**MAKE FILE**

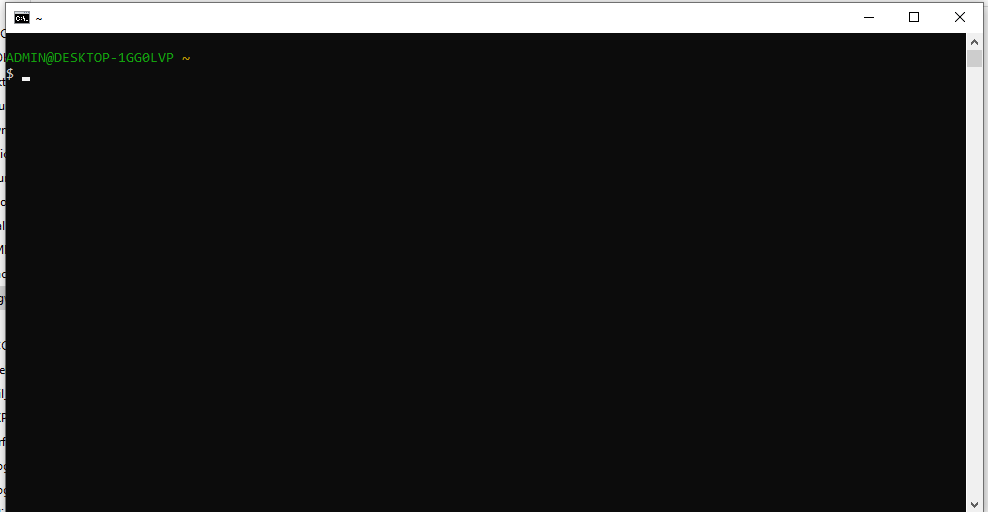
1. **Cách cài đặt Cywin:**

Tải Cywin link: <https://drive.google.com/file/d/1LVABWODqzhRjqRXSJQOxJi_KD6O7Sz3j/view>

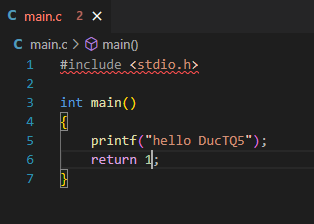
Sau khi tải về giải nén và trong ổ C của máy tính chúng ta. Nó là công cụ để chạy GCC

Tải GCC: link: <https://developer.arm.com/downloads/-/gnu-rm>

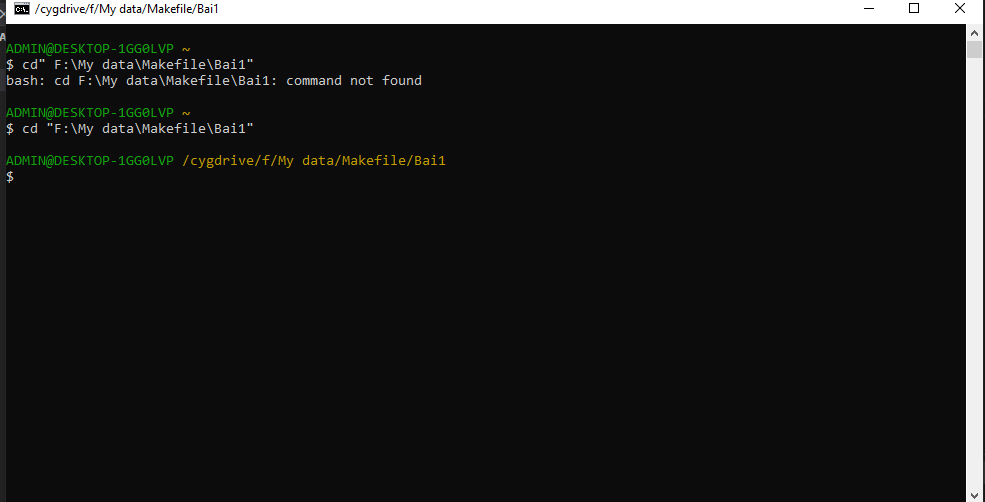
Để mở Cywin: Vào ổ C chọn Cywin-> Cywin.bat



Ví dụ:



Đầu tiên chúng ta phải trỏ đến folder chứa file Main.c



Comand:

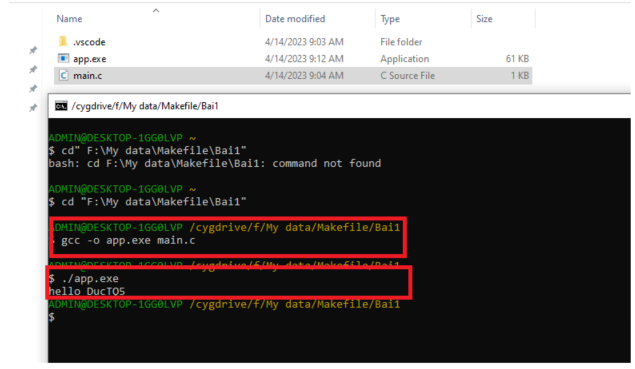
Gcc –v: để kiểm tra version của GCC

CTRL+L: Clear màn hình

ls: kiểm tra có file chưa

cat Makefile: xem file code trên cywin

Để chuyên từ file .c về file thực thi .exe



Gcc -o app.exe: file output từ file main.c

1. **Cấu trúc Makefile:**

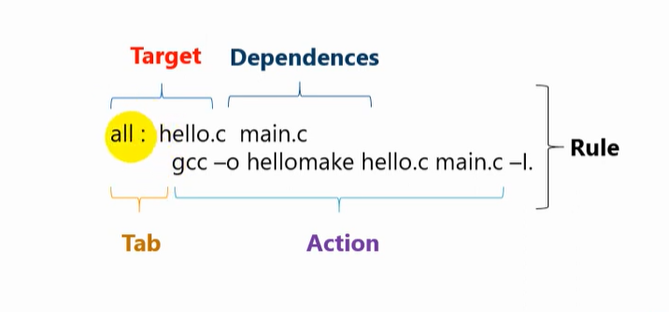
Makefile là một script bên trong có chứa các thông tin:

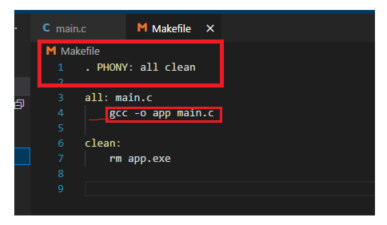
+ Cấu trúc của một project (file, dêpndency)

+ Các command line dùng để tạo – hủy file

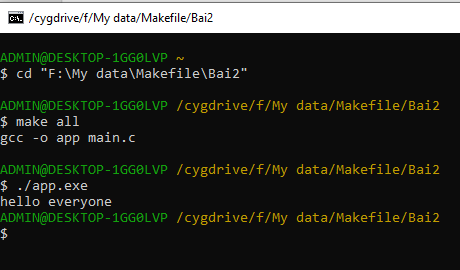
Chương trình make sẽ đọc nội dung trong Makefile và thực thi nó.

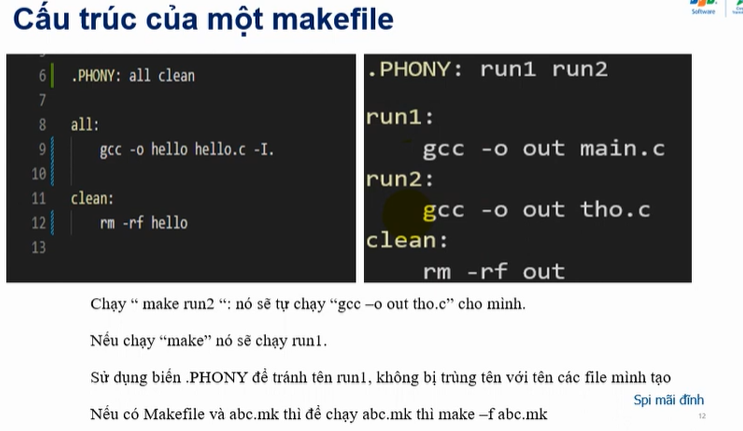
Cấu trúc make file:



app

rm app.exe : xóa file đó đi





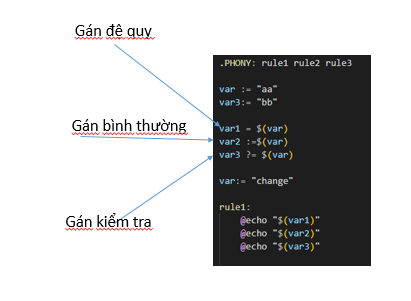
Nếu có hai file make file: **Makefile** và **a.mk**. Làm sao để biết chạy file makefile nào.

Chúng ta sẽ dùng –f để chỉ định file makefile nào được thực thi như hình vẽ



1. **Biến và Include trng makefile:**

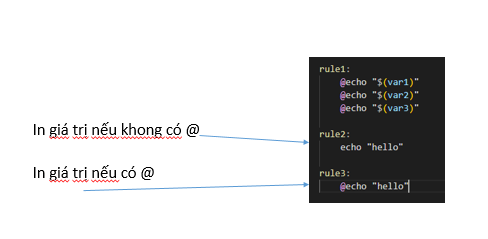
Makefile có ba loại biến thường dùng: Gán đệ quy, gán bình thường, gán kiểm tra.

mai

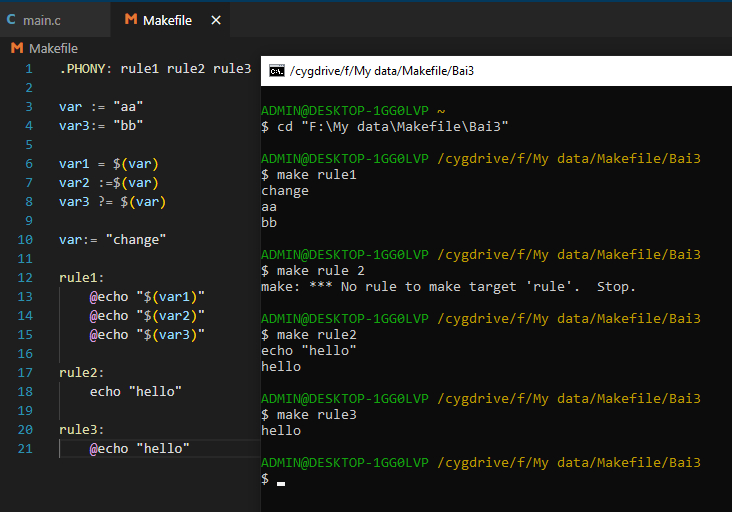
Gán đệ quy: khi thay đổi gia trị thì nó sẽ bị thay đổi theo

Gán kiểm tra: kiểm tra xem ban đầu có giá trị hay chưa.

