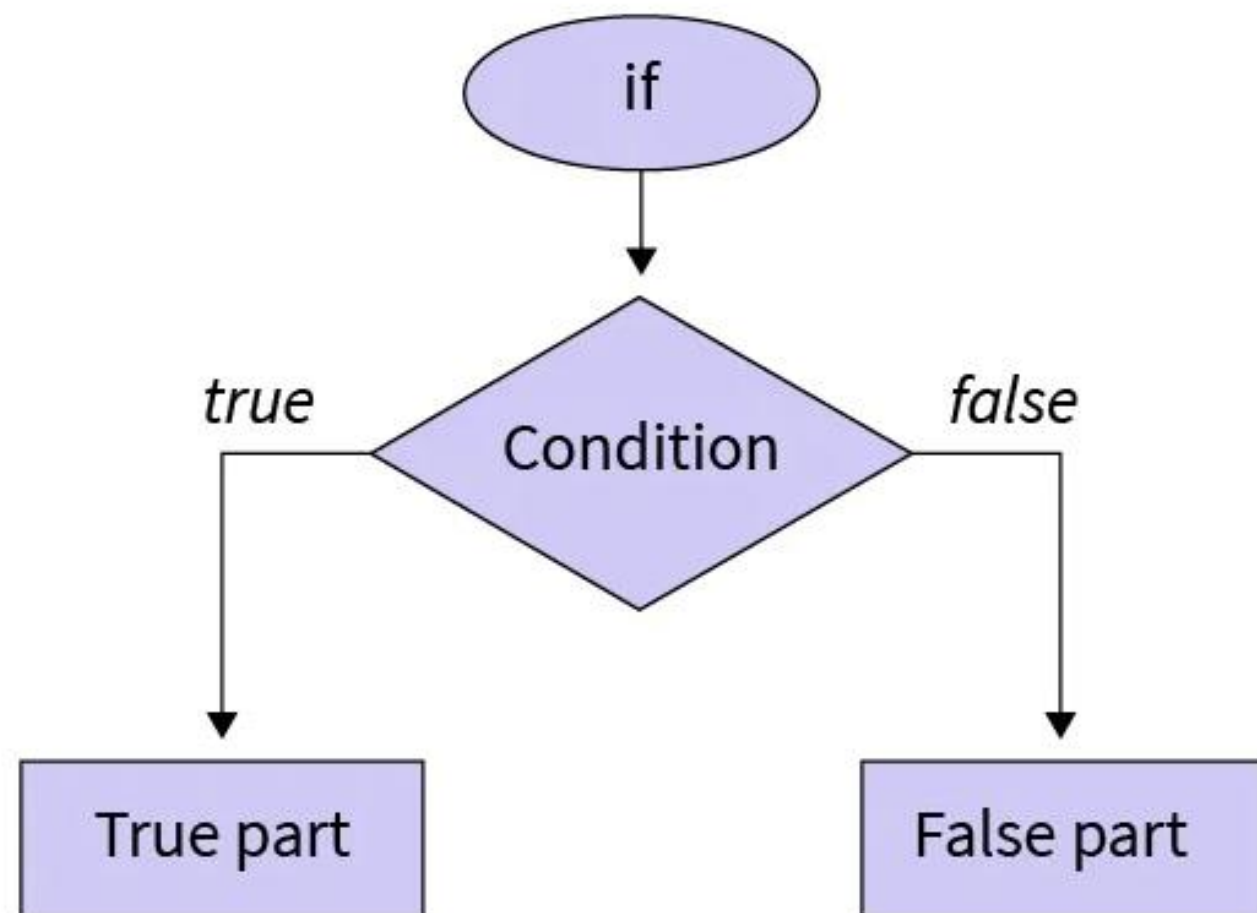


Conditional statements

Cấu trúc rẽ nhánh

- Cho phép chương trình thực hiện các hành động khác nhau dựa trên điều kiện được đưa ra.
- Các câu lệnh điều khiển như **if**, **else if**, **else** được sử dụng để thực hiện các hành động khác nhau tùy thuộc vào điều kiện được kiểm tra

