

Lec 8

Compilation Process (Build Process)

Compiler نمذج

Low Level Languages ↳ High Level Languages محو

HLL C → LLL

Cross Compilers VS Native Compilers

Native Compiler

- Compilers that generates code for the same "Platform" on which it runs.

- It converts the high level language into computer's executable format.

- The "Code Generation/Compilation" and "Running the executable" happen on the same platform.

Example → Turbo C or GCC compiler

غير executable على الـ Machine (ماشين) لكن يحوله Compiler إلى Machine (ماشين) على

Tool chain

Tool chain compiler will exactables files to target machine to run at target machine using HLL like C/C++

Cross Compiler

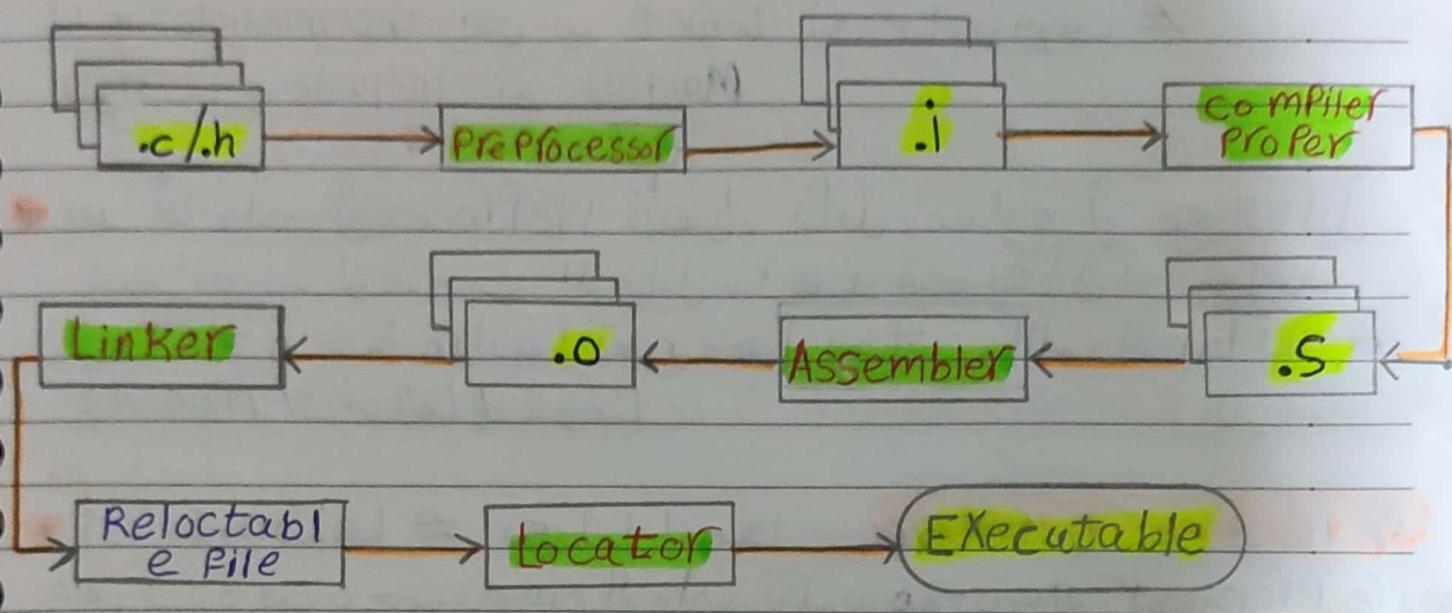
- Compiler that generates executable code for a platform other than one on which the compiler is running.
- Example: GCC compiler for ARM Embedded Processors (GNU ARM Embedded Toolchain), Micro C
- The output executable will run into "ARM based Mac"

أيضاً executable PC will be compiler to target cross to target platform MCU which will run on target machine target <--> MCU (Host or cross computer) machine will be built and run target native code will be built

MCU لـ memory (Flash memory) cross compiler toolchain

Flash memory is end address and start address of target MCU memory map (Flash memory)

