

Compilation process:

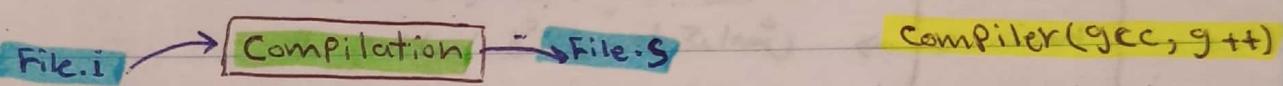
input → source Code (.c .h)

output → include header , expand Macro (.i , .ii)

في المرحلة دي بنتا هندخل الكور اللي احنا كتبناه

دبيطلع لينا ملف (.i) الملف دا بيكون خالص من أي حماقة

#include " " او # Macros



input → include header, expand Macro (.i , .ii)

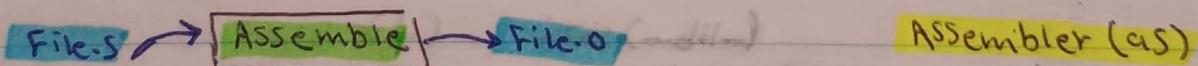
output → Assembly Code (.s)

في المرحلة دي بنتا هندخل الكور [بس الكور هنا هيكون خالص من أي

حماية بارتفع #] على الـ Compiler وهيتم تحويل الكود لـ لغة

يكتفى مثلاً لو في الكور عيننا [If condition] فهو Assembly code

هذا يعني يتحولها لـ (3 instructions)



input → Assembly code (.s)

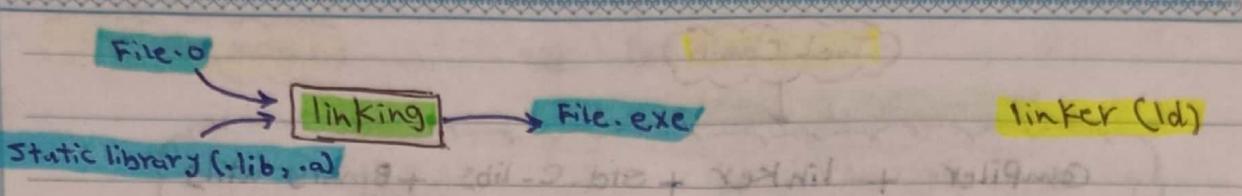
output → Machine code (.o , .obj)

في المرحلة دي هندخل الـ (Assembly code) على الـ (Machine code)

[Machine code] الموجود من الكود إلى الـ [instructions]

الـ [binary] يعني ← (0 أو 1) اللي هيتم حفناه في

[Object file] (asm)



input → Machine Code (.o, .obj) +  
 Static library (.lib, .a)

في المراحل السابقة موجود الـ **defines** بالكتابات الـ **variables**:

الـ **printf**, **scanf** ... إلخ هنـا يستخدمونها في المراحل السابقة

output → Executable Machine Code

(.exe, .hex, .elf, .bin)

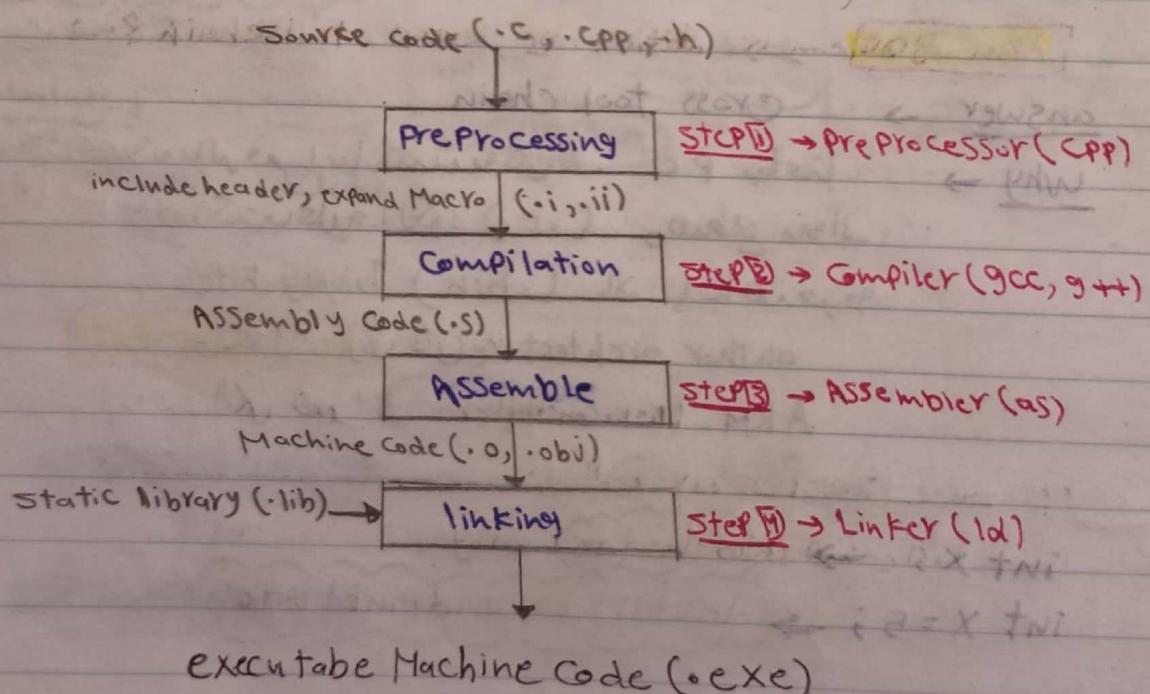
في المرحلة دى (linker) هو بيربط المراحل

الجاهزة عـنـا زىـال [object file] معـالـ [printf]