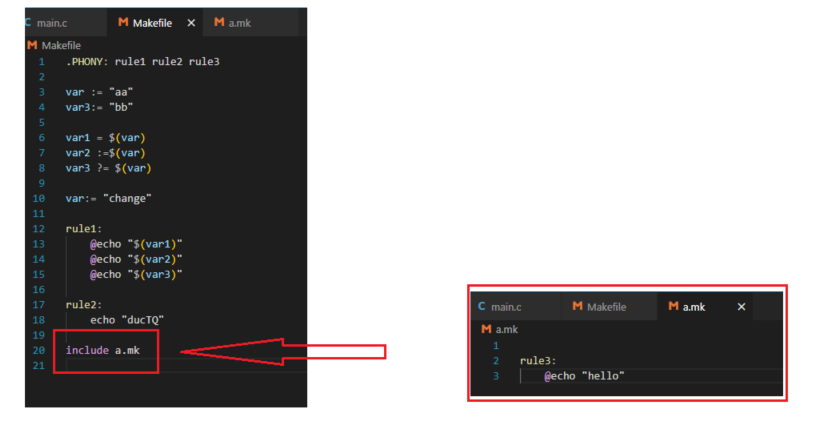
Khi in giá trị ra sẽ có sự khác nhau giữa có @ và không có @



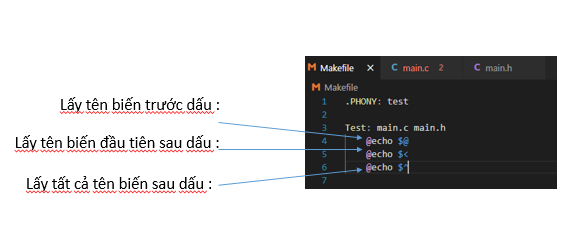
Kết quả:

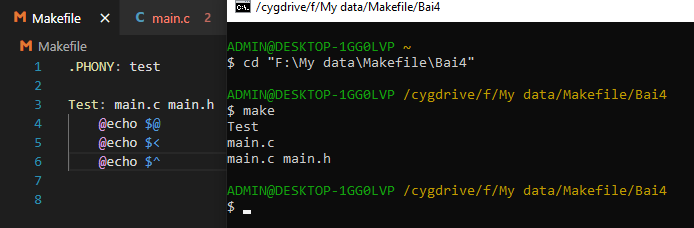


* Include một file makefile vào trong makefile



1. **Các biến đặc biệt trong Makefile**

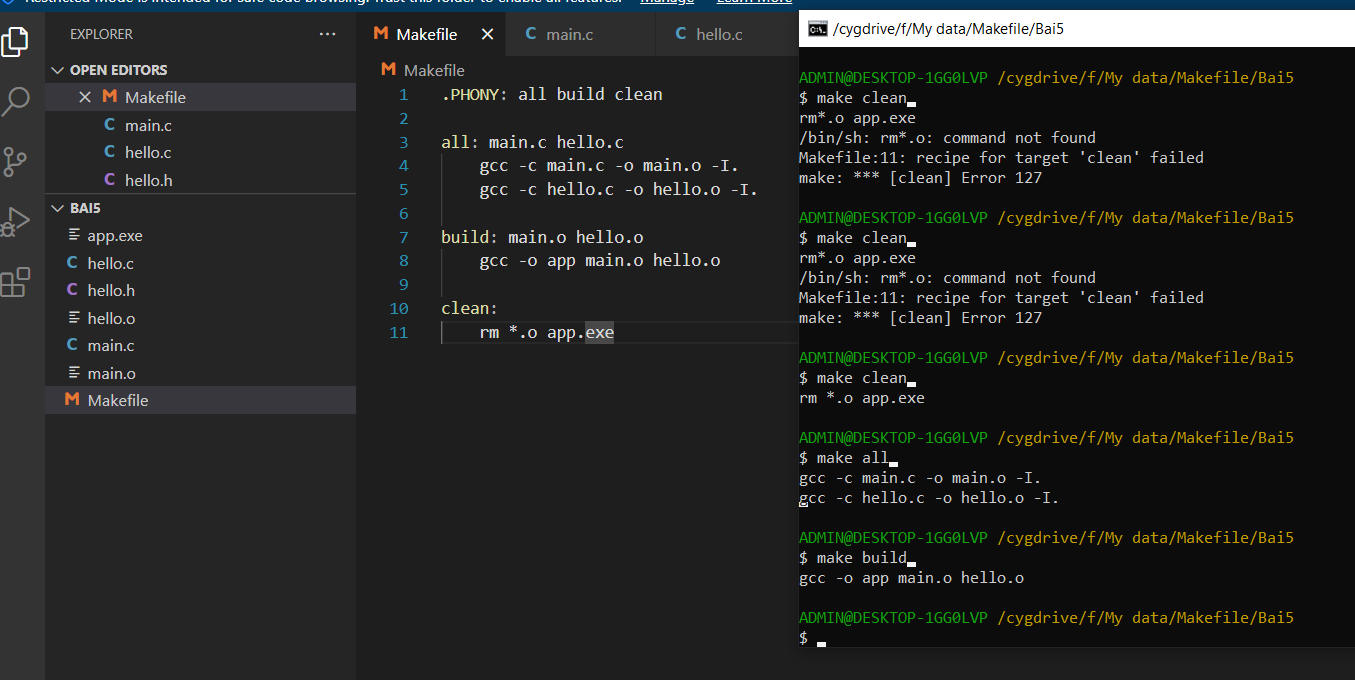




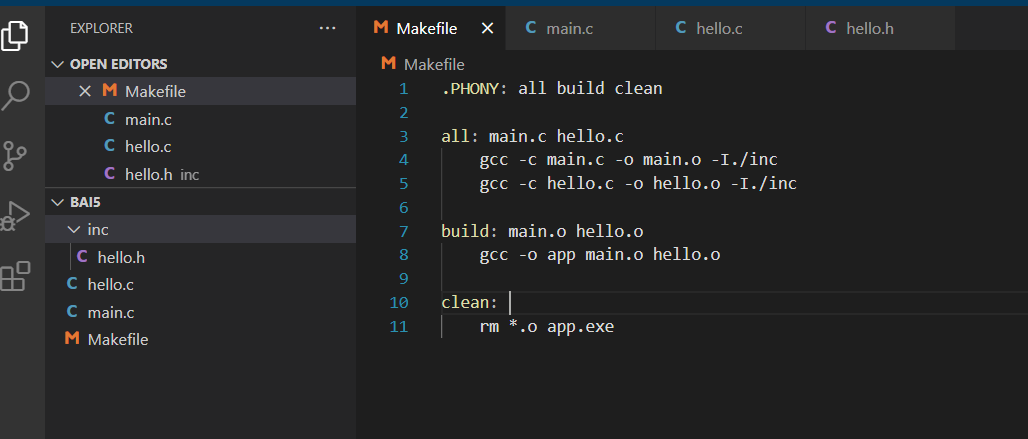
1. **Include trong Makefile**

* Include trong cùng một folder:

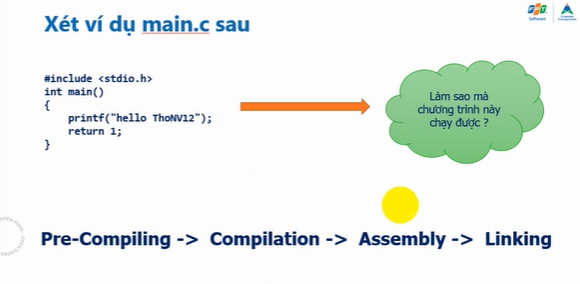
Đầu tiên chúng ta có các file .c build ra .o rồi liên kết các file .o với nhau



* Include khác folder:



1. **Bốn quá trình biên dịch trong C Makefile**



1. Pre-compiling:

**GCC –E hello.c –o hello.i**

**+** Loại bỏ các comments

**+** Mở rông các marcos

**+** Biên dịch các câu lệnh điều kiện và kết quả thu đucợ là file .i

1. Compilation

Gcc –S hello.i –o hello.s

Ở gia đoạn này mã nguồn sẽ tiếp tục thực hiện biên dịch từ file .i sang file .s

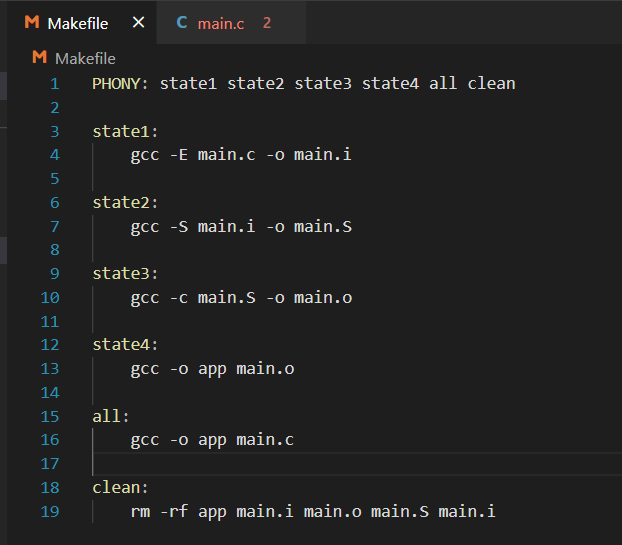
1. Assembly

Gcc –c hello.s –o hello.o

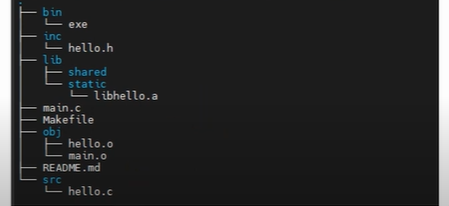
File .s ở giai đoạn tiếp tục sử dụng cho giai đoạn này và thông qua assebly, output là file .o