Build Process "compilation Process" Is a linear Process



الذواهر البرائينها execute .c , .h files ميخرج Pre Processor all sourcefile #define #include < #PreProcessor الخارجه Post Processed لـ Post Processed File والذواهر دى تكون لها الأقل

intermediate file

Pre Process file لـ Post Processed



عادةً يكون دى ما تكون

1- Pre Processing

لو ناين اعدها بنفسه بـ cmd اكتب بـ cmd اكتب اعدها الفايل c . وادوس cmd اكتب في cmd اكتب في cmd وادوس cmd اكتب في cmd وادوس cmd اكتب في cmd

CPP

app.c

> app.i

اسم الفايل

Pre Processor

→ CPP.exe (Toolchain) لـ

CPP.exe

دار Pre Processing يعني بيعمل على الفايل من source file يعني Pre Processing يعني Target أي حاجة فيها #include أو Macros أو Include لسوادكانت.

بعد أتمتة Pre Processing فهذا فايل يامتدده أن مع الفايل main.c. لوقتها دا هدفيه شئان أي حاجة فيها #include وحط مكانها تعريفها بجانب اللود دا أنا نتابه في الـ main.c أو في الفايل عومنا.

Pre Processor Directive أي حاجة قلناها #define دى كلها اسمها Pre Processor Directive وهو عبارة عن مجموعة من الأوامر يتكون موجهها فهم

#include <stdio.h>

معناها تختت أو تهات محتوى الفايل دا copy وحطه في الفايل الأذن فيه

#include <CPP.h>

#include "CPP.h"

دعا عبارة عن فايل الـ User هو العامله ستان كدا استدللت " "

#include <stdio.h>

إنها عبارة عن library ظاهره الـ User دعمها في حاجة

الـ Compiler يارف الفرق بين "" < > بس يتفاعل " " library يور عليه الأول خلف الفايل C. الأقماص موجود فيه لوعدهم هيروح يوم يوم ملفات Files Library والعمليه عايسه لاحضر الفايل بـ < > هيروح الأول خلف

لوكيز اعرف `<stdio.h>`

1- يقع على فولدر الـ compiler

`X86-64-W64-mingw32` ← Folder -فتح

`include` ← Folder -فتح

4- يدمر كل الـ header files وكل فайл جواه هو يصل إليه وتعريفه

لوكيز اعرف `"app.h"`

1- عادةً مكتن تلقيه مع `.c` ولو لم يوجد مع `.C` ← File ← C ويفتحه المبرمج بنفس المفهوم same way

2- لو كنت موجود مع نفس الفايل `C` وفوفولدر تأكى وعانياز اعلم `Path` بداعي المفهوم same way

3- لما يعمل على إيه الـ Pre Processing على الفايل `app.h` يأخذ كل حاجته موجودة في فايل `h` وينظر لها في `o`. أي حاجة موجودة في `h` يأخذها copy لها في `o`.

الـ Pre Processor time

هو الوقت الـ ياخذه المفهوم لـ Pre Process `h` أو `c`. إلى `o`.

Remove comments \leftarrow Pre Processor *
 يعمل Pre Processor على حذف comments أو comments
 multi أو single

2-compilation

المرحلة الثانية تحويل الملفات الى خارجتين Pre Processor intermediate .S او السبيلي
 يحتوي الكود على لغة الاسمياني

المرحلة الثالثة برمجه بروج على المكان فيه file .S وكتب في cmd \leftarrow cmd وكتب في cmd
 بداعم cmd

gcc -S app.i

ويعتنى بالكتاب كذا ادرس Enter باسم الفايل \leftarrow اسما الفايل
 هناك ملف يسمى S اسم الفايل لو أنشأ فتحت الفايله مادئته
 كتابة لغة الاسمياني

* يعمل كل فايل لوحده في سطر لوحة خالد cmd

3-Assembler

المرحلة الرابعة يتحول الاسمياني كود الى هو بصيغة S. إلى object file \leftarrow بصيغة O.