حـالـةـ وـنـمـكـنـشـ مـوـجـودـ دـوـلـ جـاهـزـ زـىـالـ [printf]

الحالـةـ دـىـ يـعـتـدـ بـمـوـكـبـ المـلـفـاتـ الـمـتـابـعـ [Static library]

هيـطـلـعـ مـعـ الـ linkerـ هـتـبـ حـاجـاتـ الـ object fileـ فـيـمـ



١٠ الفايل المرجود فيه ال main Func

main.c

ال دى أول حاج بتتعدد في ال C

(printf)

التعريف بجا عها متغير من ملف lib.  
ستاخذ أى حد يقدر يستخدمو

مكتوب بالـ (binary)

مينفعش يدخل لو ده على مرحلة

main.o

ال linking لأن لود دخل كده اهش

يعرف المعرفناعي الدوال مثل

(ii) static library: نلاز منحط مكتامع

executable Machine code فعن طريق دمج الـ

main.o

+ .lib

linker

.exe

ي بتعنى مرددة على جزئين (printf) دى Function ال

.lib (.h)

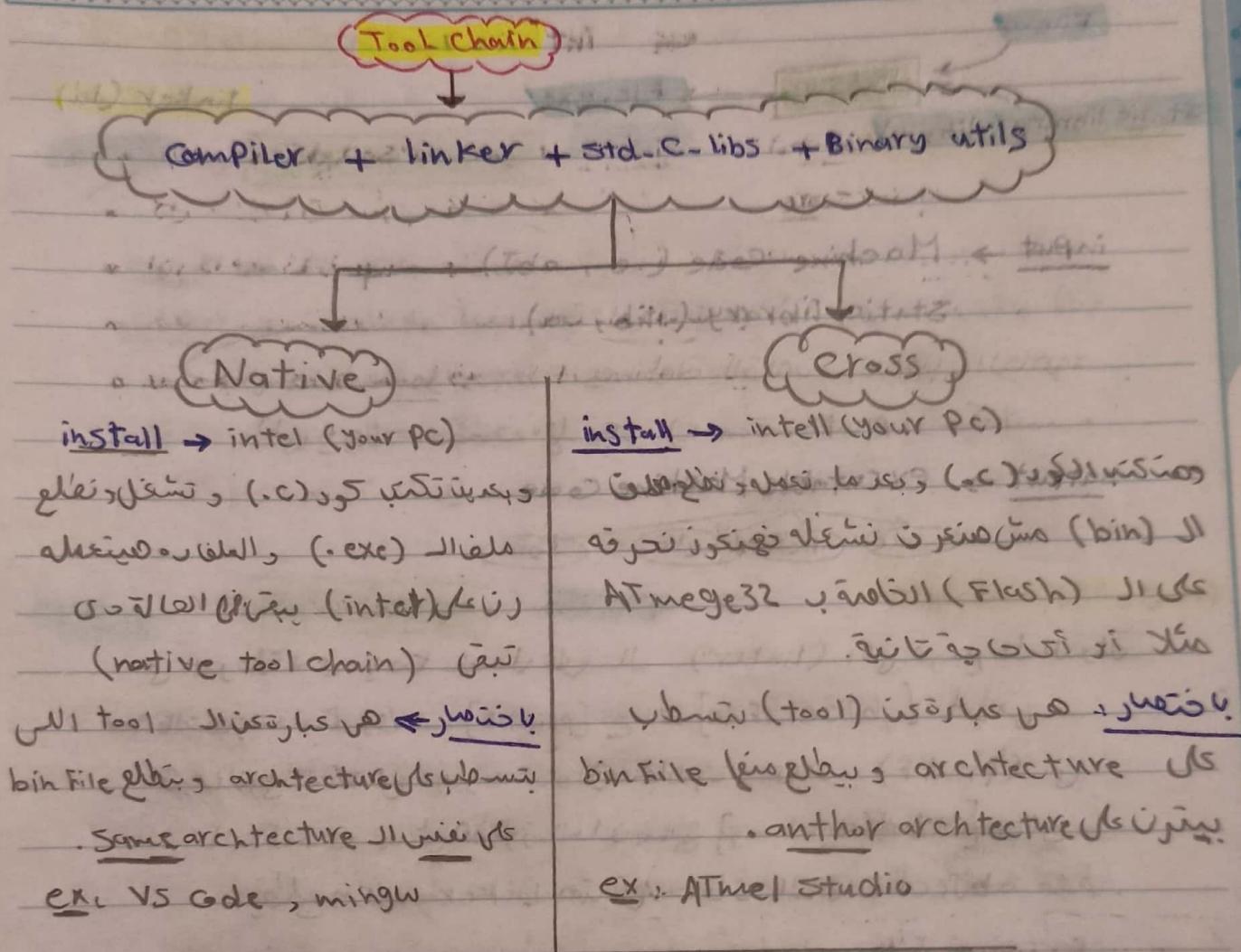
.c (.h)