3. Cấu trúc điều kiện



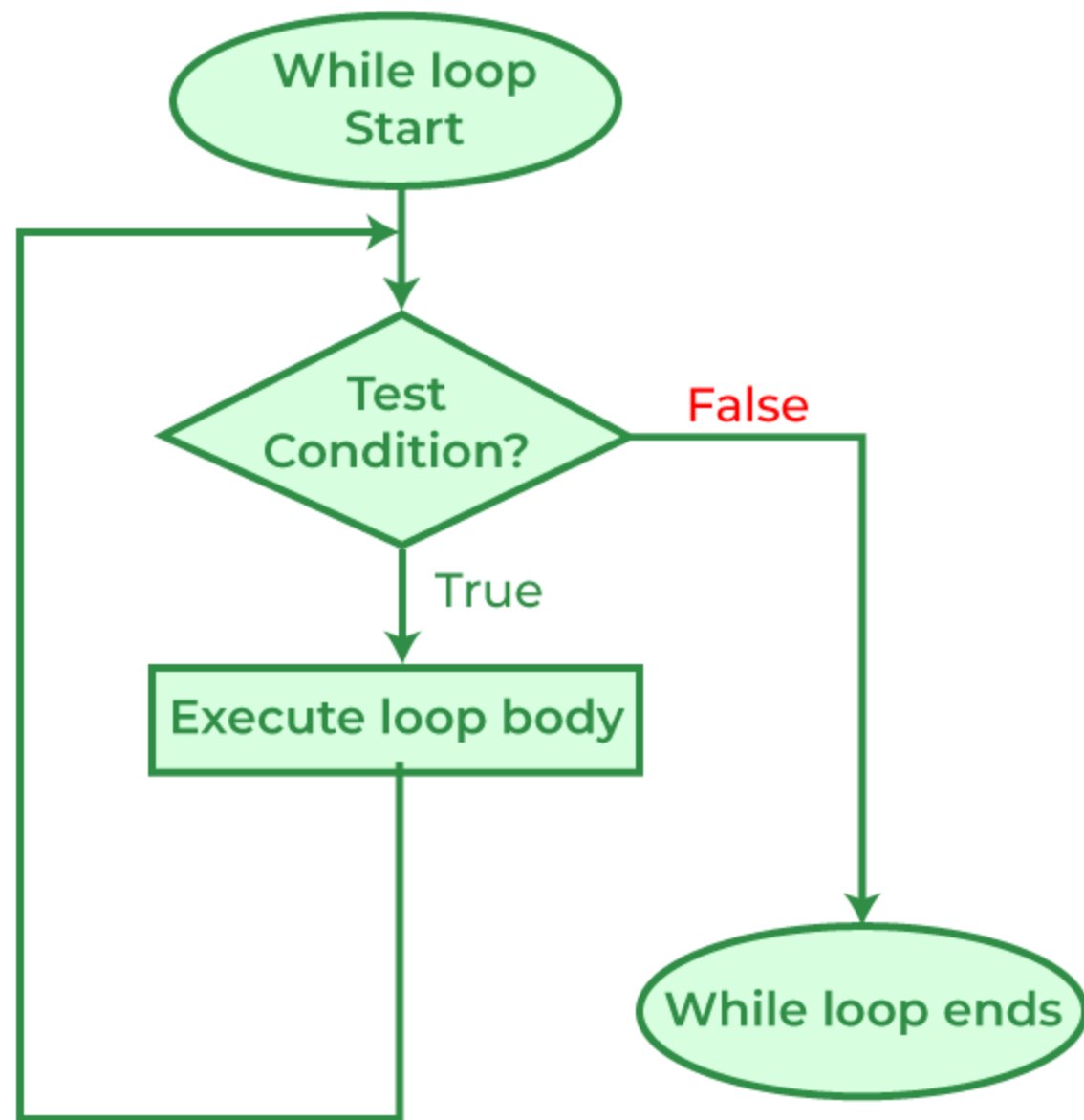
Conditional statements
Cấu trúc rẽ nhánh

```
python
```

```
if condition:
    # thực hiện hành động nếu condition đúng
else:
    # thực hiện hành động nếu condition sai
```

