1. Con số không có ý nghĩa rõ ràng

Ví dụ:

Bad :

Enable\_sim(15);

Good:

Const int FIFTEEN\_SECOND;

Wait\_enable\_sim(FIFTEEN\_SECOND);

1. Tráng dùng nhiều vòng lặp lồng nhau

Bad : for( ; ; )

{

for( ; ;){}

}

Good:

Void function()

{

For( ; ;)

}

For{ ; ; }

{

Void function();

}

1. Không phụ thuộc vào command

Ví dụ:

Bad practice:

Const int a = 60 // số phút của 1 giờ

Good practice:

Const int MINUTES\_OF\_THE\_HOUR = 60

1. Tránh viết function quá dài, mỗi function chỉ nên thực hiện 1 chức năng

Bad:

Void bigFunction()

{

Function1;

Function2 ;

}

Good:

Void function1(){}

Void function2(){}

Void bigFunction()

{

Function1();

Function2():

}

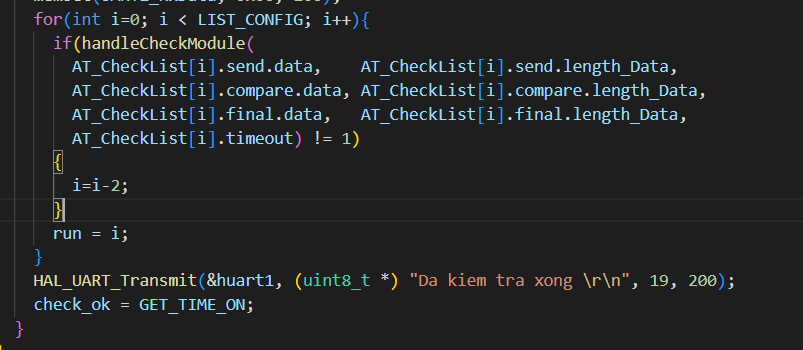
1. Tránh lặp code

Bad:

Text

Description automatically generated

Good:



1. Đặt tên biến theo chuẩn camel case

Tên bắt đầu bằng chữ cái viết thường và tất cả chữ cái đầu tiên của những từ tiếp theo sẽ được viết hoa (áp dụng cho biến, hàm, các định danh khác ).

Vi du:

Int setModule;

Void getData();

1. Đặt tên có ý nghĩa

Bad: void chay(char abc[]);

Good: void setupModule(char keyCheck[]);

1. Ưu tiên tên mô tả chi tết hơn ngắn gọn

Bad:

void cut( void \*data, const char source[], const char startString[], const char finalString[]);

Good:

void getStringToString( void \*data, const char source[], const char startString[], const char finalString[]);

1. Sử dụng một động từ nhất quán cho mỗi khái niệm

Bad: getHours();

receiveMinute();

Good: getHours();

getMinute();

1. Viết hoa các giá trị const

Vi du: const int TEN\_SECOND = 10;

11. tên có ý nghĩa

Sử dụng tên tiết lộ mục đích

Ví dụ:

Tránh tên với thông tin sai lệch

Tên phải phân biệt được ý nghĩa

Sử dụng tên có thể tìm kiếm

Tránh những ký tự mã hoá

12. hàm

Hàm nên chia thành những chức năng nhỏ, để có thể tái sử dụng

Mỗi hàm nên thực hiện 1 chức năng

Hàm không có tác dụng phụ

Không lặp lại

Link:

<https://hackernoon.com/how-to-write-clean-code-d557d998bb08>

<https://techmaster.vn/posts/36509/13-quy-uoc-can-thiet-de-viet-clean-code>