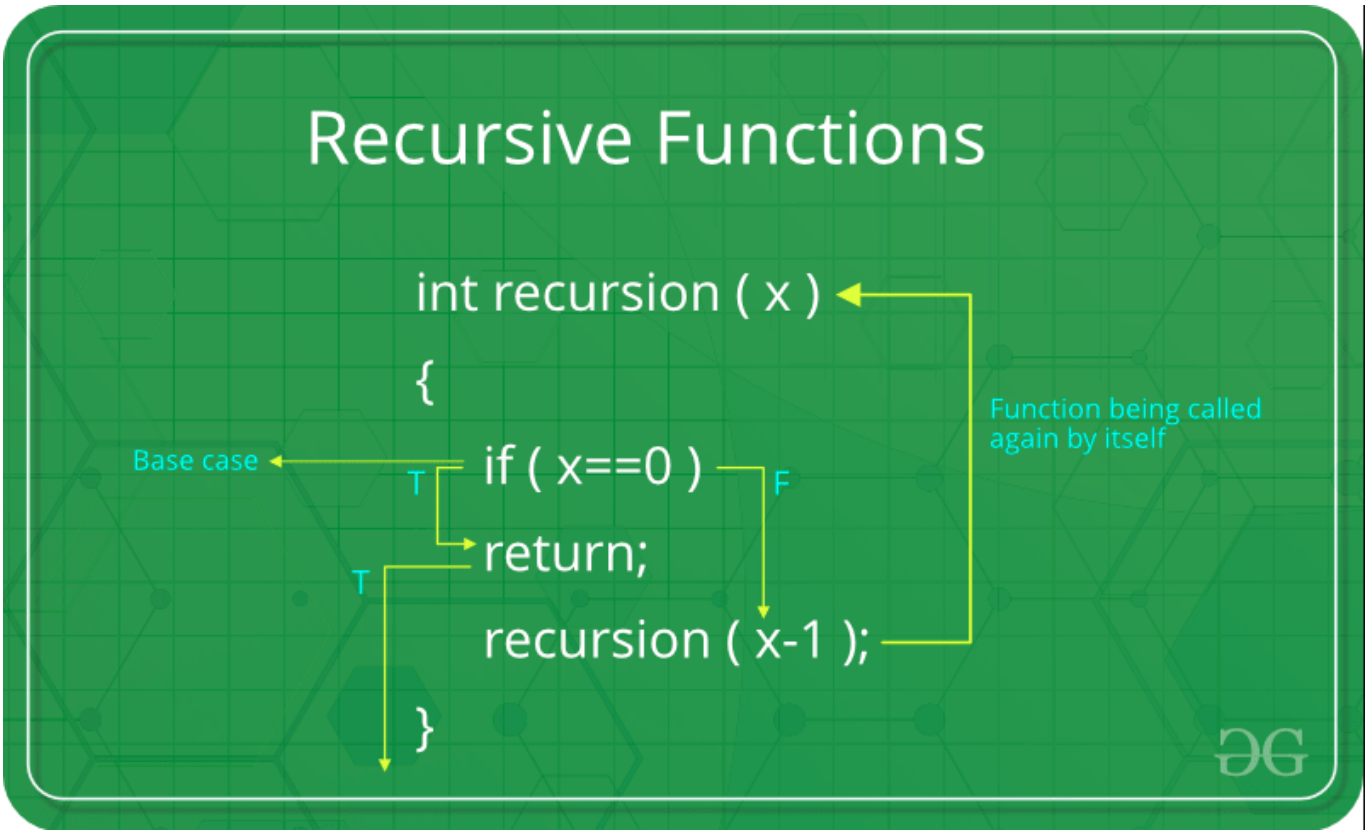
Recursion - Đệ qui



> Source: https://www.geeksforgeeks.org/recursive-functions/

AGENDA

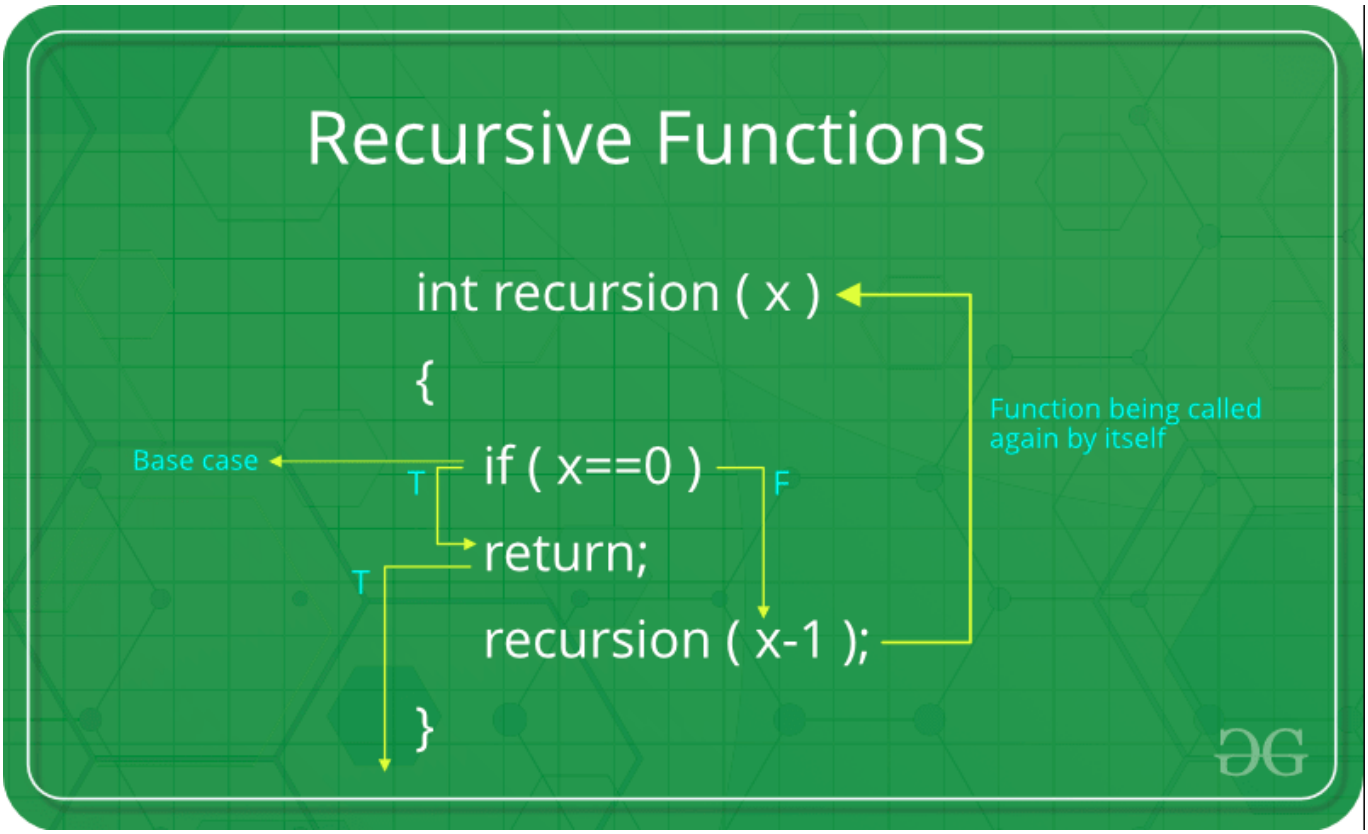
1. Recursion là gì?
2. Một số lưu ý khi làm việc với recursion
3. Có bao nhiêu loại recursion
4. Một số ví dụ về recursion

1. Recursion là gì?

> The process in which a function calls itself directly or indirectly is called recursion and the corresponding function is called as recursive function. Source

Tạm dịch:

- **Recursion** (đệ qui) là quá trình một function nó gọi lại chính nó một cách gián tiếp hay trực tiếp. Và hàm này được gọi là hàm đệ qui (**recursive function**)