اما هنا موجود الكور  
لطريقه سعادها

فيه موجود (2)  
رسن تكر يفديهم كذا

أفاد في الجزء الخاص  
بالـ linking

الجزء بيتمان من  
preprocessor عملية ابر



**Arm-gcc** is Native or cross tool chain?

answer → cross tool chain

why → ليس على PC لذا فهو مكتوب عليه الكور (.c) و ينطوي على (.bin) binary File

ونعمله على جهاز ARM-Microcontroller لـ صور الـ

`int x;` → Declaration only

`int x=5;` → Declaration and initiation

**Variable name**

ex int x = 7 ;

شرط كتابة اسم المتغيرات :

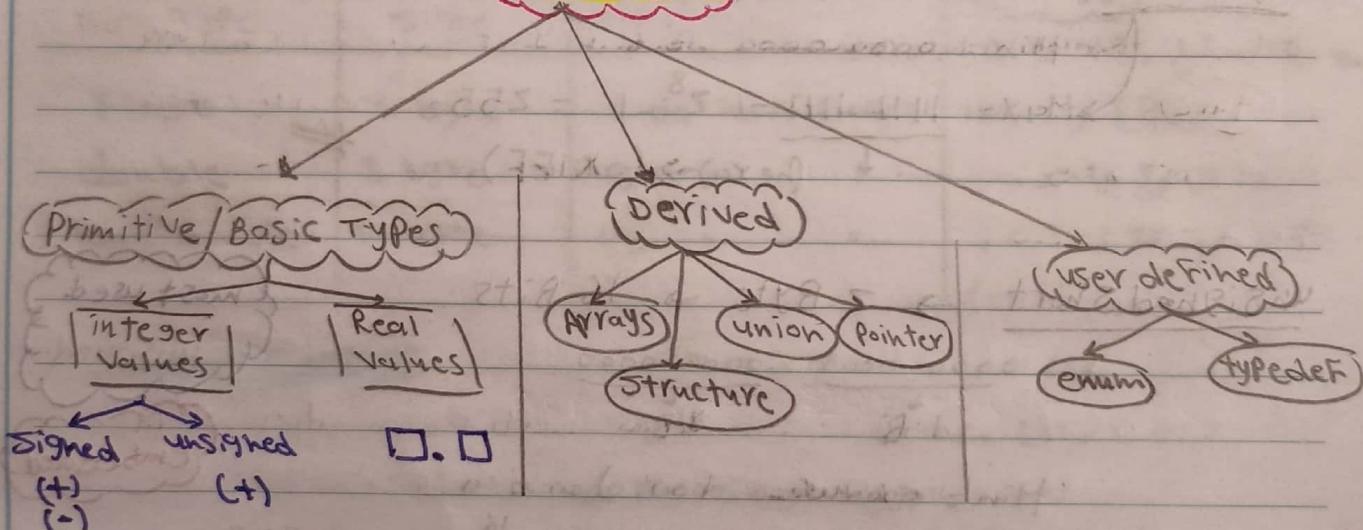
Capital or small كلها مسميات سواء

underline كل الدالقات والدوال مكتوبة بـ ( \_ )

الكلمات المفتاحية في البرمجة كاردي بس مكتوبون ون اسم المتغير يبدأ بـ

استخدام الكلمات المفتاحية في اللغة زر iF, while, ...

وجود مسافة حر ( اسما المتغير )

**Comments**Single line → // You should write // before it.multiple lines → /\* start . . . \*/ \*!**Data types****Integer Values****Range**unsigned integer → 0 to  $(2^S - 1)$ Signed integer →  $- (2^{S-1} + 1)$  to  $(2^{S-1} - 1)$ 

S → size in bits

1 byte = 8 bit

Data type	Major Type	Size(Bytes)	Range
char	Integer	1	-128 → 127
unsigned char	Integer	1	0 → 255
short	Integer	2	-32768 → 32767
unsigned short	Integer	2	0 → 65535
int	Integer	4	3
unsigned int	Integer	4	4
long	Integer	4	5
unsigned long	Integer	4	6
long long		8	7
unsigned long long		8	8

unsigned Char → 1 Byte → 8 Bits

$$\text{Min} = 0000\ 0000 = 0$$

$$\text{Max} = \underbrace{1111\ 1111}_{\downarrow \text{(hexa)}} = 2^8 - 1 = 255 \#$$

unsigned Short → 2 Byte → 16 Bits

$$\begin{array}{c} 0000\ 0000 \quad 0000\ 0000 \\ \downarrow \text{B} \quad \downarrow \text{B} \end{array}$$

$$\text{Min} = 0000\ 0000\ 0000\ 0000 = 0$$

$$\text{Max} = \underbrace{1111\ 1111\ 1111\ 1111}_{\downarrow \text{(hexa)}} = 2^{16} - 1 = 65535 \#$$