1. Linking

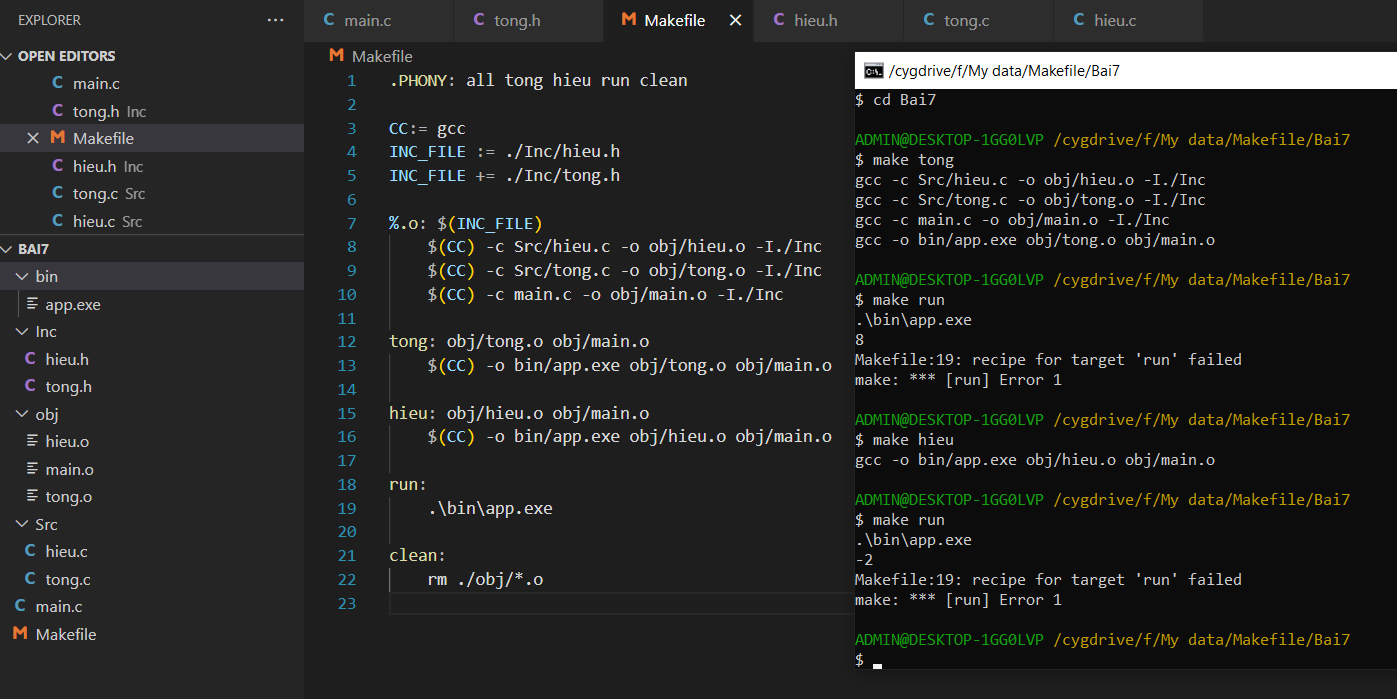
Gcc –o hello hello.o



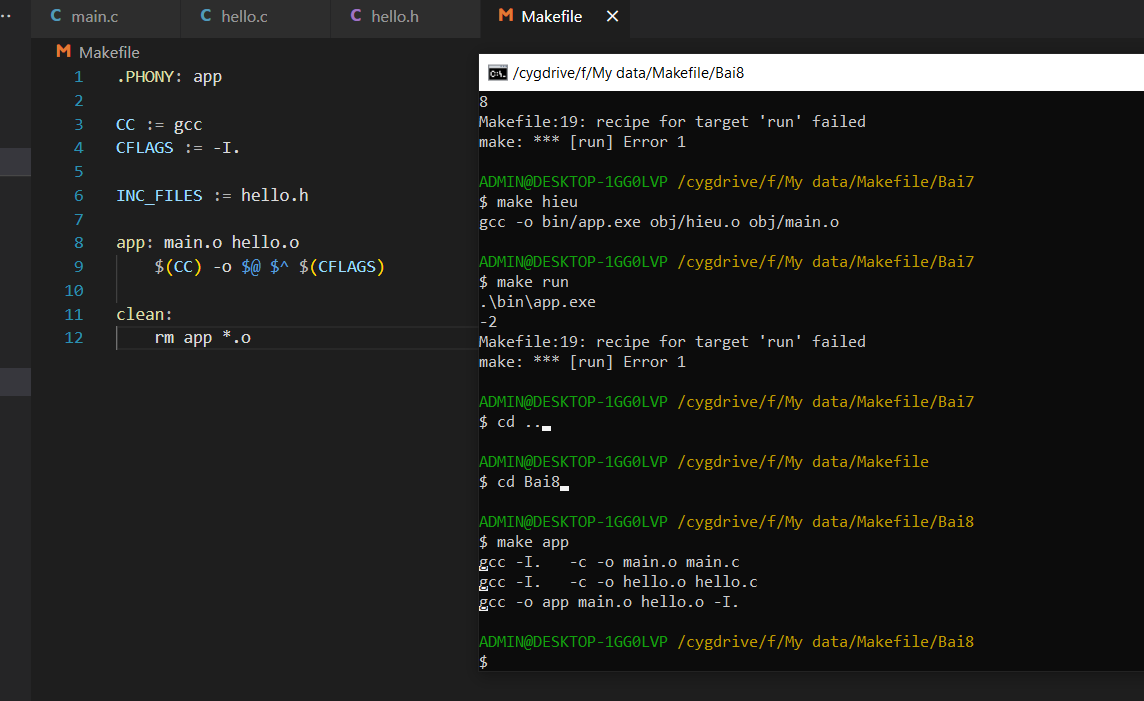
1. **Cấu trúc folder**



Ví dụ: Cách phân chia cấu trúc folder:

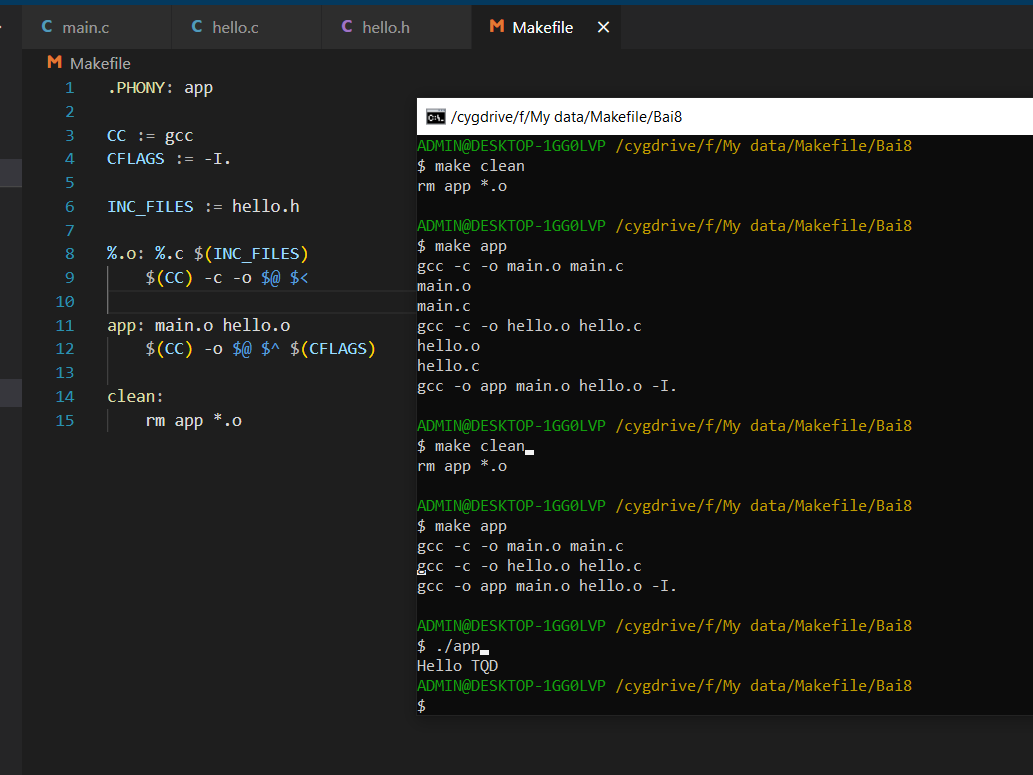


1. **Rure ẩn và cách tự động build trong makefile**

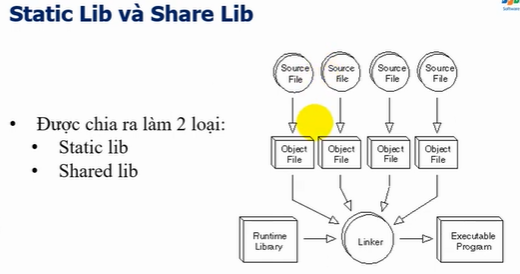


Vì sao khi chúng ta chạy make app nhưng nó lại chạy hết 3 dòng. Do chung có main.o và hello .o bằng cách tạo ra hai rule ẩn. Sau đó nó mới kết hợp vói file.h. Tuy nhiên sẽ khó phát hiện lỗi sai.

Cách khắc phục:



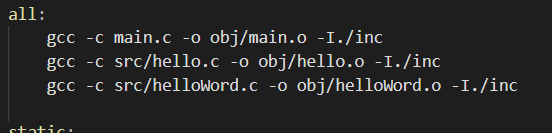
1. **Tạo static Lib trong Makefile**



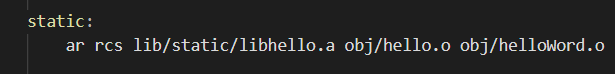


Ban đầu chúng ta phải có folder inc, src, file main.c và makefile

B1: tạo các file object từ các file .c

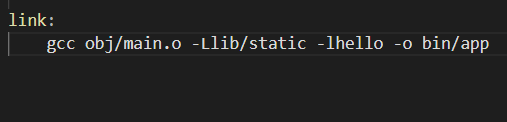


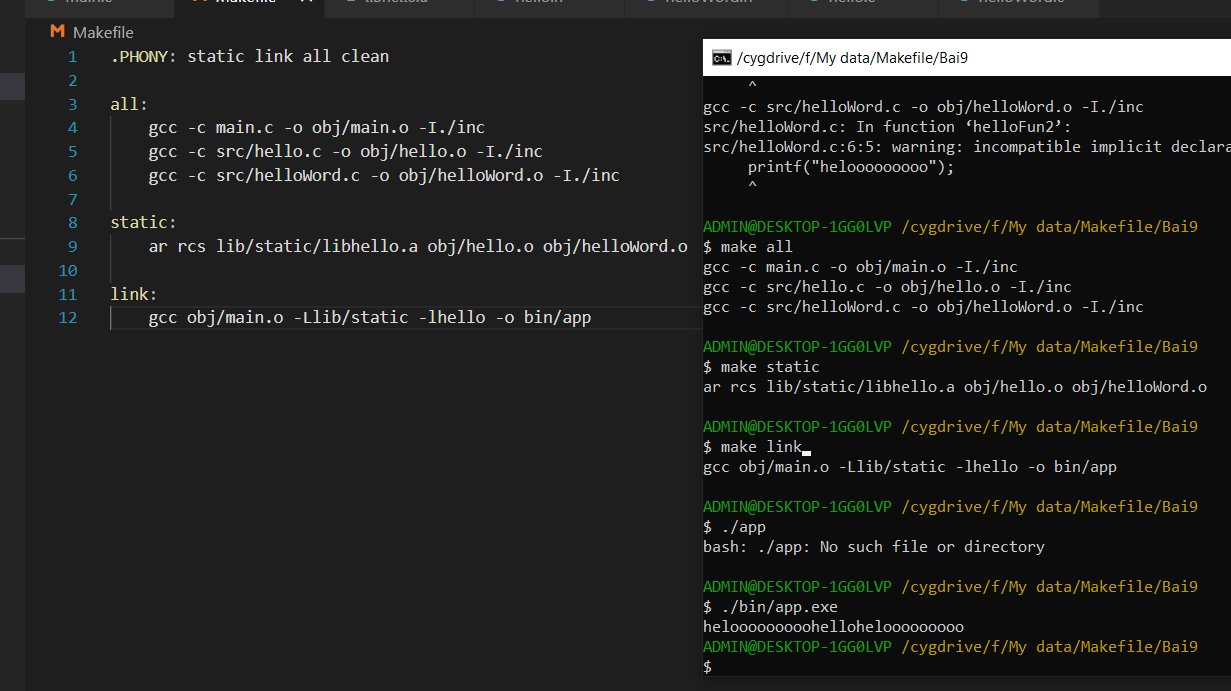
B2: Tạo ra static lib bằng cách link các file .o lại với nhau, bản chất là link các hàm lại với nhau



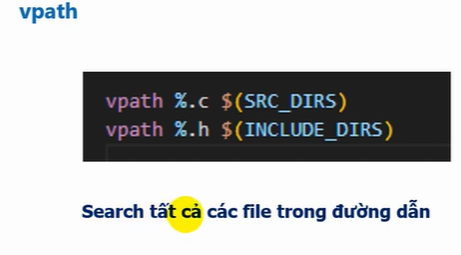
B3: từ file main.o và file libhello.o để build ra file thực thi

Thêm “-L” vào đường dẫn của file.a và thêm “-l” và bỏ kí tự “lib” và “.a” khi đó sẽ trở thành “-lhello” là được.