> Source: https://www.geeksforgeeks.org/recursive-functions/

2. Một số lưu ý khi làm việc với recursion

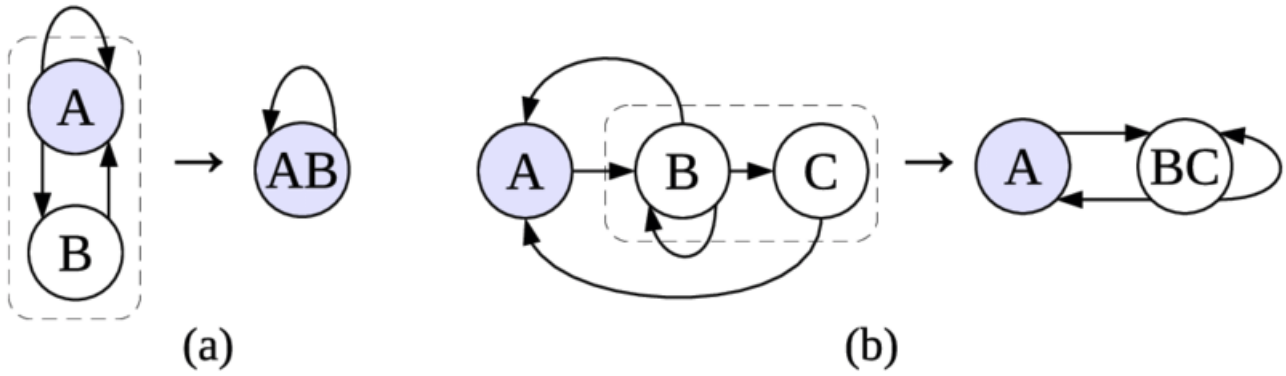
- PHẢI CÓ **BASE CASE** (termination point / điểm dừng), kẻo nó chạy vô tận và xảy ra lỗi `StackOverflow`.
- The memory is incremented by the number of times a recursive function is called.
- Mobile has very limited memory space to execute any apps. If you are developing a mobile application, avoid using recursion.

> Source: https://www.csestack.org/recursion/#Interview_Questioned_asked_about_recursion

3. Có bao nhiêu loại recursion

#	Name	Desc
1	Primitive recursion	Có thể chuyển đổi sang dạng loop
2	Tail recursion	Là một dạng primitive recursion mà lời gọi recursive function nó nằm ở cuối hàm.
3	Single recursion	Chỉ có một lời gọi recursive function
4	Multiple recursion	Có nhiều lời gọi recursive function
5	Mutual or Indirect recursion	Có ít nhất từ 2 functions trở lên tạo nên recursion này. `A -> B -> C -> A`
6	General recursion	Trái ngược với Primitive recursion, chỉ có thể biểu diễn dưới dạng recursive function

> Source: https://www.csestack.org/recursion/



> Source: https://www.researchgate.net/figure/The-elimination-of-indirect-recursion-by-merging-of-functions-code_fig3_327408526

4. Một số ví dụ về recursion

```
Tính S(n) = 1 + 2 + ... + n
```

```
// using math
function calculateS(n) {
  if (n <= 0) return 0;

  return n * (n + 1) / 2;
}
```

```
// using loop
function calculateS(n) {
  if (n <= 0) return 0;

  let sum = 0;
  for (let i = 1; i <= n; i++) {
    sum += i;
  }

  return sum;
}
```

```
// using recursive function
function calculateS(n) {
  // base case
  if (n <= 0) return 0;

  // tail recursion
  return n + calculateS(n - 1);
}

calculateS(5); // 15
// S(5) = 5 + S(4)
// S(4) = 4 + S(3)
// S(3) = 3 + S(2)
// S(2) = 2 + S(1)
// S(1) = 1 + S(0)
// S(0) = 0 BASE CASE / TERMINATION
```

Tham khảo

- https://www.geeksforgeeks.org/recursion/
- https://www.csestack.org/recursion/#1_Primitive_Recursion

Khoá học Javascript cho người mới bắt đầu 2021 🎉

- Tác giả: **Hậu Nguyễn** - Founder Easy Frontend
- Khoá học chỉ được published trên Udemy, không thông qua trung gian.
- Khoá học không bán dạng videos upload trên Google Drive hay bất cứ hình thức nào tương tự.
- Khoá học có nhóm discord để hỗ trợ trong quá trình học tập.

☎️ Liên hệ tác giả để được hỗ trợ:

- ✅ Facebook: https://www.facebook.com/nvhauesmn/
- ✅ Fanpage: https://www.facebook.com/learn.easyfrontend
- ✅ Youtube Channel: https://www.youtube.com/easyfrontend