most used  
unsigned  
C  
embedded

Char → 1 Byte → 8 Bits

$$\text{Min} = -(2^{8-1}) = -128$$

$$\text{Max} = +[(2^{8-1}) - 1] = +128 - 1 = +127 \#$$

$$\text{Range} \rightarrow -128 \text{ to } 127 \#$$

**note**

**char** → يمثل ترتيب الحروف ورموز ايكار في ASCII كود ينادى لهم ونقدر برصو  
نخزن فيه (أرقام) بس الأرقام محددة تكون في  
الـ range الخامس بار **char** يمكنه من هنا لو نهنا  
أرقام محدودة يفضل إنها تحفظ في **char** حيث  
نقلل المساحة . الـ range (-127 → 128) ← range

**note**

في بعض الـ Compilers يخلو دمج الـ int 2 byte ← int الـ default 4 byte ← int يخليون الـ 1 byte ← int  
يتحدر كل نوع بيأخذ كلام بيت ← STANDARDS بحسب

### Floating - Point Types

Data Type	size	Precision	range
float	4 byte	6 decimal places	IEEE 754
double	8 byte	15 "	ينحدرا كل نوع بيأخذ
long double	10 byte	19 "	كلام بيت ←

### 2's Complement

$$\text{ex } +5 = 0000\ 1010$$

$$1111\ 1010$$

$$+ 0000\ 0001$$

$$(-5) \leftarrow 1111\ 1011$$

#

$$2^{\text{'}} \text{Comp} = 1^{\text{'}} \text{Comp} + 1$$

$$\text{ex } -13 = 1111\ 0011$$

$$0000\ 1100$$

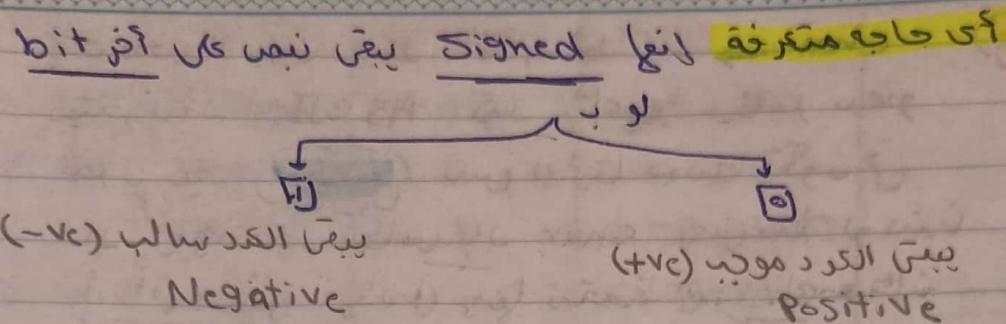
$$\rightarrow 1^{\text{'}} \text{Comp}$$

$$215 + 0000\ 0001 = 216$$

$$215 - 216 = -1$$

$$(413) \leftarrow 0000\ 1101\ 0110$$

$$0.3105 \# 0.5105$$



bool is a data type?

مكتش مترددة ← ANSI C      خارج ←  
اخراج ← اذكر نفسك هنا مستخدمة لازمه تفعل  
 دوك بس ← true or False ←  
 مس بعاف ← 0 or 1 ←

→ #include <stdbool.h>

جزء من الـ STL فيه مس من الـ basic data type

### Quiz

```
int x=5;
float y=2.0;
if(x/y == 2)
    printf("int/float >> int\n");
else if(x/y == 2.5)
    printf("int/float >> float");
```

0x1000	X = 5	4 byte
0x2000	y = 2.0	4 byte

### Memory

### examples

Operation	Result	Operation	Result
5/2	2	2/5	0
5.0/2	2.5	2.0/5	0.4
5/2.0	2.5	2/5.0	0.4
5.0/2.0	2.5	2.0/5.0	0.4

الإجمالي 2.5

## Type Conversion

Float a, b, c;

int s;

$$S = \frac{a * b * c}{100} + 32 / 4 - 3 * 1.1$$

= Float / int int - float

$$= \text{float} + \text{int} - \text{float}$$

$$= \text{float} + \text{float}$$

$$\therefore S = \leftarrow \quad \text{(int)} \rightarrow \text{عمر فزانه (uncertain)}$$

K is an integer value

a is a real value

Arithmetic instructions	Result	Arithmetic instructions	Result
K = 2/9 = 0	0	a = 2/9 = 0	0 + 0.2222
K = 2.0/9 = 0.22	0	a = 2.0/9 = 0.22	0.2222
K = 2/9.0 = 0.22	0	a = 2/9.0 = 0.22	0.2222
K = 2.0/9.0 = 0.22	0	a = 2.0/9.0 = 0.22	0.2222
k = 9/2 = 4	4	a = 9/2 = 4	4.0 → 4.0
k = 9.0/2 = 4.5	4	a = 9.0/2 = 4.5	4.5
K = 9/2.0 = 4.5	4	a = 9/2.0 = 4.5	4.5
K = 9.0/2.0 = 4.5	4	a = 9.0/2.0 = 4.5	4.5