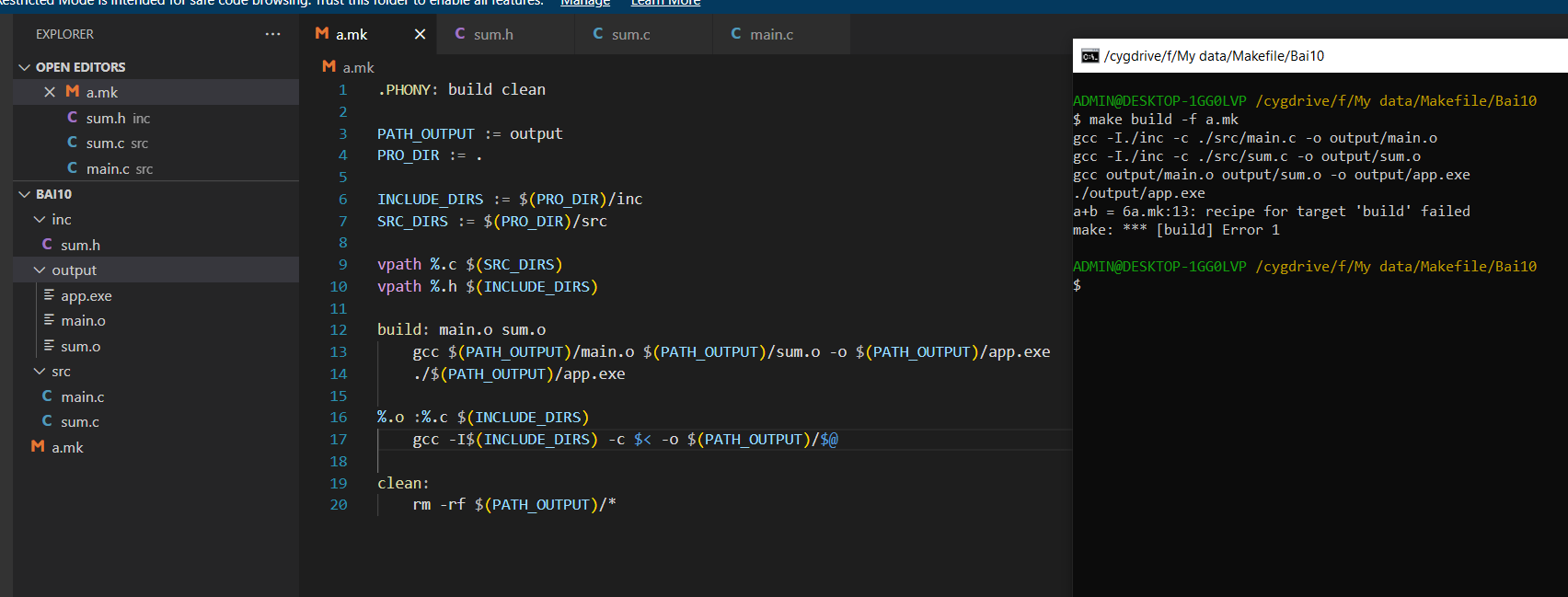




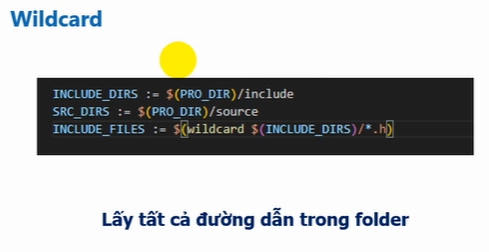
1. **Vpath trong makefile**



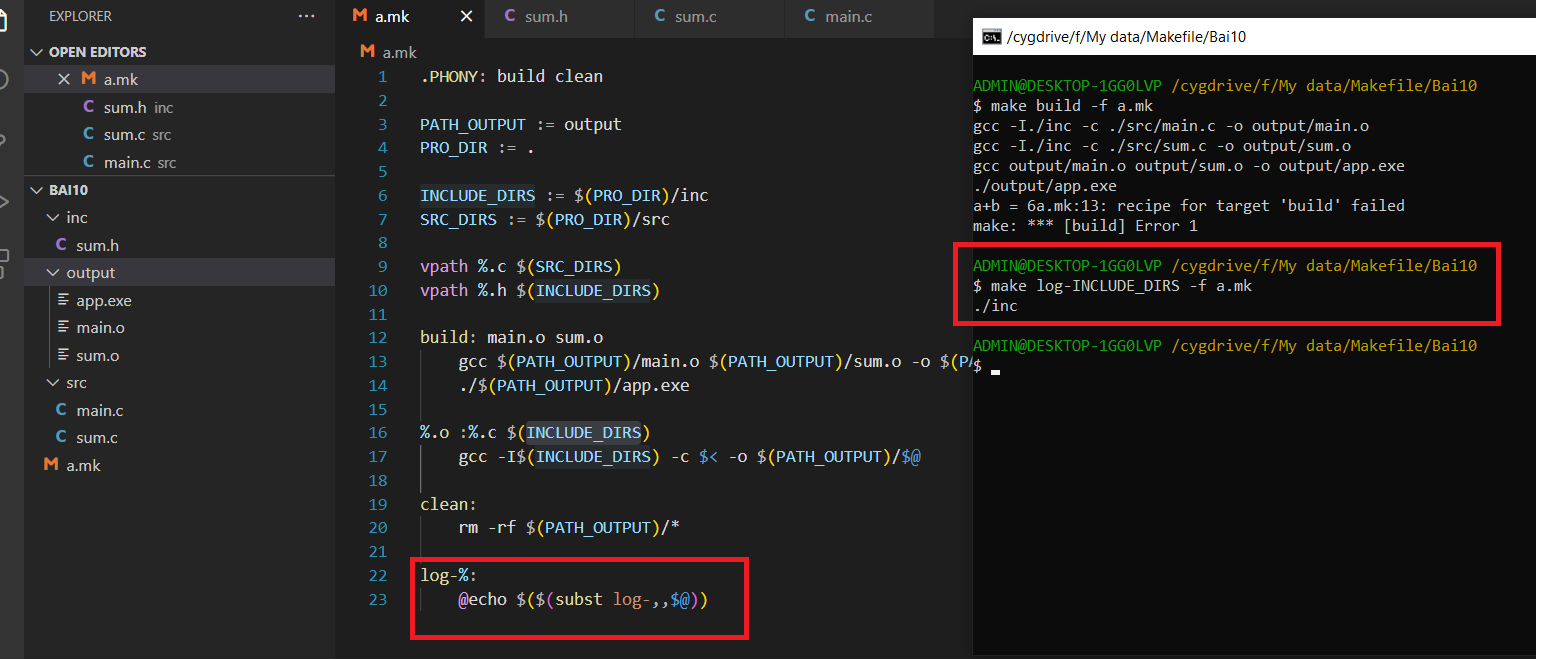
Vpath: tìm tất cả các file trong folder source và include



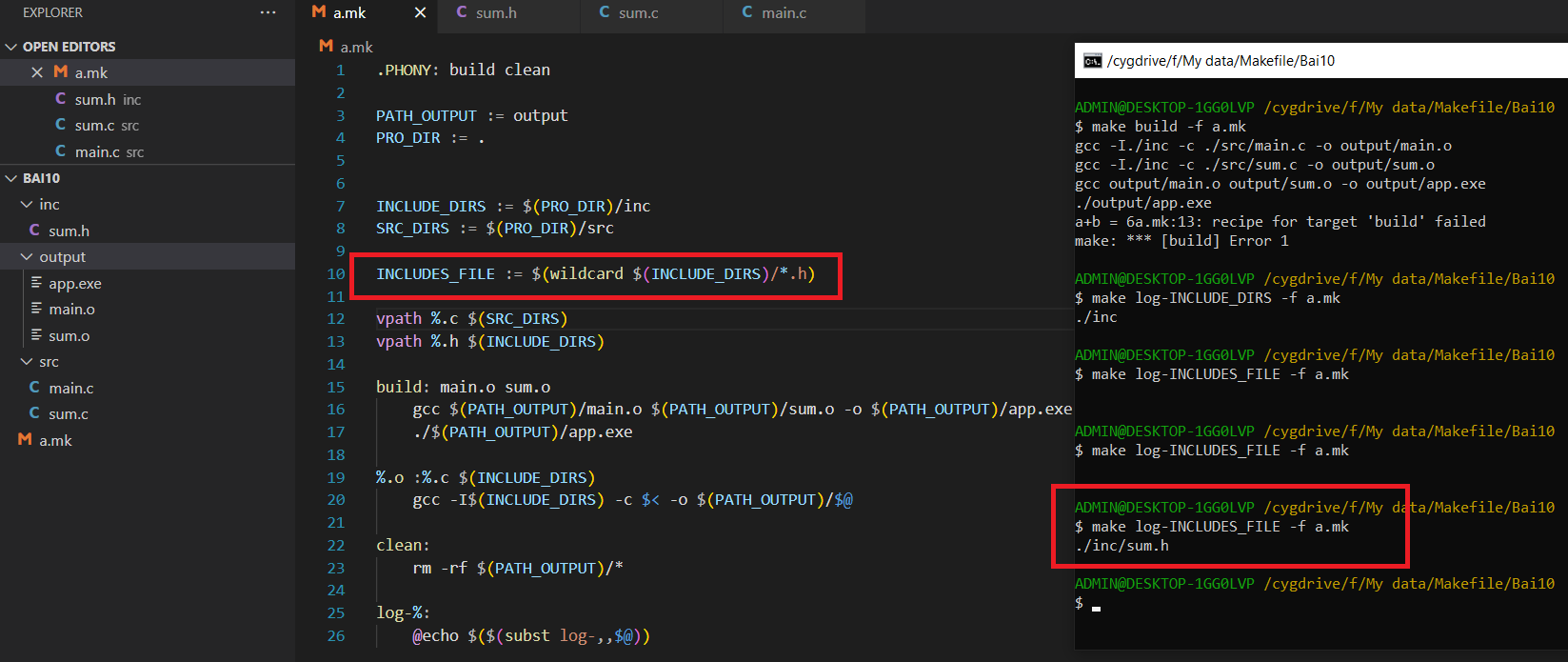
1. **Các lệnh Subst, willcard trong makefile**



* Lệnh subst: In giá trị của 1 biến



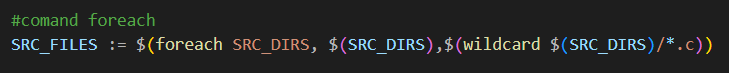
* Lệnh wildcard: lấy tất cả các đường dẫn trong folder



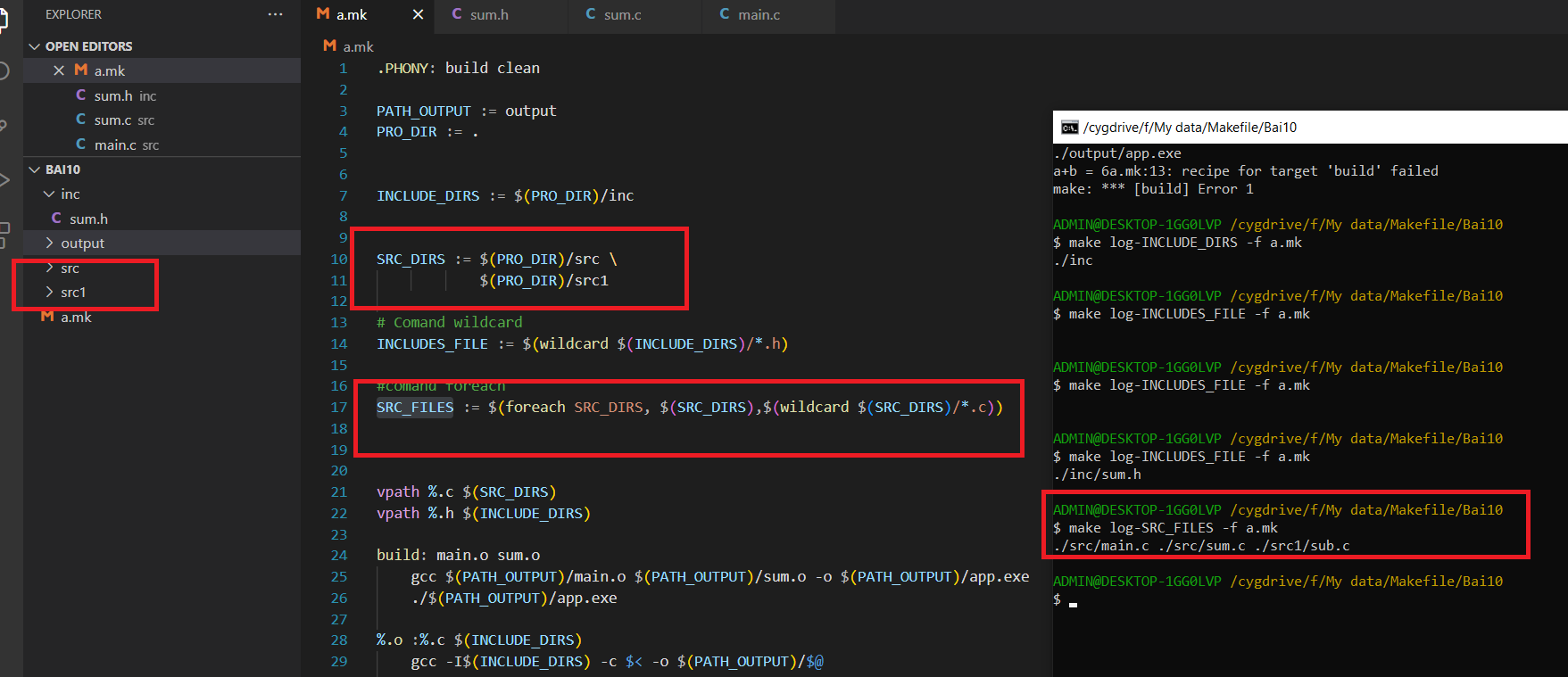
1. **Lệnh foreach trong makefile**

Lệnh forech sẽ duyêt tất cả các file trong nhiều folder như vòng for

Sytax:

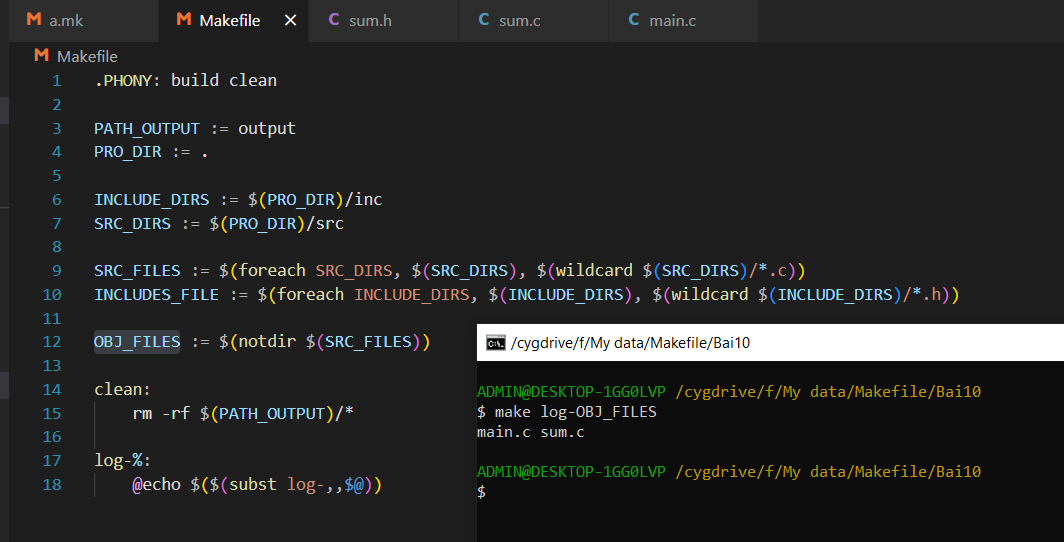


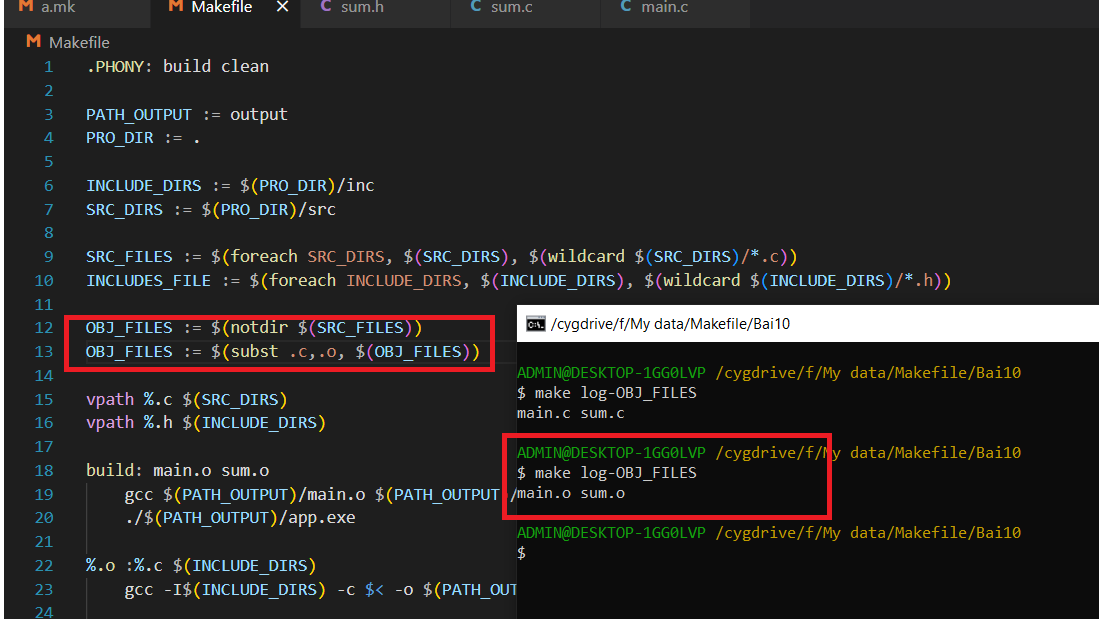
Ví dụ:

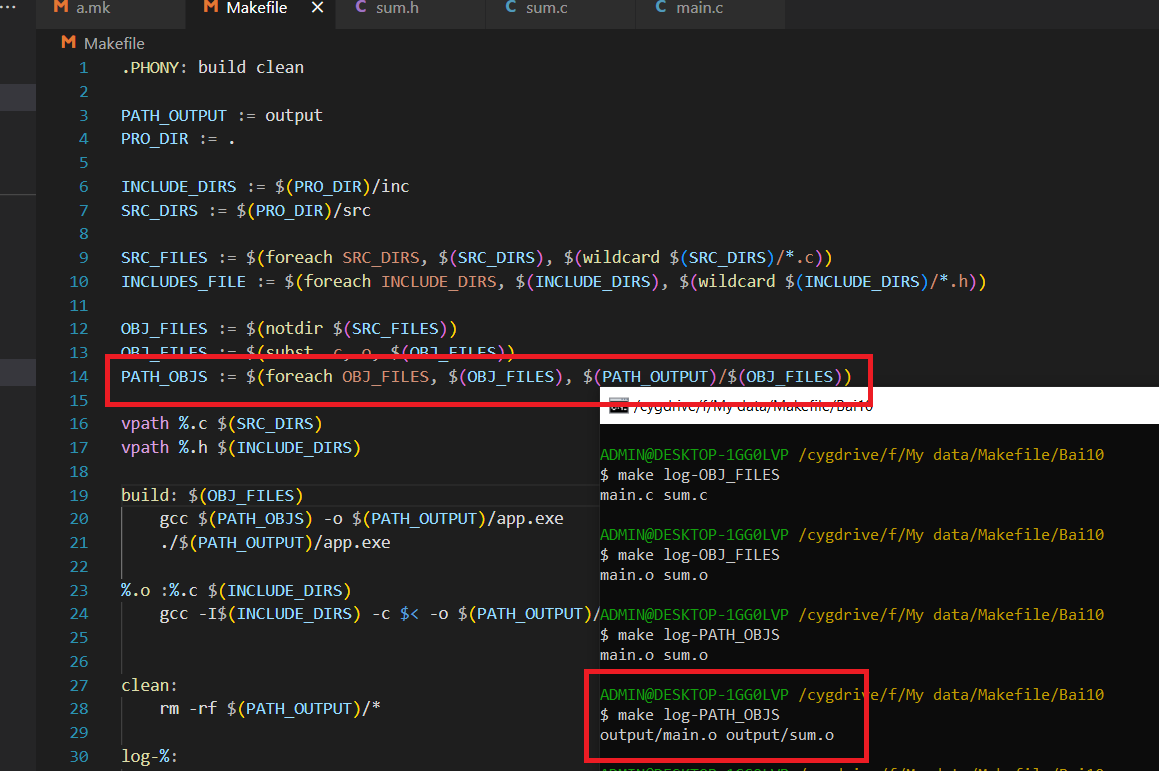


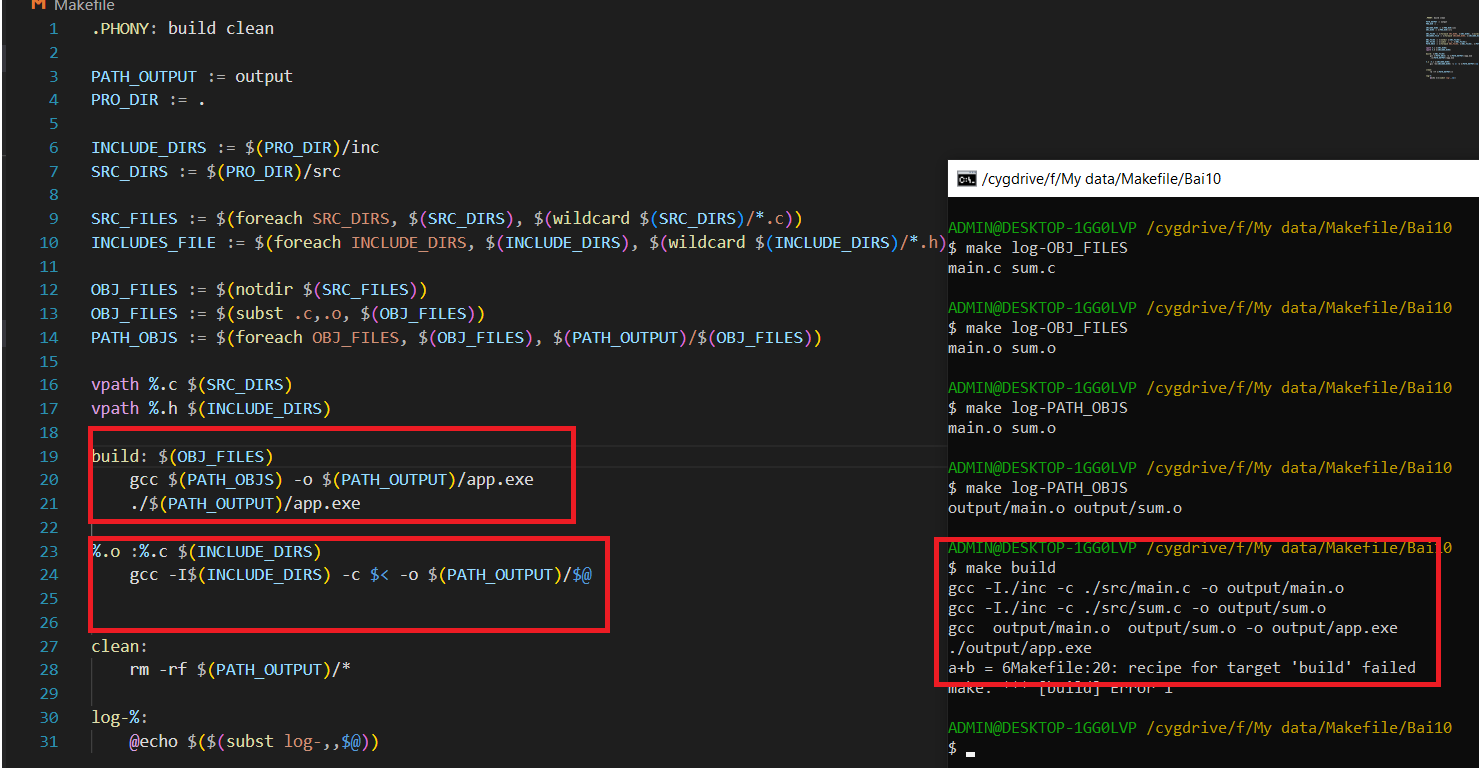
1. **Lệnh notdir trong makefile**

Lệnh notdir nó chỉ lấy tên file không lấy đường dẫn







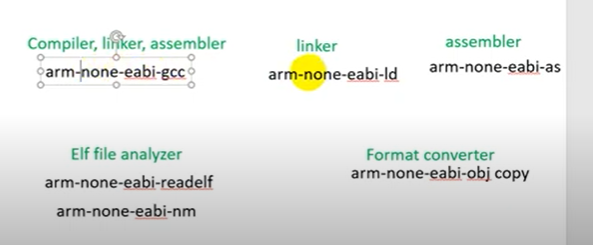


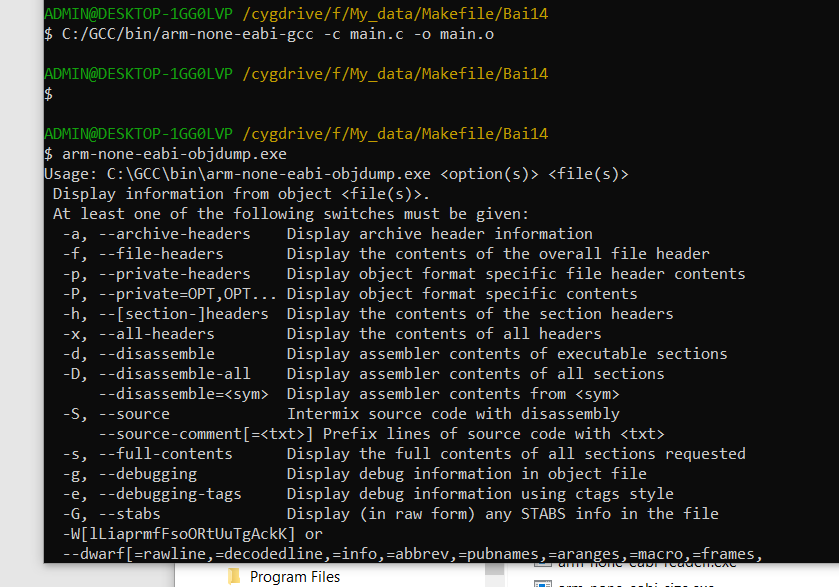
1. **Tạo các thư mục và setup ban đầu:**