Copy

3. Cấu trúc điều kiện

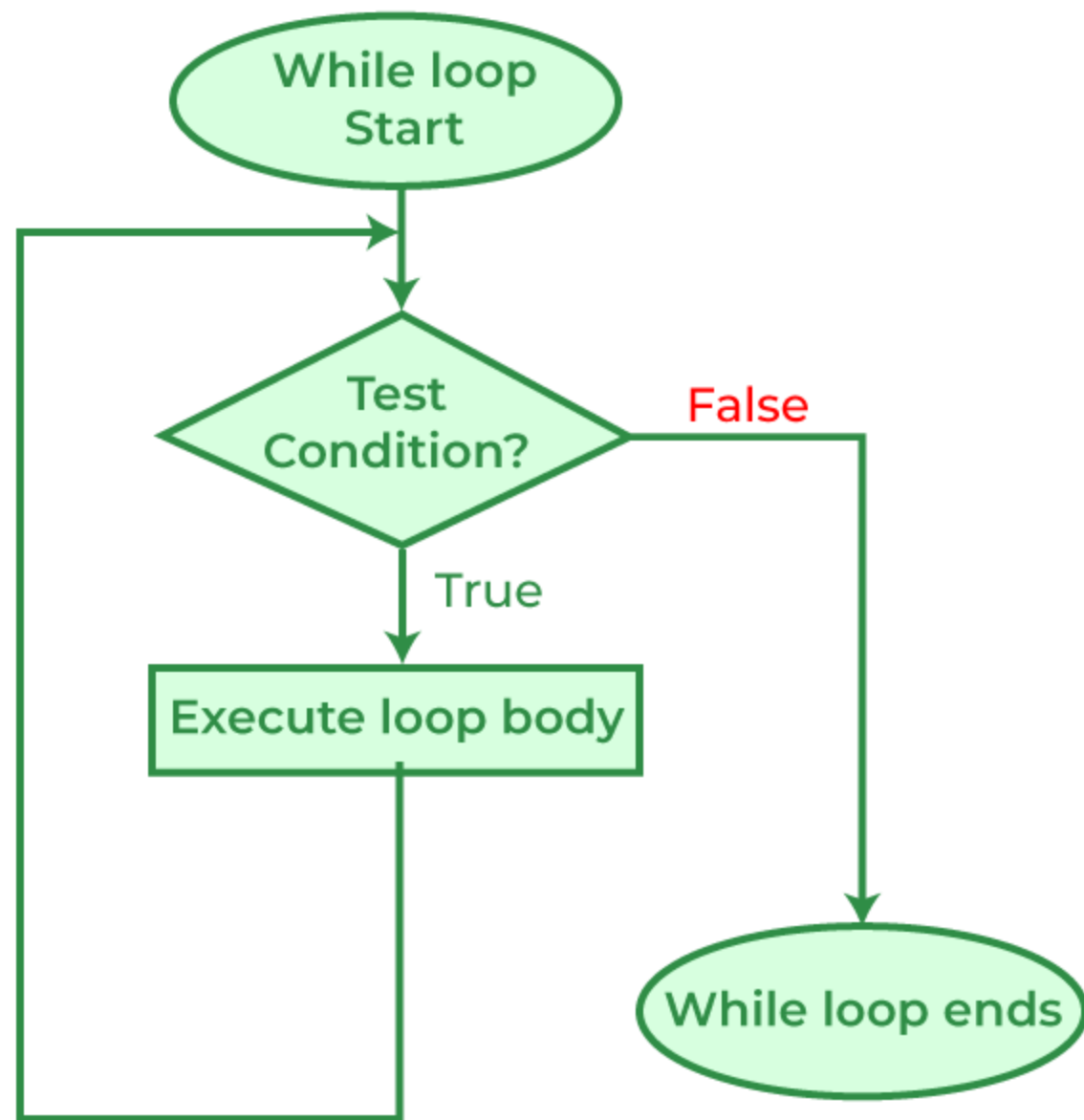


- Thực hiện một nhóm hành động lặp đi lặp lại cho đến khi một điều kiện được đáp ứng
- Các câu lệnh điều khiển như **for**, **while** được sử dụng để thực hiện vòng lặp