## Type Conversion in C

Explicit

في المقام

بعوله يعمل

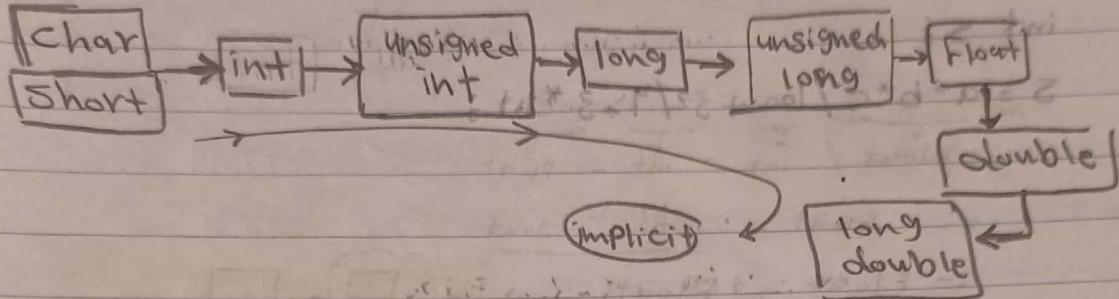
Implicit

automatic

Compiler

implicit

→ "Automatic" ديناميكي من الأعلى للأدنى



## يَعْلَمُ أَنَّهُ الْكَلِمَاتُ

int فالنچر char / int main() {

long int long

~~link~~ float " " int / float : " " "

## Explicit

→ Nipkow Multi automatic Jazzy (Auslieferung)

Compilers يعمل على تحويل الكود البرمجي إلى برمجيات ملائمة لـ CPU.

## تایپ کیسینگ [Type casting]

higher duty type

ex

double x=1.2;

int Sum = (int)X + 1;

→ type casting from double to int - cast op ->

$$\therefore \text{Sum} = 1 + 1 = 2 \quad \#$$

\* هنا نستعرض ما ذكره الله في بدر الكلمة: الكتبة

8 byte (optional) contains memory type limits

• has 4 byte في الـ ٤ بت

ex

$$w + x = ox$$

int x = 0xFF00FF01

$$\text{char } y = (\text{Chair}) \times ;$$

$$y = 0 \times 0 \quad \#$$

# 1000 data وباصر

4 byte → 1 byte  
First byte → 1 byte  
~~#~~ (bias)

**Priority**

1st  $\rightarrow * / \%$  multiplication, division, modulus ↑ أولاً  
 2nd  $\rightarrow + -$  addition, subtraction ↓ ثانياً  
 3rd  $\rightarrow =$  assignment ↓ ثالثاً

$$\begin{aligned} \text{ex: } i &= 2 * 3 / 4 + 4 / 4 + 8 - 2 + 5 / 8 \\ &= 6 / 4 + 4 / 4 + 8 - 2 + 5 / 8 \\ &= 1 / 4 + 1 / 4 + 8 - 2 + 5 / 8 \\ &= 1 + 1 + (8 - 2) + 5 / 8 \\ \therefore i &= 10 \text{ (لذلك } \boxed{8} \text{ )} \neq \# \end{aligned}$$

**printf(), scanf()** → using in C programming only.

مئ هيئه استخدامه في C ؟

فيما يلي دليل على embedded C في microcontroller

Micro screen output يطلع عليه screen

your PC الخاصة

**printf**

string → printf("\_\_\_\_\_");

number → printf("%d", \_\_\_\_\_);

**scanf**

scanf("%d", &\_\_\_\_\_); ← تعيين القيمة ويحفظها في address

**حل مشكلة الـ buffer**

• **fflush(stdout)**

• **fflush(stdin)**

دول بينصفوا الـ buffer عما لو مخزن فيه قيمة او اي

حالة تانية واجهنا عايزين نستخدموه سواه printf() او

scanf() ومن الغالب بترجع بين الـ two lines.

- '\r\n' → Make a new line \n equivalent to '\n' .  
 '\n' → new line بحسب الامر في المطبول بعده  
 '\t' → insert a tab (4 spaces)  
 '\\' → Prints '\"'  
 '\"' → Prints '\""  
 '%d' → prints or scans integer  
 '%c' → prints or scans char single  
 '%X' → an integer value in small hexadecimal Format  
 '%X' → Capital " "  
 '%F' → a real (float) value  
 '%lf' → (double)  
 '%u' → an(unsigned int) value

Memory

note

Char x = -1;

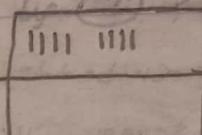
1 → 0000 0001

1111 1110

+

(signed حالة الـ 0)

&amp;x



1 Byte

-1 → 1111 1111

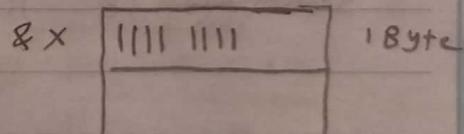
printf("%d", x); → -1

(unsigned حالة الـ 1)

فمن هنا يطبع سخن رف سالب من الـ memory

char x = -1;

printf('x = %u', x);