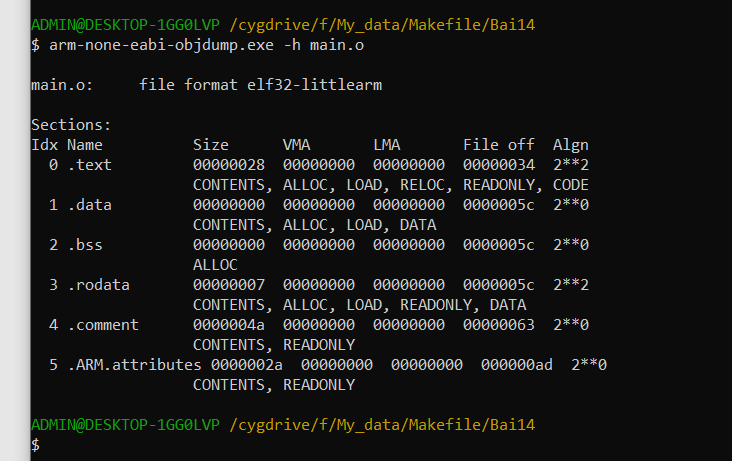
<https://www.hostinger.vn/huong-dan/cac-lenh-co-ban-trong-linux>

LINKER FILE AND START UP FILE

1. Các vùng trong file o và lưu các kiểu dữ liệu trong FLASH linker và startup file







Một chương trình : có hai phần là code và data

+ Code: sẽ được lưu trữ trong .text trong FLASH (flash chi de ghi)

+ Data sẽ được lưu trong FLASH hoặc trog RAM (doc, ghi)

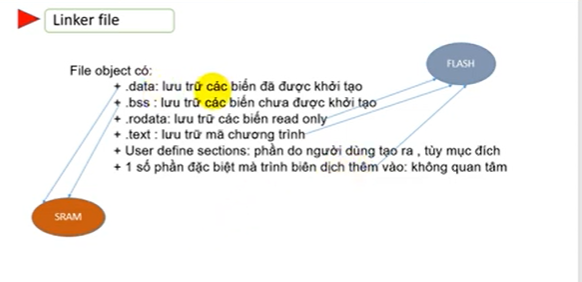
Những biến là const thì lưu trữ trog FLASH vì lưu trữ trog RAM không có ý nghĩa

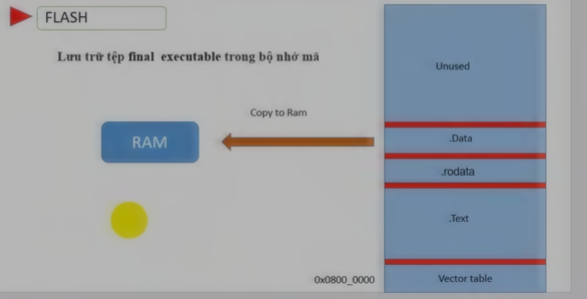
FLASH là bộ nhớ chỉ ghi

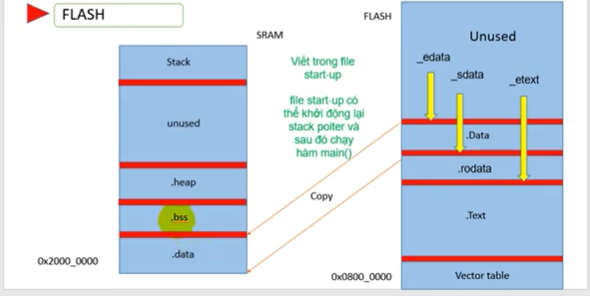
RAM là đọc ghi

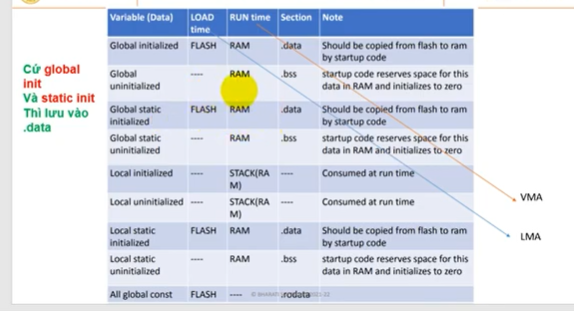
+.rodata: không lưu trữ trong RAM vì rodata lưu trữ hằng số của chương trình