gcc -O app.o app.s

أعمال البيانات
 الموجورة .exe
 Tool chain
 Machine code
 object file

الcommand line
 command line
 object file

بعد ما تكتب command `ll` لـ list الملفات، يتم تولى ملف بمحضه `o`.
عبارة عن `object file` ويسعى لـ `Machine code`

4- Linker

خلال مرحلة دى يتم تول الملفات إلى بصيغة `o`. يرجعنا `Linker`
وينخرجهما في صيغة ملف `exe`. `executable file`

* عيوب افضل لا `exe`. تكتب `ll` command

`gcc` -o app.exe app.c

Compiler

الفايل المكتوب باسم `app.exe`

الصيغة

executable

أنا لا أحتاجه عادي

* بعد ما تكتب `ll` command `ll` command `ll` command
`app.exe` لـ `app.c`. لا `run` الفايل `c`.

* لوعاير اعرف شئحة `ll` compiler او `ll` `cmd`

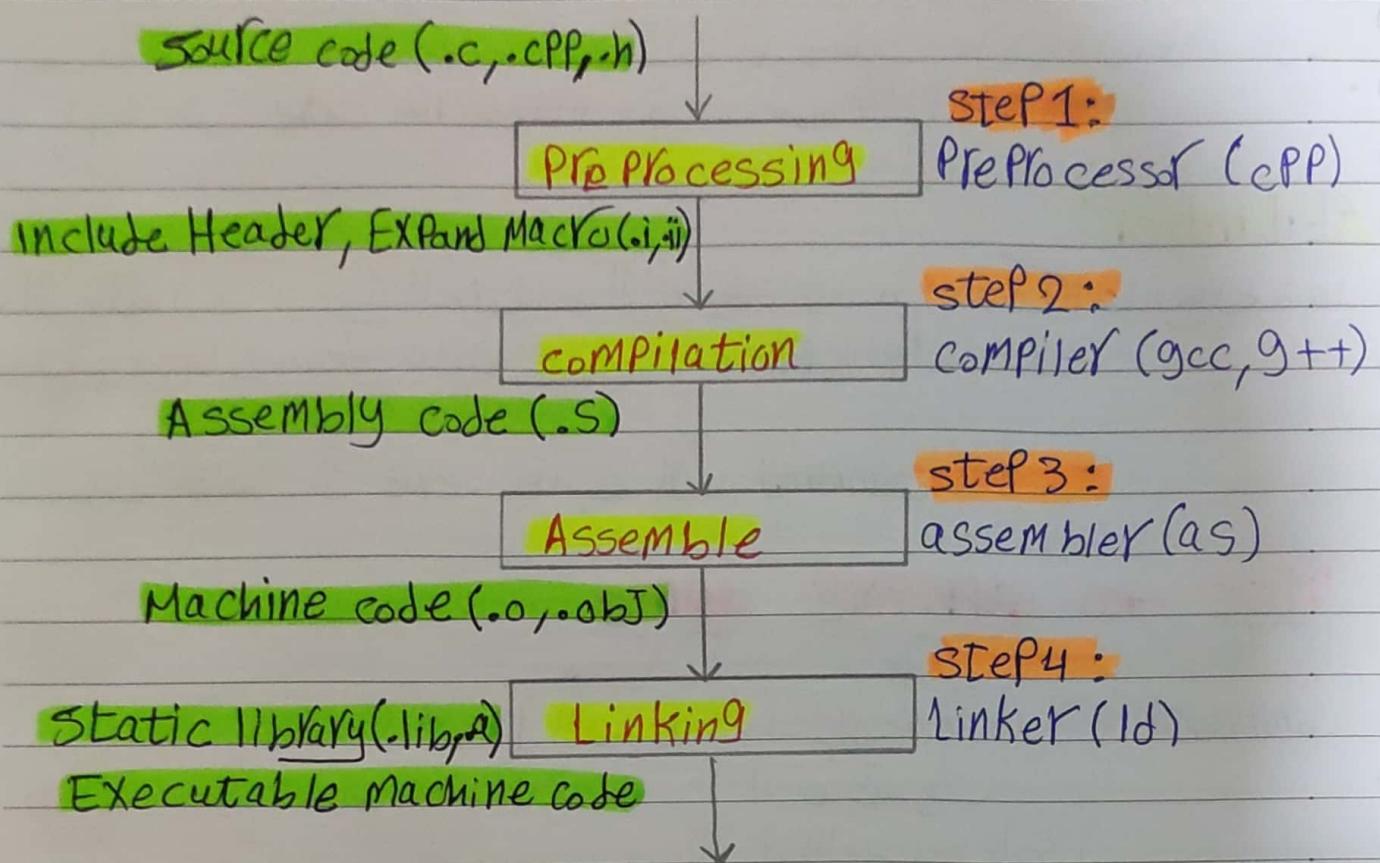
`gcc --version`

* لوعاير مساعدة ممكن تكتب `ll` command `ll` command

`gcc --help`

* لوعاير اقدر فايل `exe`. خلقني الـ `cmd`

اسم الفايل العلمنه
`app.exe`



STRUCTURE OF FILE .h → header file

/* Include section */

• codes.

/* Macros Definition Section */

• codes.

/* User Declaration Types */

• codes.

/* Function Declaration Section */

• codes.

Macros :-

#define PI (3.14189)
 1 space \downarrow symbol
 \downarrow 1 space

* يمكن خط ال () وهم
 تشيها ببس يفضل بذلك تحفظها

UPPER case , - لواسمهين يستخدم

#define macroName value

* ال مايكرو > 1
 Pre Processor
 العن غير مرحلة ال مايكرو
 يسجل ال اسم المايكرو ← MacroName
 ولكن زى مثال PI ← PI في نوع
 ع المايكرو يسجل أى كلمة PI
 ويحظر مثباتها ال Value ← ال بعد المايكرو هى حرف زى A وهو موجود