1 Byte

unsigned char x = -1; ←

ومن هنا يطبع العوارد ما نحن نشوف

الـ (-1) يقاوموا زيد من الـ

وهو ما يطبع خاتما بـ 255

255 ←

Pre Fix	Post Fix
$a = ++x ;$	$a = x++ ;$
هنا هنا زد قيمة ال $x$ <u>أولاً</u> وبعدها نخزن القيمة المنهوبة	هنا هنا نخزن قيمة ال $x$ <u>أولاً</u> وأكملناه زد قيمة ال $x$ بعنصر واحد
<u>ex:</u> ⑨ حال	<u>ex:</u> ⑩ حال
int $x = 1 ;$ $\text{printf}(\%d, ++x) ;$ res = 2 $x = 2$	int $x = 1 ;$ $\text{printf}(\%d, x++) ;$ res = 1 $x = 2$

**Bitwise operators****1) AND (&)**

$$x \& y = 1010$$

$$\begin{array}{r} \& 0101 \\ \hline \text{res} = 0000 \end{array}$$

**2) OR (|)**

$$x | y = 1010$$

$$\begin{array}{r} | 0101 \\ \hline \text{res} = 1111 \end{array}$$

**3) NOT (Complement ~)**

$$N X = \sim (1010)$$

$$= 0101$$

**4) XOR (^)**

$$x ^ y = 1010$$

$$\begin{array}{r} ^ 0101 \\ \hline \text{res} = 1111 \end{array}$$

حال (XOR) لوارد  $=$  digit مختلفين يطلع ١  
اما لوارد متاثرين فهو يطلع ٠

### E) Shift Right ( $>>$ )

$x >> n$

variable to shift      shifting steps

ممكن ان نقوم بـ shift right على digits من العرارة  
ناحية اليمين او بمعنى آخر صوره  $n$  من الأصفار من ناحية  
اليمين

$$(1010) >> 2$$

$$\underline{00} \underline{10} \rightarrow \text{res} = 0010 \#$$

### F) Shift left ( $<<$ )

ممكن ان نقوم بـ shift left على digits من العرارة  
ناحية اليمين او بمعنى آخر صوره  $n$  من الأصفار من  
ناحية اليمين

$$(1010) << 2$$

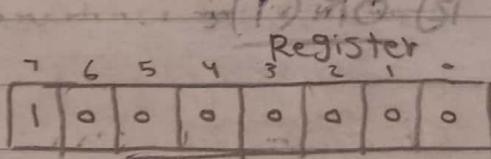
$$\cancel{1} \underline{000} \rightarrow \text{res} = 1000$$

### Set specific bit

يمكن تحطيم bit محددة بـ

$$R \mid = (1 << n)$$

$$R = R \mid (1 << n)$$



يمكن تحطيم bit بـ

$n \rightarrow$  number Bit

$$\text{ex: } R = 1000 \ 0000$$

$$R = (1000 \ 0000) \mid (1 << 1)$$

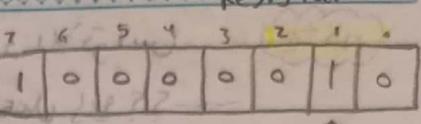
$$R = 1000 \ 0000$$

$$\underline{0000 \ 0010} \text{ or}$$

$$\text{res} = R = 1000 \ 0010 \ #$$

Clear Specific Bit

يعني هنا نخلي bit مكينة بـ (0)



نخلي bit مكينة بـ (0)

$$R \& = n(1 \ll n)$$

$$R = R \& n(1 \ll n)$$

$n \rightarrow$  number bit

ex:  $R = 10000010$

$n = 1$

$$R = (10000010) \& n(1 \ll 1)$$

$$= (10000010) \& N(00000010)$$

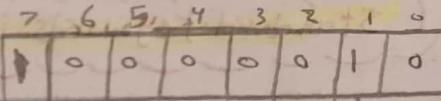
$$= 10000010$$

$$\begin{array}{r} 1111101 \\ \hline & 8 \end{array}$$

$$res = 10000000 \#$$

Toggle Specific Bit

يعني على زين نعكس قيمة bit مخصوص



نعكس دى نظيرها بـ (1)

$$R \wedge = (1 \ll n)$$

$$R = R \wedge (1 \ll n)$$

ex:  $R = 10000010$   $n = 1$

$$R = (10000010) \wedge (1 \ll 1)$$

$$= (10000010) \wedge$$

$$(00000010) \wedge$$

$$res = 10000000 \#$$

**note**

لوجست كايد كود والكود ده المساواة والاتو اس  
فيه مي مطبوب طو فانت عايز تطبب حق عمل بيبي جي

**ctrl + A** then **ctrl + i** → in eclipse only

**Identifiers**

أى حاجة بندر حفها ذى:  
 ↗ Variables      ↗ symbolic constants      ↗ Functions

شروع الايم ← زى شرط اسم المتغيرات

صحيح بكل الحروف سواء بالكليل او بال الكبير

صحيح بكل الأرقام بس ميصح الا يبدأ بـ رقم

غير صحيح بكل العلامات ما عدا الـ ( \_ ) underscore

غير صحيح بالكلمات المحفوظة خـ المـ

**Standard keywords in C are:**

void	char	int	float	double	signed	unsigned	short
long	auto	static	register	const	struct	union	if
else	goto	for	continue	switch	case	break	default
do	while	sizeof	volatile	enum	extern	typedef	return