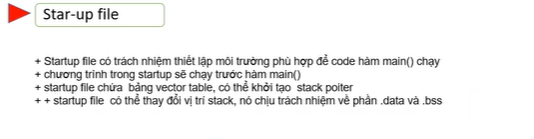
+ Các biến bthuong lưu trog bộ nhớ





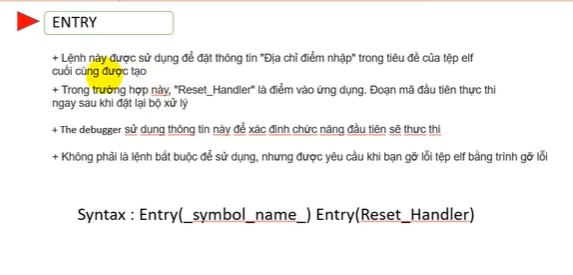


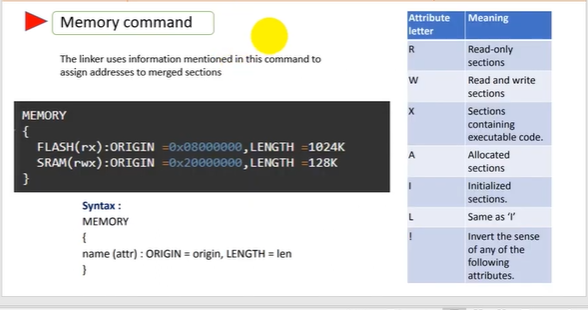


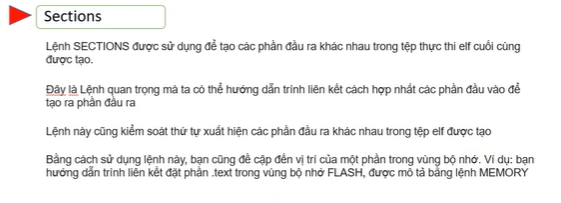


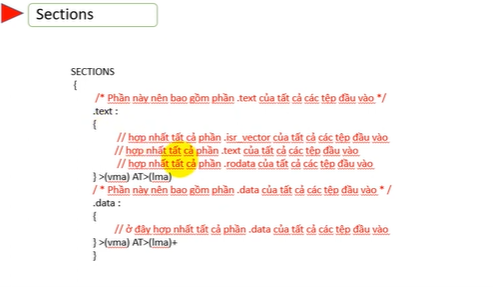


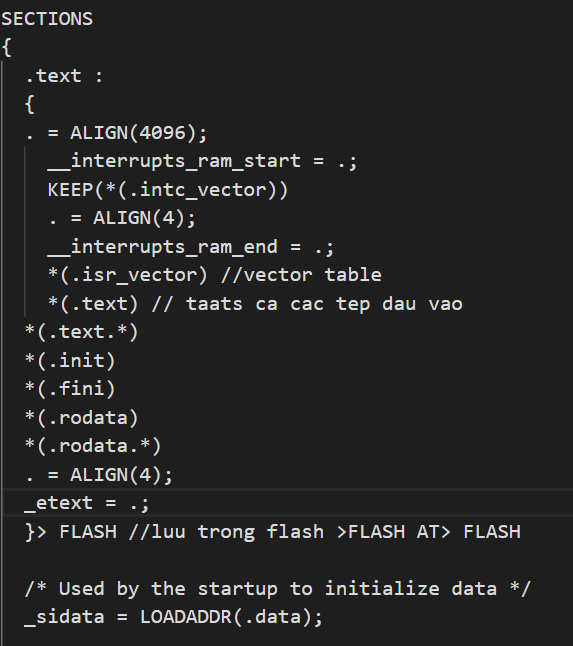


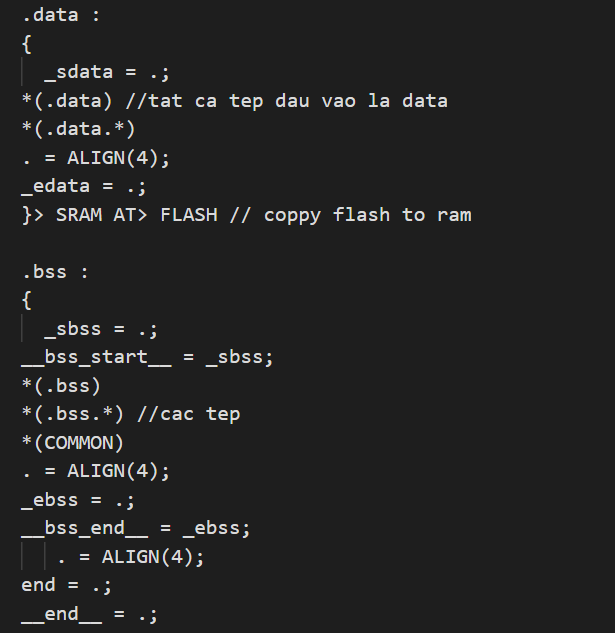


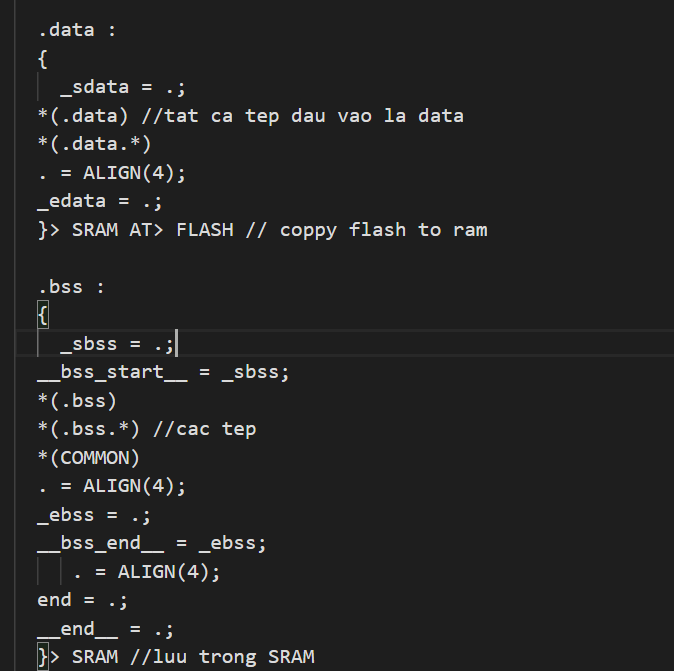


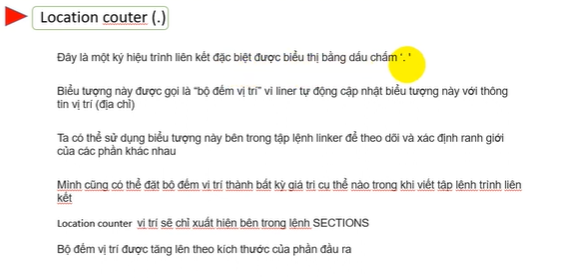


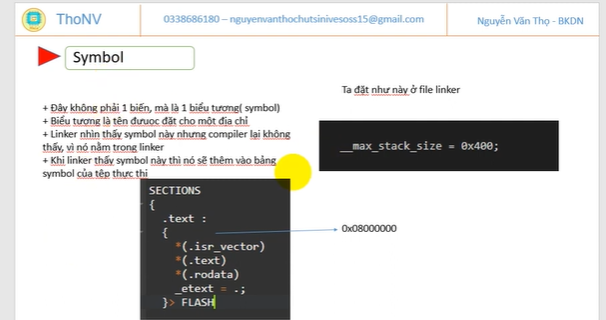


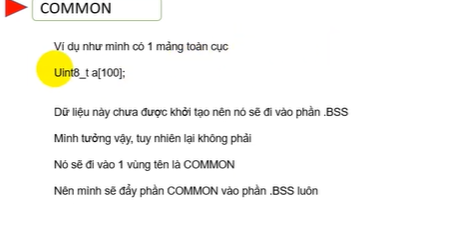


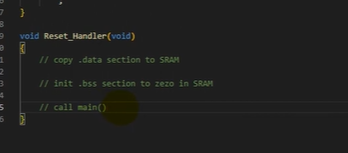






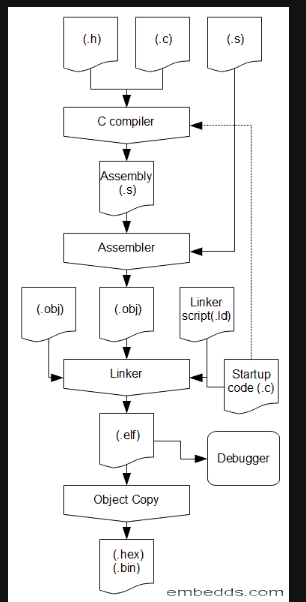






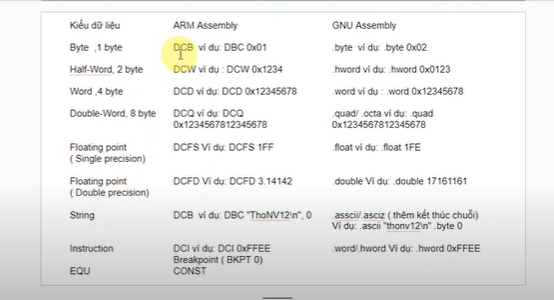
<https://embedds.com/programming-stm32-discovery-using-gnu-tools-linker-script/>

<https://gcc.gnu.org/onlinedocs/gcc/Common-Variable-Attributes.html>



ASSEMBLY (KeilC)

1. Kiểu dữ liệu, các keyword hay gặp và so sánh GCC và KeilC

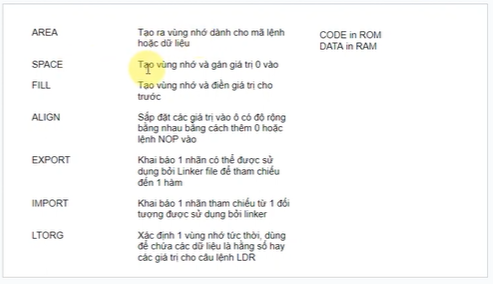


Khai báo biến:

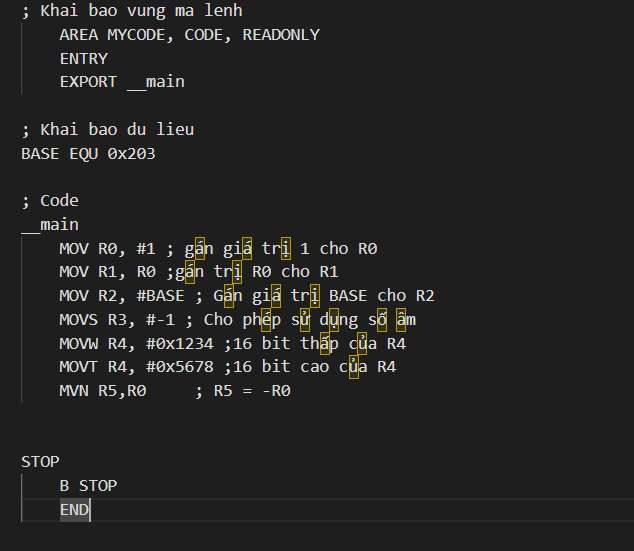
Tên biên kiểu dữ liệu giá trị

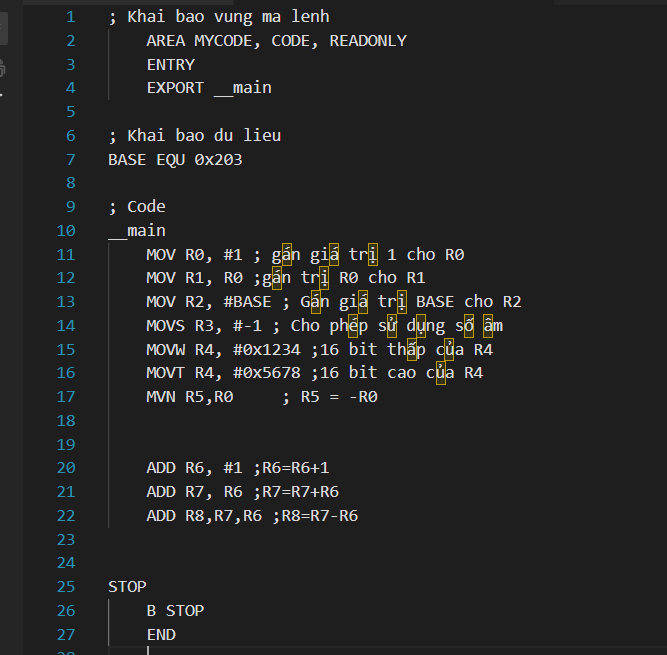
Ví dụ: Duc DCD 100

DCD 0x12345678

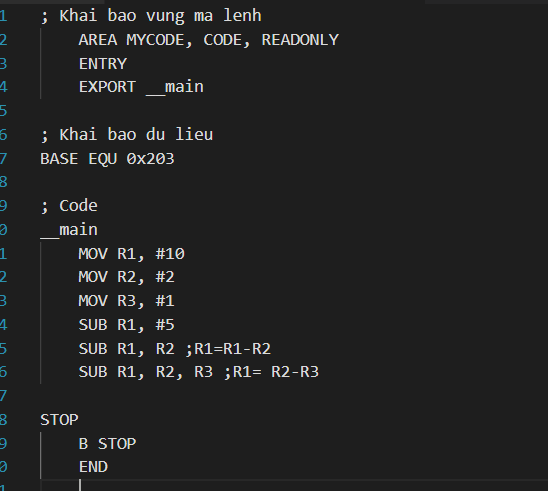


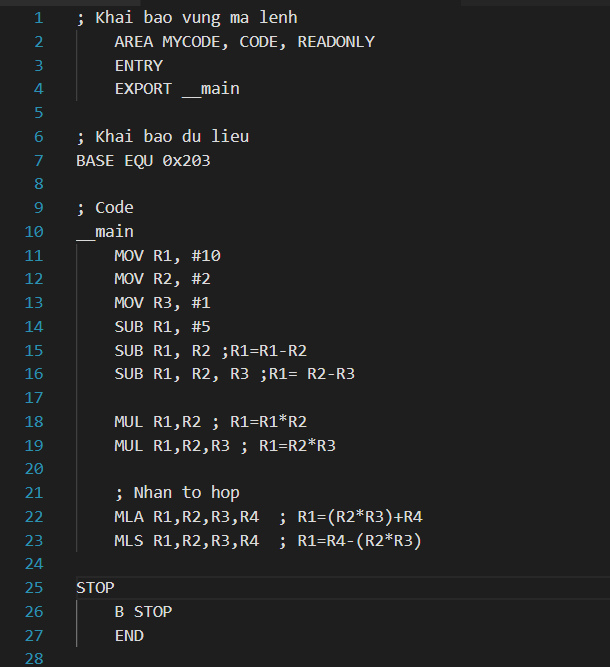
1. Lệnh Move và Add trong assembly



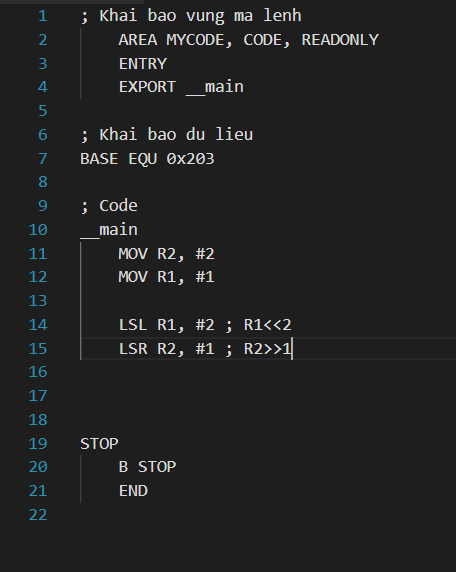


1. Khai báo biến , Trừ giá trị SUB, Nhân tổ hợp MLA MLS RSB, Nhân giá trị MUL

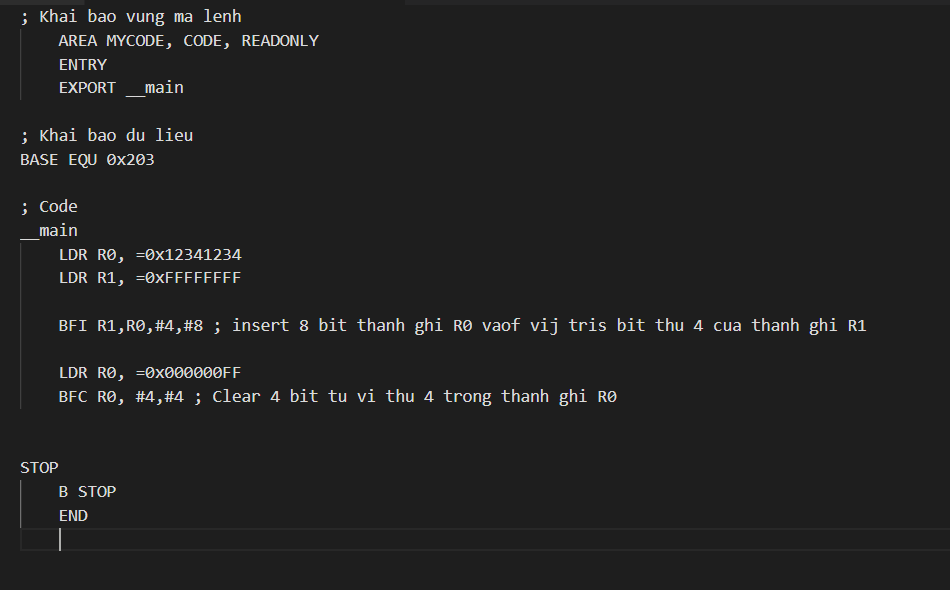




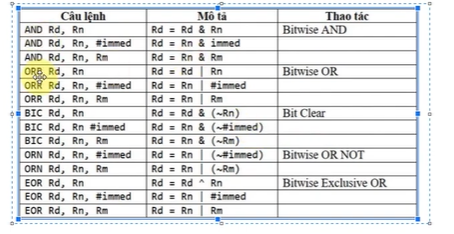
1. Lệnh dịch bit trái, dịch bit phải LSL LSR, Lệnh ROTATE ROR