 فى علامات كى مثل بالعا ومن منطقى أو بع comments
 #define
 Value

#define PI → empty Macro

هيرجع على اللود ويستبدل أى لمحه PI ومن هى جمل طبيعية

array ← size مع #define يتحمل ال boundary variation checking array ←

* يمكن بيقا احداث الع expression ← Value ذى لمحه

#define EXP (3*5+5)

أى كان بها expression الموجود هى جم فى مرحلة ال Pre Processing
 يحصل كل لمحه EXP وخط ال (3*5+5) يعني لو فتحت سطر قابل أه
 (3*5+5) → EXP مكان substitute بـ C . هادفعه على وقارنه بـ C .

Macro with multi Line :-

```
#define TEST    78
TEST
```

Space

} → #define TEST 78

هذا يعمل \ على كل سطر
و ينبع مساواة

* الإنسيين ي يعملون نفس المرض

`#define MY_NAME "Mostafa Alaa"`

هذا عادي لا يوجد هناك خطأ ولا اوجه اميجه هذب كـ امثله

`printf("My name : %s \n ", MY_NAME);`

لـ string عبارات

حاجط الاسم بين " "

(()) بين

Macro ما compiler من هيعتبر دا * MY_NAME لـ أنا حطيت عادي
هيعتبر text

Function Macro , Macro Function, Parameterize :-

`#define SUMING(NUM1, NUM2) (NUM1+NUM2)`

دـى كـا يعنى (فى فـانـشن بـس هـا دـى مـدى خـدـقـع الدـاتـا تـابـ)
المـعـيـرـات وـلـا اـجـهـ استـخـدـمـها (2,3) فـي مـفـرـجـه
(2+3) لـاـتـعـدـهـ مـكـانـهـ SUMING(2,3) غـفـالـاـ . Pre Processor

Macro Functions وظيفة الماكرو Function

int summing (int NUM1 + int NUM2)

{

return (NUM1 +NUM2) ;

}

لما جئت على call لاتكتف بنفسك وضعيه الى ابرو فانكشن طلب

عذرا

printf ("summing = %i \n", summing("2", 3));

printf ("summing = %i \n", SUMMING("2", 3));