**Conditions**

الناتج بناء على سبق

↑ ↓  
true      False

**True** → ①

أو ② - أو

أى رقم سواه موجود او سواب

**False** → ② only

(Zero)

## operator

## Meaning

$>$  → greater than

$\geq$  → greater than or equal

$<$  → less than

$\leq$  → less than or equal

$=$  → equal

$\neq$  → Not equal

$!$  → Not       $\rightarrow$  input is true  $\rightarrow$  output is false

"False"  $\rightarrow$  "true"

$\&\&$  → AND  $\rightarrow$  output is true if

all inputs are true

$||$  → OR

$\rightarrow$  output is true if

at least one of the inputs are true

Note:

$x = 7 \quad y = 1$

$(x++) || (++y)$

نهاية x بـ 7 يعني

Compiler يرجع true

حيثما يتحقق كل فلول

ب دون مانع

بعن بعد العملية دي

ال x يتعقب بـ [8]

ال y يـ 10 حـ 10 ، اـ 10 اـ 10

يـ 10 سـ 10 الـ y يـ 10 اـ 10

يـ 10 سـ 10 لـ و سـ 10 اـ 10

اـ 10 كـ 10 بـ true

$x = 0 \quad y = 1$

$(x++) \&\& (++y)$

هـ هنا دـ دـ دـ

الـ x قـ قبل مـ نـ تـ زـ رـ

خـ الـ x بـ 0 يـ 1

فـ الـ النـ اـ نـ اـ نـ

اـ شـ طـ كـ الـ هـ يـ بـ

فـ الـ شـ عـ بـ عـ مـ يـ بـ

فـ دـ هـ دـ

بعـنـ بـ دـ الـ حـ مـ لـ دـ دـ

اـ 0 x يـ 10 بـ ① دـ الـ y بـ ② دـ كـ دـ

حـ 10 اـ 10 اـ 10 جـ 10 سـ 10 دـ

حـ 10 اـ 10 اـ 10 جـ 10 سـ 10 دـ

فـ شـ طـ اـ دـ عـ اـ دـ عـ

**Syntax error**

إذا أردت أن تجعل لها خطأ SYNTAX alors يلي الخطأ ذلك

**Logical error**

لما كتب كود ديناميكي output غير الذي كنت متوقعة

**Compilation error**

أى خطأ يضر بـ compiler يلي يتبع المزيده زي

أنك مسخ فـ variable ما تكون معروفة أو مستخدمة بدون ما تدخل . prototype

**Runtime error****Execution error**

هذا الخطأ المي يحصل والبرنامجه معه زي لو جيت تحمل access على مكان وانت ملاكس (permissions) على error ما يحوله للبرنامجه صحيح وافق .

**if statement**

```
if (* if condition *)
```

{

```
// if body
```

```
{ } // if body
```

```
else if (* else if condition *)
```

{

```
// else if body
```

{

```
else
```

{

```
// else body
```

{

**Note**

لو ارد body المي هيمنت لما يكون الشرط true عباره عن جملة واحدة فنفس العالة دى عادي منحطش { }

**in line Condition**

( condition ) ? if true : if False ; ( X ) ?

ex:  $( 5 > 3 ) ? \text{Anas} : \text{mostafa} ; ( X ) ? \text{Anas} : \text{mostafa}$

**Switch Statement**

switch (\* switch expression \*)

{ caseValue1 : caseBody1 ; caseValue2 : caseBody2 ; ... }

case /\* caseValue \*/ : /\* caseBody \*/ ;

{ /\* caseBody \*/ ; /\* caseBody \*/ ; ... } ;

// case body

break; /\* exits the switch statement \*/

}

default :

{

// default body

break;

}

}

لو نفذنا الشرط هيفي فيه حالتين ويميزيه بعبارى أى شرط فوندر.

من حالة اى IF

IF ('a' || 'A')

switch

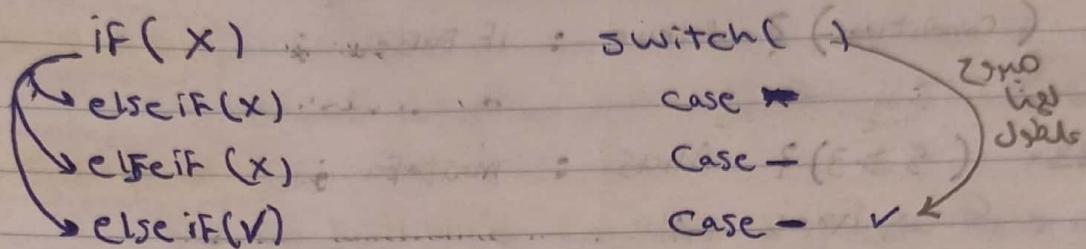
case 'a' :

Case 'A' :