Loop statements
Cấu trúc vòng lặp



3. Cấu trúc điều kiện



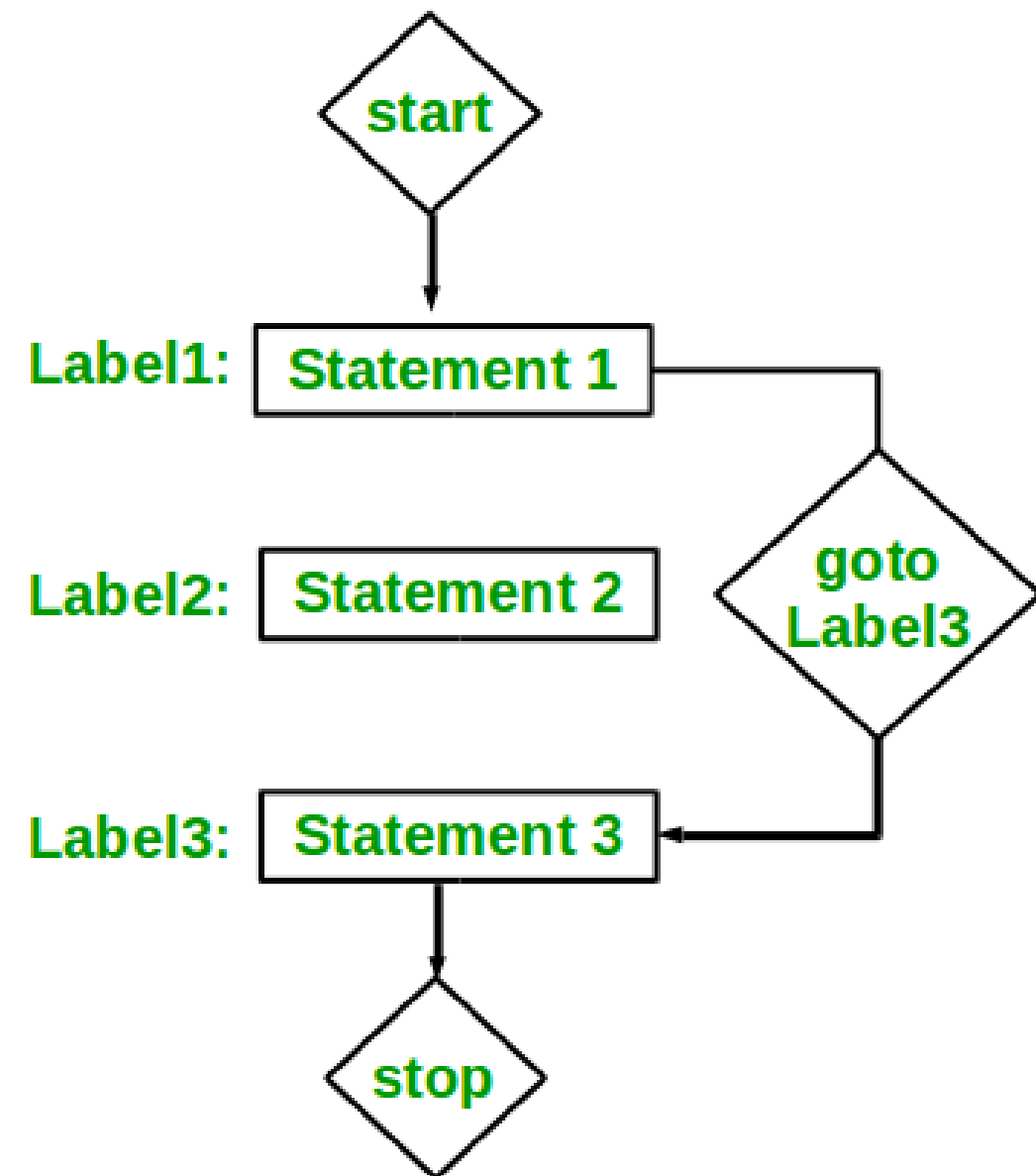
Loop statements
Cấu trúc vòng lặp

python  Copy

```
for i in range(5):  
    # thực hiện hành động 5 lần  
  
while condition:  
    # thực hiện hành động cho đến khi condition sai
```