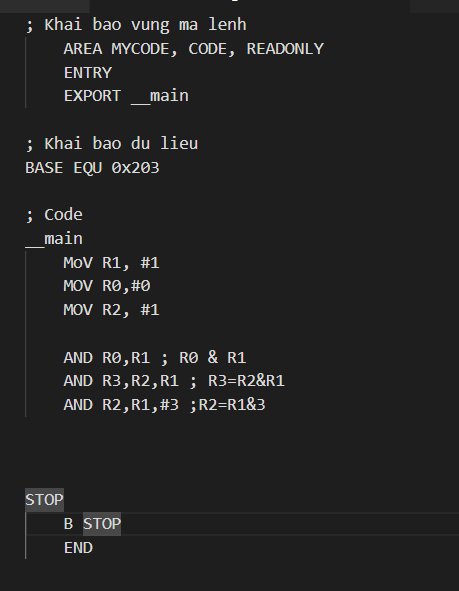


1. Clear bit field và insert bit field



1. AND, ORR, BIC, ORN, EOR bit, Clear bit, Xoá bit



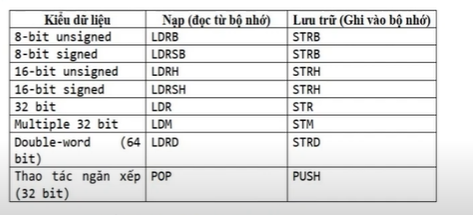


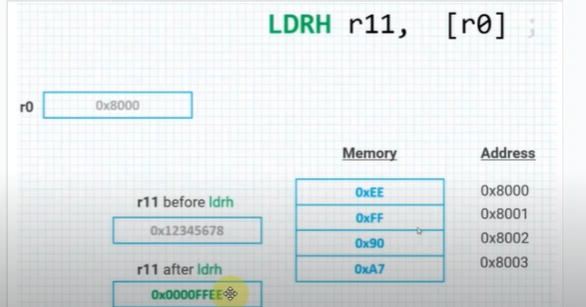
1. Load LDR và Store STR trong ASEMBLY

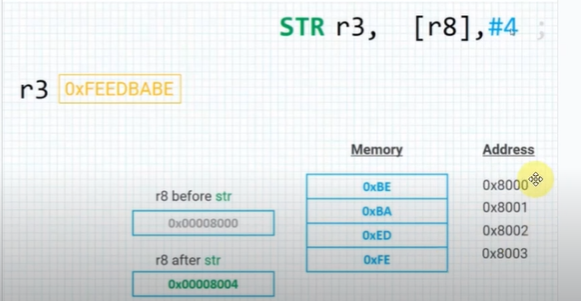


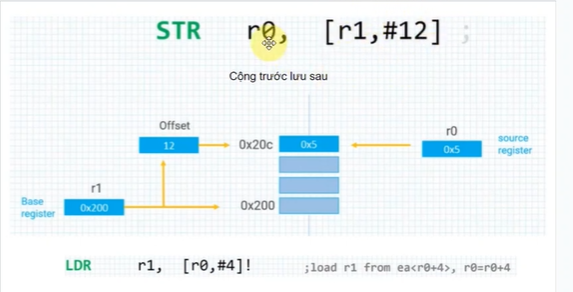
Load: laays value từ memory tới thanh ghi (32bit)

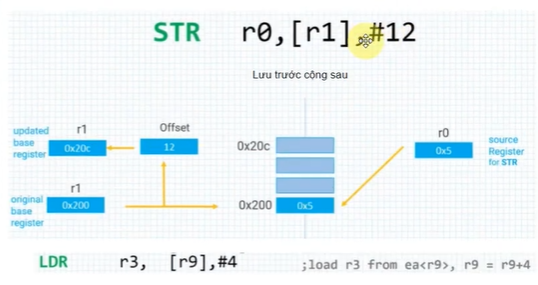
Store: Đọc value từ thanh ghi và ghi vào bộ nhớ(32bit)

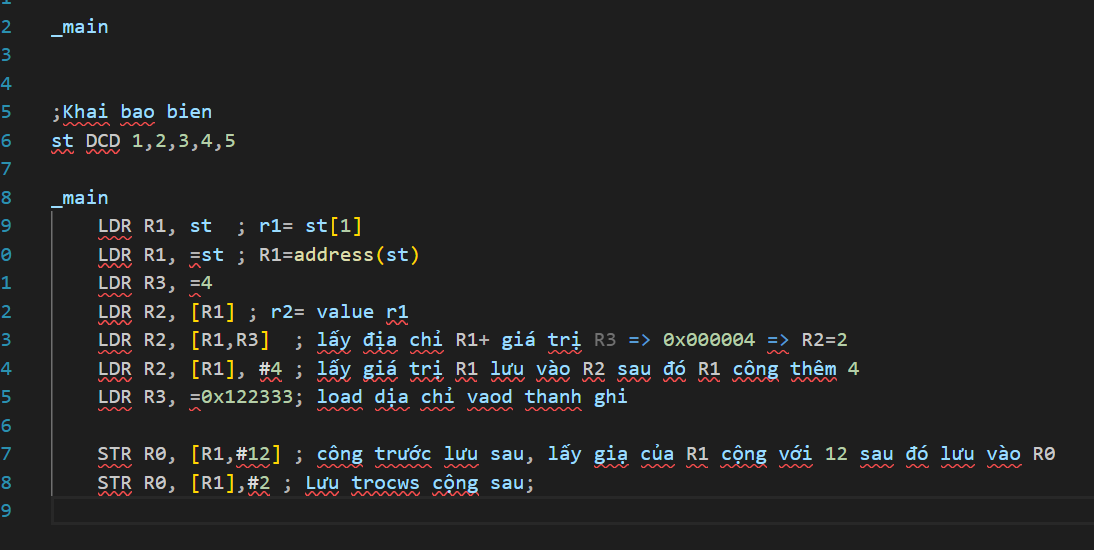




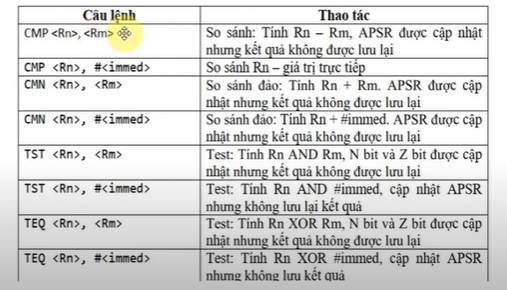


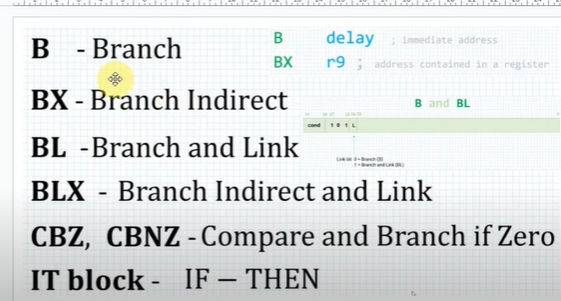


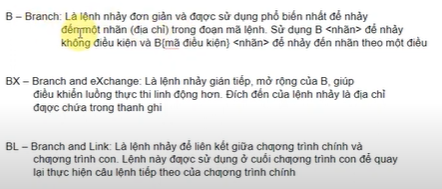


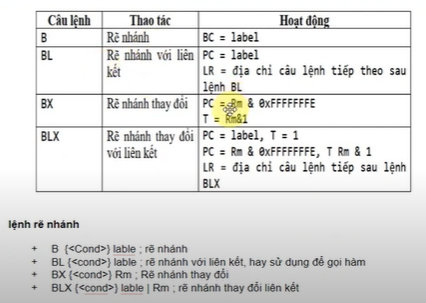


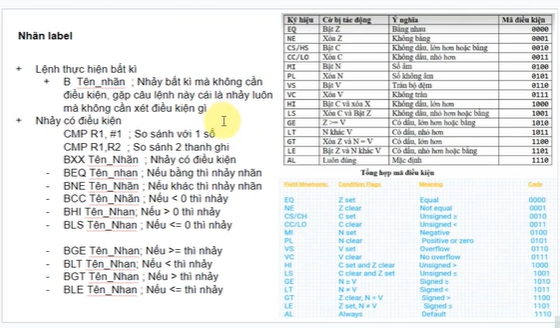
1. Compare CMP, Label, Branch B, Branch And Link BL,Branch eXchange BX

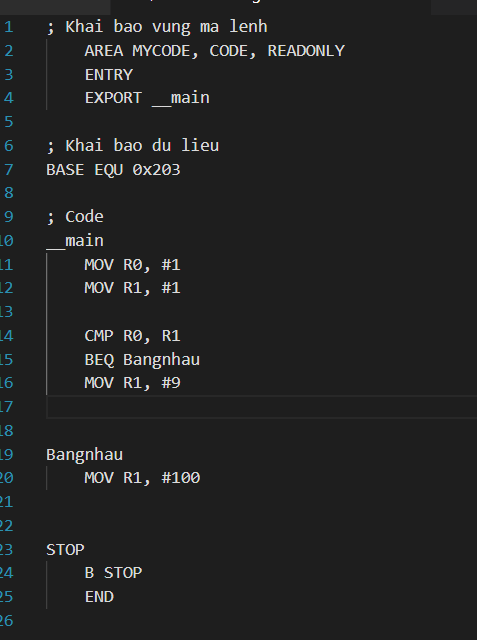


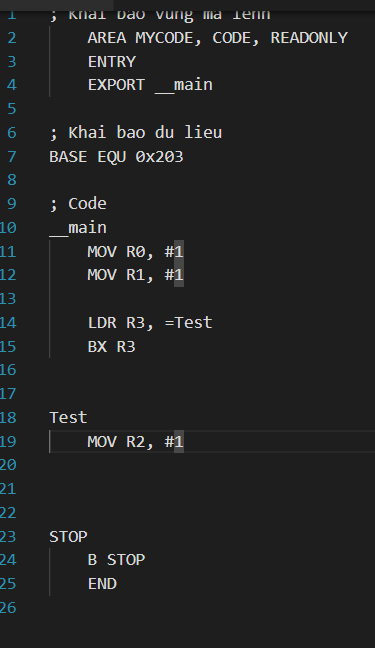




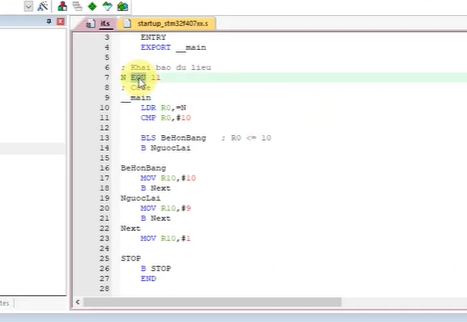








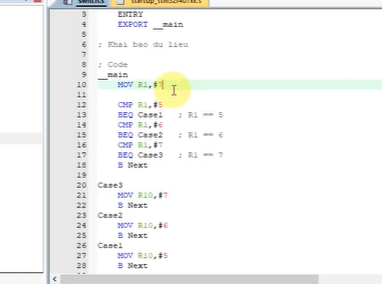
1. Lệnh If trong ASSEMBLY



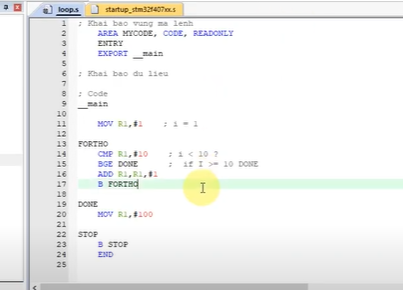
Khi khai báo kiểu EQU thì nó sẽ lưu giá trị vào thanh ghi, thì phải làm như dưới với biến bth



1. Lệnh Switch Case trong assembly



1. Lệnh For trong Assembly



1. Nhóm lệnh điều kiện và các cờ Z,N,V trong assembly