ال خطأ Compilation error أو warning يرجع الى summing string اما اتصاص فيها ابرو فانكشن لها parameters او undefined behaviour compilation error

continuation operator

#define SWAP(X,Y)

{

*(X) = *(X) ^ *(Y) ; \

*(Y) = *(X) ^ *(Y) ; \

*(X) = *(X) ^ *(Y) ; \

}

حيثما تكتف فانكشن بتحل
تبديل بين قيمة X, Y بضربيه
X & Y هنفترجها

X = 5 Y = 6.

0101 0110

X = 0101 ^ 0110 = 0011

Y = 0011 ^ 0110 = 0101

X = 0011 ^ 0101 = 0110

X = 0110 = 6 Y = 0101 = 5

SWAP (&X, &Y);

call لاتكتف بالadress

ما يكتفي بالfunction

يفضل أن يكون Macro Multi Line * comments بينها وبين comment وأى حاجة يتغير بعد الـ comment يتغير الكلمة دا لو انا حاطم او comment بعد \ لأنها لو تغيير comment هنأدى وظيفتها

EXAMPLE on Macro Function

ماضية الـ Bitwise operator تابعى تلك معاشرات ١- معادلة لوعاينز اعل لـ Set

$VAR = VAR | (1 << BIT_Pos)$

لوباءز اعماها بالمايكرو ميالكت

#define SET_BIT(VAR,BIT_Pos) (VAR |= (1<<BIT_Pos))

هذا هو ما نشتت بياصى لها العمدة أو المتغير ورقم البت اعماها

٢- معادلة لوعاينز اعل لـ clear

$VAR = VAR & ~(1 << BIT_Pos)$

#define CLEAR_BIT(VAR,BIT_Pos) (VAR &= ~(1<<BIT_Pos))

نفس القاعدة أصلها بياصى المتغير ورقم البت

٣- لوعاينز اعل لـ toggle يعنى لو هوا ٠ بيعطى ١ أو لو هوا ١ بيعطى ٠

$VAR = VAR ^ (1 << BIT_Pos)$

#define TOGGLE(VAR,BIT_Pos) (VAR ^= (1<<BIT_Pos))

لأن Macro Function لها قيم ثابتة تكون Parameter لها ملحوظات مثل المدخلات والخرجات التي تكتب بين brackets أو round brackets أو Parentheses () .
أو أنها تكون ملحوظة داخل Parameter فقط.

Problems with function by Macro?

- 1- Comments
- 2- semi colon
- 3- round brackets

Macros vs Function

Macros :

- 1- Macros are Pre Processed → Processed before your program compiles.
- 2- No type checking (incompatible operand) is done → Errors / side effects in some cases.
- 3- Macros do not check for compilation error.
- 4- Difficult to debug as they cause simple replacement.
- 5- Using macro increases the code length.
- 6- Speed of execution using macro is faster.
- 7- Macro are useful when small code is repeated many times.
- 8- Macro does not check any compile-time errors

Functions :

- 1- Functions are not Pre Processed but "compiled".
- 2- There is a type checking → warning if issue found.
- 3- Check for compilation error.
- 4- Easy to debug and the stack frame located in the stack memory.
- 5- Using function keeps the code length unaffected.
- 6- Speed of executing using functions is slower than macros.
- 7- Function are useful when large code is to be written.
- 8- Functions check compile-time errors

compilation condition Pre Processor directive

#if

#endif if درجة الحرارة

#endif

* يستخدم لبيان شرط وظيفة
عند اعطاء بناءً على الشرط معين
الشرط ما يكتب بحسب ادار

#if (expression or condition)

// code 1

elif (expression or condition)

// code 2

#endif

else if شرط اخر

هنا لو الشرط اذا True ي

فمنها لو code 1

صحيح يشوف الشرط True

او code 2 هو False

شيء بالا If و منها

#endif if درجة الحرارة من اهم حاجة

#if 0

* اى حاجة موجودة بين #if 0 وال

/* */ comment كذا ها هي موجودة في او

// code

#endif

لو حطي اهناك 0 او لا يدخل في code فال الشرط True دائما

#if (condition)