{ //body

}

## difference between **if** and **switch**



في حالة الـ **if** يبدل على المترددة بالترتيب يتبعون الترتيب  
أول يطلع **true** من **case** وبعدهما  
يخرج من البرنامج

**ويمثل **switch** كـ **lookup table****

بمعنى أن **switch** يخرج من الـ **if**

بسن في **switch** ينزل إلى **if** نعم نعمل إلـ **if** **مقدمة**

ما هي كل **switch** لازم يكون **Condition** **حالة** **سلسلة** و **سيط** و **يعني** **يتحقق**

و **switch** **cases** **حالات** **هي** **وكل** **case**  **تكون** **إذا** **الـ** **switch**

**ex → Case y (xx) case y-5 (xx) case y (v) int const**

**int** **const**

**+**

**For Loop**

أول **حالة** الـ **counter** **يبدأ** **من** **قيمة**

ويذكر **أقصى** **تنتهي** **جبي** **خطوة** **دى**

بعد **كـ** **مـ** **وحـ** **متـ** **شرط** **لـ** **وبـ** **عـ**

**فـ** **تـ** **رـ** **جـ** **أـ** **لـ** **وـ** **بـ** **فـ** **يـ** **نـ** **هـ** **الـ**

**For** **body** **يـ** **تـ** **أـ** **عـ** **يـ** **عـ** **يـ** **عـ**

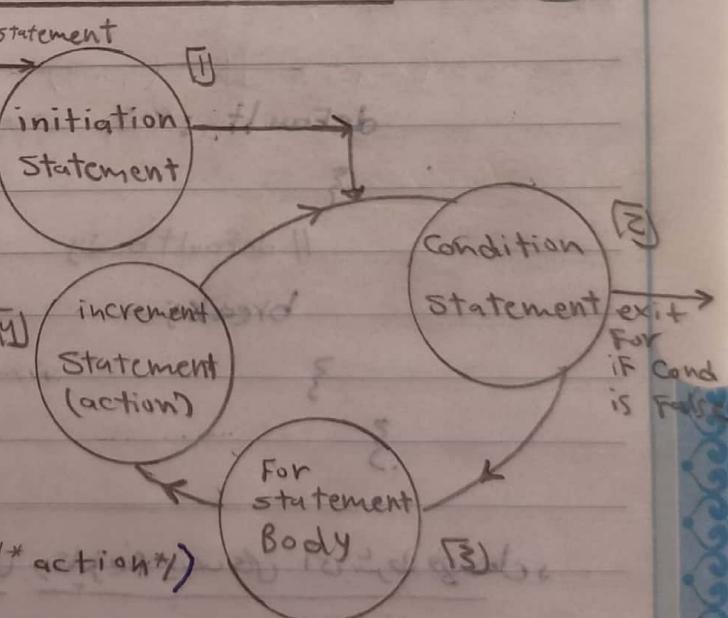
**action** **أـ** **عـ** **يـ** **عـ** **يـ** **عـ**

**For( /\*initiation\*/ ; /\*condition\*/ ; /\*action\*/ )**

**{**

**// For Body**

**}**



**T1 → only one**

**T2, T3, T4 → loop if**

**Condition is true**

**note**`for( ; ; ) ;`**null statement**

~~transliteration~~  
~~commented at top~~  
~~transliteration 3~~  
~~transliteration~~  
~~generalized~~

~~translated to~~  
~~الكلام دة من~~  
~~فيما لا يدور~~  
~~ما نظير ادار~~  
~~ما نظير ادار تبع~~  
~~بساب وجود ادار~~  
~~generalized after~~

**while loop**`while (* condition *)`~~commented at top~~~~Handle the beginning of // while Body~~~~action like increment or~~**note****break** statement is used to exit from  
any loop type**infinite loop**using for loop → `for ( ; ; ) { }`using while loop → `while(1) { }`**do - while loop**do  
  {// do - while body{ while(\* condition \*) ; }**note****continue** statement is used to continue the next  
iteration of loop

ذلك لما ينادي **continue** يندرى الخطوة دى خرا الloop يعني ما شئ نسيت باى  
الكود جوا ال loop فى الخطوة دى ويدرين زمان عاملون ل ريم الloop  
ونكمي باى الloop عارى.

## goto loop

```
// statement
labelname:
    // statement
    // statement
    goto labelname;
    // statement
```

هنا هو هيكل اسفلر عادي  
بس كدول لما يطلبك goto  
فخريج ينفع اسلفرو العادي  
الـ goto الى labelname

```
// statement
```

```
goto labelname;
```

```
// statement
```

```
// statement
```

```
labelname:
```

```
// statement
```

هذا هيكل اسلفرو عادي وبعد ١٥  
يمكنك goto فخريج نازل من  
السلفرو زيادت العدد ١٦  
بعد goto وبعدين يبقى يطلع  
يكملا الكور.

## nested loop

يتبع عبارة عن loop جو في loop في حالة اد

```
while
do - while
for
```

lecture من اكرة اد

lectures labs دل اد المطلوب

assignment دل اد

GitHub رفيقو على دل اد

quiz حل اد ( )

quiz سينتن اد

حل اد quiz اد مرة اخرى