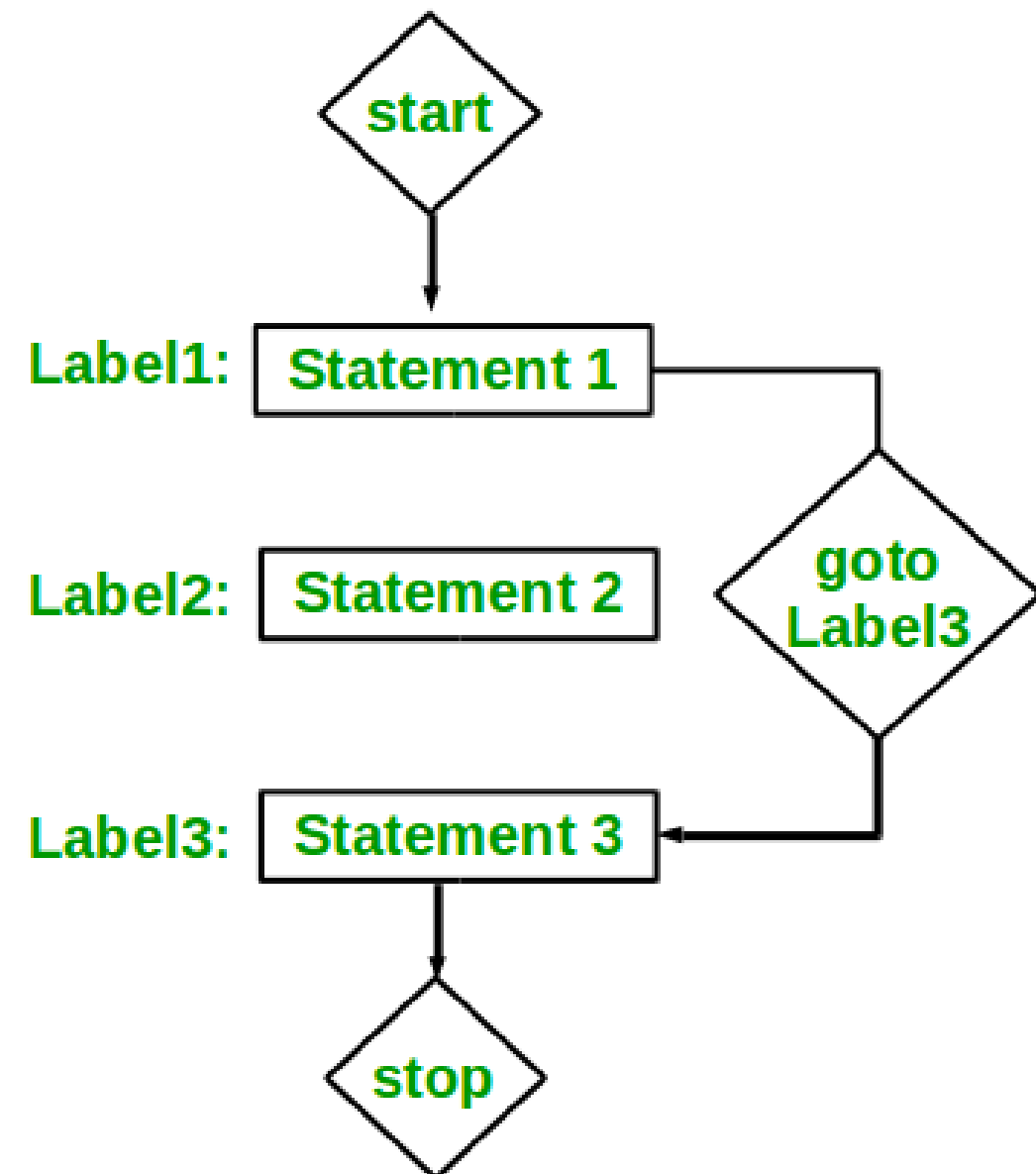
3. Cấu trúc điều kiện



Jump statements
Cấu trúc nhảy

- Cho phép chương trình:
 - Nhảy đến một điểm nhất định trong mã nguồn
 - Thoát khỏi một vòng lặp
- Các câu lệnh điều khiển như **break**, **continue**, **return** được sử dụng để thực hiện các hành động nhảy