// code

#elif (condition)

// code

#else

// code

else شرط وظيفة

مع او If اذا عادي وكل الشرط

او قبلها او الخطا

او قبلها او الخطا

#ifdef (Macro definition)

endif (درسته) ~~#ifndef~~

#endif

هذا يعمل فإذا كان الماكرو بعد **ifdef** check ما إذا كان الماكرو بعد **#define name Macro** ود **#define name** align **#define out** **Macro** **as** **Value** **alone** **in** **line** حسناً لو منعنا **Value** **alone** **in** **line**

أولاً بعد **True** \rightarrow **if** **LLJ definition** **Lo** **what** **ifdef** **compilation** **after** **it** **use**

File guard \rightarrow header files

if not define

#ifndef symbol \rightarrow **header file**

هذا أنا بقوله لو **symbol**

define out **in** **one** **line**

define out **in** **one** **line**

/*All code in header file here */

Line File guard

لجعل **header file** **include** **from** **one** **file** **only** \leftarrow **define out** **in** **one** **line** \leftarrow **header file** **is** **not** **repeated**

#include < neglect definition > \leftarrow **definition** **and** **declaration** **in** **one** **file**

error **or** **issue** **will** **be** **there**

header file **or** **name** \rightarrow **symbol** \rightarrow **upper case**

أو في **header file** **use** **lower case** **for** **name** \rightarrow **upper case** **for** **symbol**

#ifndef APP_H_

header file **is** **not** **used** **again** **in** **another** **file**

يلعب **header file** **role** **as** **header** **file**

#endif

ما هي الماكروات Pre define macros

ـ يرجع اسم الفайл كنص
ـ يرجع رقم اول LINE الا لست فيه LINE

ـ يرجع التاريخ DATE

ومن غيرهم كثير بل إنـ و هي تمثيل

#ifdef SYMBOL

#ifndef SYMBOL

#endif

#endif

ـ هذا يحول symbol الى صورة
ـ True لـ define او محوه define
ـ ينزل الى code

ـ هذا يقول هل الا symbol هو من define او محوه
ـ True لـ define او محوه define
ـ ينزل الى code

#ifdef TEST

#error "TEST IS defined"

#endif

ـ define الى صورة TEST وهو *

ـ واظهر في الاسطرة *

Test IS defined

ـ طبع او مسح محوه define الى صورة *

ـ (النحو الاخير ولا يتحقق اي error)

لو اتفاج وصل لها وعما execute له خطأ ~~error~~ باخر تنفي البرنامج عسان كما تتحقق معاشرة الفراغ لوحدها

*~~#line new-line-number new-filename~~

* استخدم لو أردنا عاليز غير الـ line لا يعيشه هناك أو هذب فيه
أو عاليزه يظهر) ومهكم كمان اعدل في اسم الملف بعد ما
بس هذبته بين (" عياد نص

*~~#line 33 "APPLICATION.C"~~

*~~#line --FILE-- --LINE--~~ لو جست الموج

printf("%s - %i \n", --FILE--, --LINE--);

output → APPLICATION.C - 33

* Pragma

Compiler depended

اعنى بختلف وظيفتها من compiler depended level لـ application

* Pragma في لها الفرق بين الموجات compiler's

* Pragma once

*~~ifndef filename~~

*~~define filename~~

.code.