3. Cấu trúc điều kiện



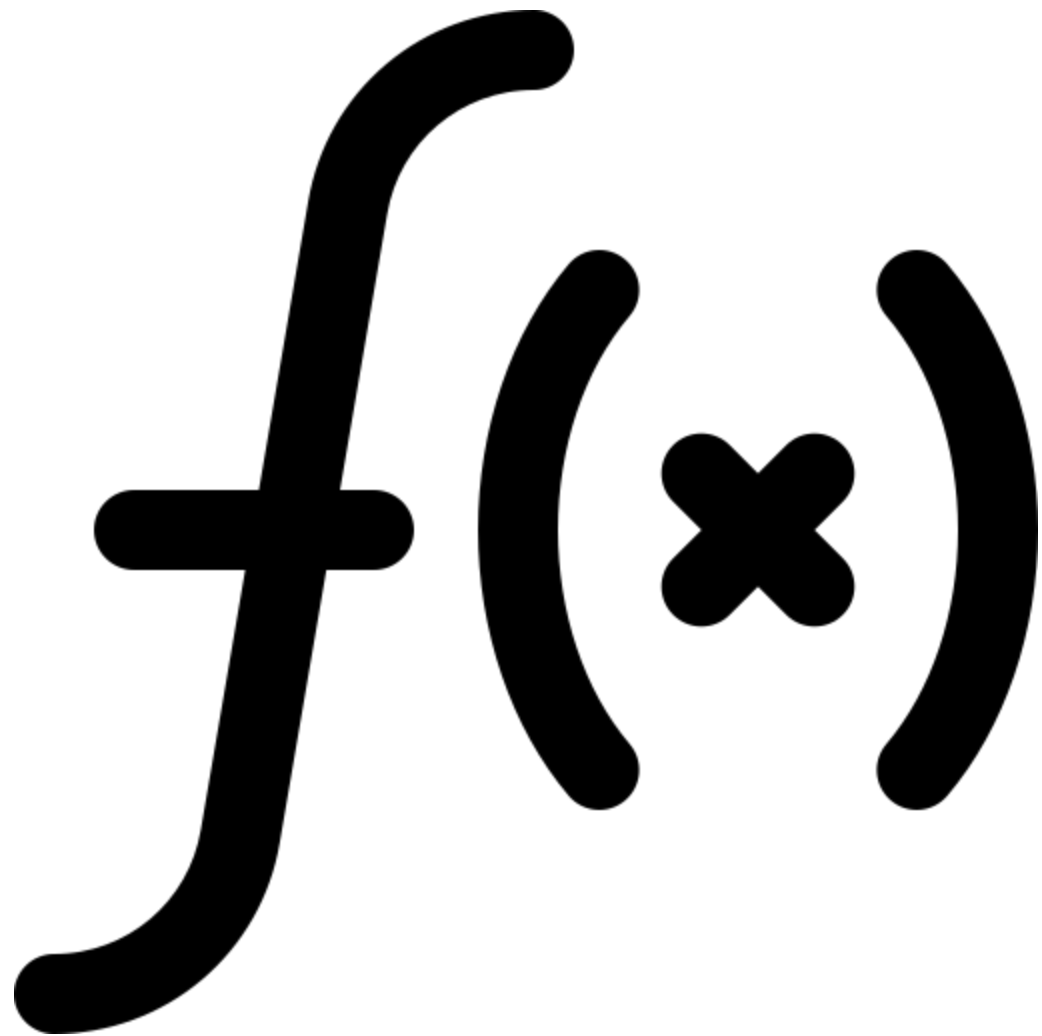
Jump statements
Cấu trúc nhảy

python

 Copy

```
for i in range(10):  
    if i == 5:  
        break # thoát khỏi vòng lặp khi i = 5  
    if i == 3:  
        continue # bỏ qua các hành động còn lại và tiếp tục vòng lặp  
    print(i)  
  
def function():  
    # thực hiện một loạt các câu lệnh  
    return # thoát khỏi hàm và trả về giá trị
```