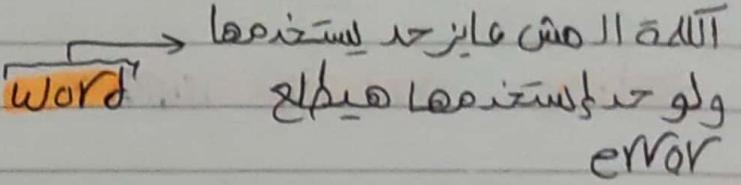
*~~endif~~

fileguard يتحمل نفس وظيفة

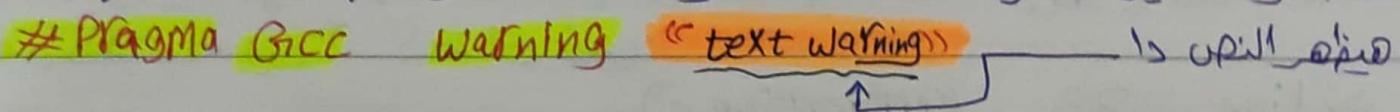
*~~ifndef~~ ليك استخدم الـ

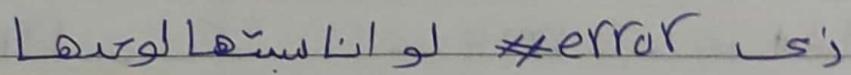
* لوندرى identifier وعندما يغيرها أو يغير user

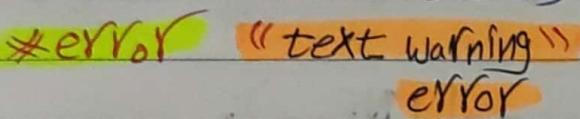
* يستعمل **#PRAGMA** عند اغلاقها

* **PRAGMA GCC Poison word** 

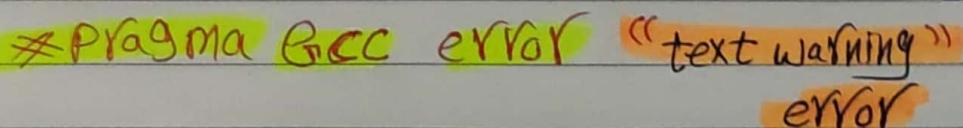
* لوندرى warning يعنى نوعاً من رسالة السطر

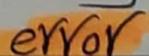
* **PRAGMA GCC Warning** 



* **error** 

* لوندرى **error**

* **PRAGMA GCC error** 



لهم بعض الديسخ إمارات الأخرى تعرف عليها بجزء

PRE Processor operators :-

defined()

يسأل Macro إذا كان أو لا check Macro لـ defined
 Macro لا يوجد ولو $\#define$ defined
 لا يوجد Macro $\#define$ defined
 $\#define$ ولو مخضي True يترجع
 False يترجع

Example:-

$\#if$ defined (Macro name)

• Codes • → هل هناك Macro

$\#endif$ Macro لا موجود

$\#define$ HELLO

$\#if$ defined(HELLO)

Void fun-dec-1(Void);

define لـ HELLO هي معرفة

هذاMacro موجود لـ fun-dec-1

فهي تؤدي وحة خط

Void fun-dec-1(Void);

$\#else$

Void fun-dec-2(Void);

$\#endif$

$\#if$!defined(HELLO1) & & defined(HELLO2)

define لـ HELLO2 و define لـ HELLO1 فيكون المخرج

True بـ الخط

printf("Hi ", "iam ahmed\n");

* هنا عادي هيكت اطبع لوحده string \hookleftarrow 2 string
output \rightarrow Hi , i am ahmed

Macros ممكن اعمل حماجزي لك ابا استخدام الـ

#define PRINT_NAME(F-N, L-N) printf(#F-N " " #L-N \n)

Stringize operator

function by Macro ياخذ الـ Parameter

وتحولها لـ

PRINT_NAME(Mohamed, Ahmed);

output \rightarrow Mohamed Ahmed

PRINT_NAME("Mohamed", "Ahmed");

* لو أنا عملت حماجزي بالشكل دا أو compiler هيأخذ الـ string بار (" ") وتحلها
أو يطبقهم بار (" ") لو فتحت كتابه زى هادئها بالشكل دا
printf ("\" Mohamed ", "\" Ahmed ");
فهم ذلك غير راجح !! فقط قبل ("")

* لوحظت \ قبل string \hookrightarrow string
* لوحظت " " هيطبقها على مع الـ string
فضن الحروف تتفق إن أطبعها