4. Hàm

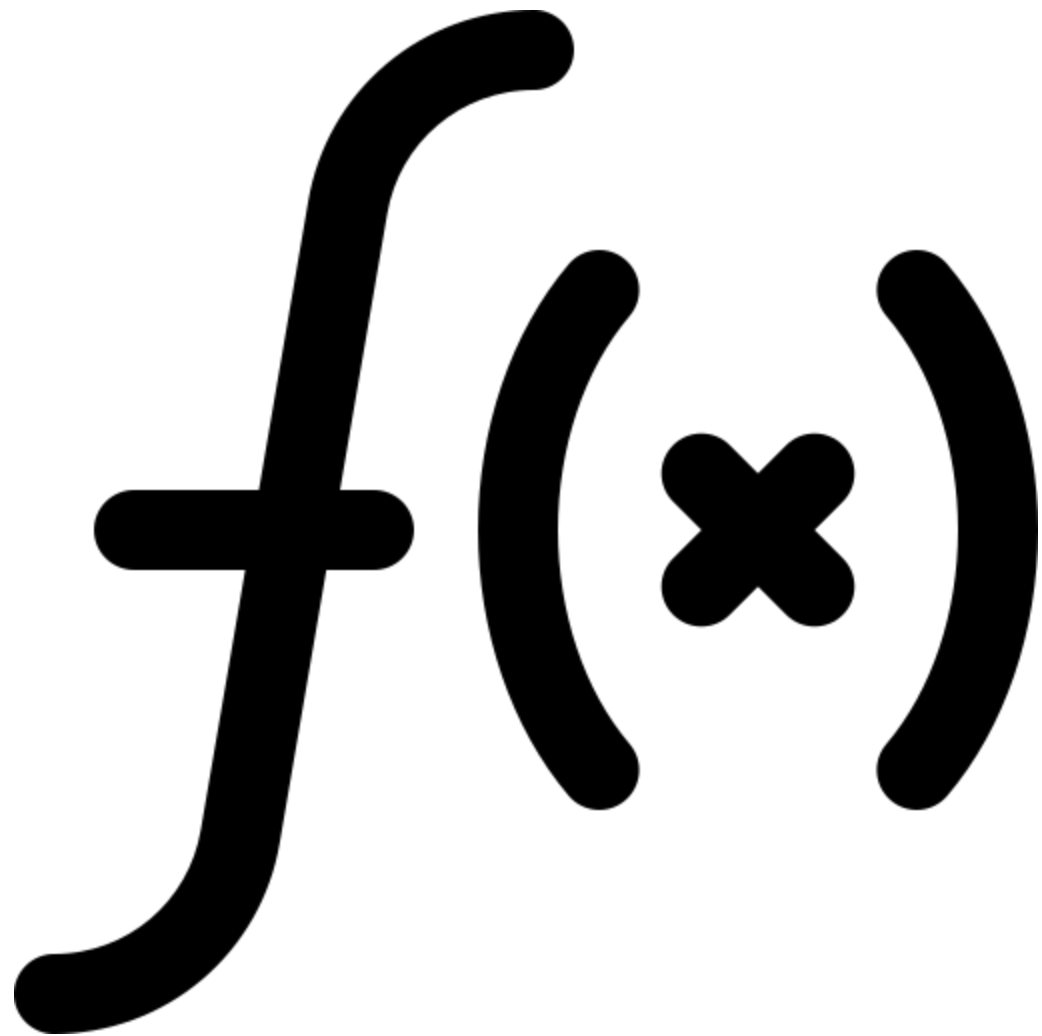


- Hàm là một khối mã được đặt tên và được sử dụng để thực hiện một tác vụ cụ thể
- Hàm giúp tái sử dụng mã
- Có thể được gọi và thực thi từ nhiều nơi trong chương trình

Function
Hàm



4. Hàm



- Hàm thường nhận đầu vào (tham số)
- Thực hiện một số công việc cụ thể dựa trên các đầu vào đó
- Có thể trả về một giá trị

Function
Hàm



4. Hàm

$f(x)$

python

 Copy

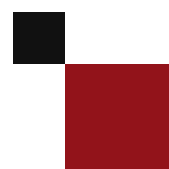
```
def greet(name):  
    print("Hello, " + name + "!")  
  
greet("Steven") # Kết quả: Hello, Steven!
```

Function
Hàm





OPENGIS



THANK YOU

