"CS50 course Summary"

Dr; David malan "Harvard university"

CS50 → Computer science → Problem solving

Input → المعالجة → output

المشكلة — كمبيوتر — الحل

التكوين

← ازاى بتتواصل مع الكمبيوتر؟ عن طريق اشارات من (1,0) تسمى "Binary"

→ Byte — English = Byte ⟷ BIT = (0,1)

1111 1111 — a = 01100001 ⟷ Byte = 8 BIT

← كل حرف فى لغة الانجليزية والصور والفيديوهات كله تتكون من (0,1)

→ Decimal system → Binary system ببساطة

النظام العشرى — النظام الثنائى يتمثل (1,0)

10^2	10^1	10^0
100	10	1
	1	
	2	
	3	

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
						1	1

100 = 128 64 32 16 8 4 2 1 ∴

100 → 0 1 1 0 0 1 0 0

حلو ازاى بقا!! الكمبيوتر بيفهم الاصوات . الفيديوهات . الصور للغات .

ASCII code → (0,1)

American Standard Information Technology.

هنا البيانات تمثل من bit (8:7) وكانه غير كافى حرف 127 فقط

غير كافى 255 ← 8 BIT 127 ← 7 BIT

unicode (32 bit, 16 bit) من هنا ممكن امثل اى لغة

R G B — اقل درجة لونه 0 — Byte — الالوان

10 01 11 → 3 Byte — 1111 1111 — 255 اكبر درجة لونه

← الصور (Images)

Photo → Pixel → Color → Byte

← الفيديوهات

اى فيديو فى الدنيا عبارة عن مجموعة من الصور المتتالية

Video → Photo → Pixel → Color → Byte

⇐ الموسيقى و الاصوات :

الموسيقى → نغمات → ترددات → Byte

يعنى فى نهاية كل حاجة بترجع إلى (0,1) علشان كمبيوتر يفهمها

← يعنى ايه الخوارزميات ؟

هو الوصول إلى حل مشكلة بى أبسط وأسرع طريقة

أهم مثال هنا سجل الهواتف

2+2+2 = 6

6+2 = 8 → دى الأفضل

2+2+2+2 = 8 → أبطأ

← (Pseudo code) هى الخطوات البرمجية اللى هنكتبه داخل تطبيق

مهم جداً جداً

Example :-

Pick up Phone Book
open To middle of Phone Book
look at Page
If Person is on Page
call Person
Else if Person is earlier in book
open To middle of left half of Book
Go back To line 3
Else if ....

→ function أوامر

الجمل الشرطية

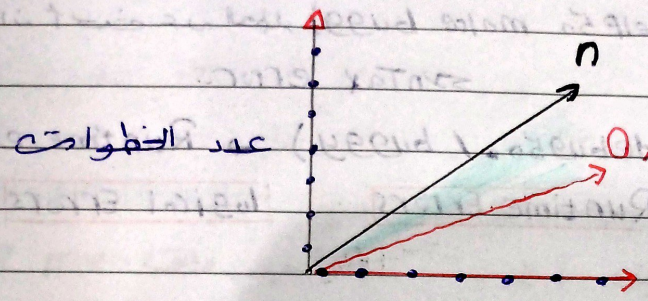
→ if , Else if , Else

→ loop التكرار

→ Conditionals

→ Boolean expression

تعالى نطبق عملى ...



أسوء سيناريو big O ← O

عدد العناصر (n)

أفضل سيناريو big Ω

Binary search

1	2	3	4	5	6	7	8	9	10

تعالى بقا.. نأخذ حاجة جديدة:

الخوارزمية بتمسك اول عنصر وتقارنه بكل عنصر بعد كدا في حاله (Array) غير مرتبة.

n	4	3	8	7	2	1

1	2	3	4	7	8

$n + n-1 + n-2 + n-3 + \dots$

القانون العام للحساب عدد الخطوات $O(n^2)$ تقارنه - تمسك

$\Omega(n^2)$

```
For i from 0 to n-1
    Find smallest number between number[i] and
    number[n-1]
    Swap smallest number with numbers[i]
```

← تعالى يا هندسة "bubble sort"

هي بتاخذ اول عنصرين وتقارنه ما بينهم العنصر الأصغر بتأخذه وتحطه في الأول والعنصر التاني بعدها وبعد كدا تأخذ الاتنين اللي بعدهم.

→ language C :

EXPLORE

hello

hello.c

TERMINAL

make hello → يشغل الملف

clear → (Termnal) بينظف

./hello → بيشغل ويطبع

CS50.h (معلومة) تحتوي على

get_string() →

get_char()

get_double()

get_float()

get_int()

get_long()

```
# include <stdio.h>
# include <CS50.h>
int main(void) {
    string answer = get_string("what's your name?");
    Printf("hello%s\n" answer);
```

"Basic"

Code

```
# include <stdio.h>
int main(void) {
    Printf("Hello, world\n");
    Printf("Hello, world\n");
}
```

→ DaTa TYPES البيانات

string name = "Mahmoud" نصية

int number = 10 رقمية

Float number = 1,5 كسر

double number = 0,65432 رقم عشري

long number = 111349567

bool True/False

Char 'a' حرف

comment in C language → //

← عاوز أعمل برنامج يكتب بي أسمي؟

* "Define logic gates" هى element اساسى فى أى بوابة و هى تتكون من مجموعة من "Transaster" و يكون لها "InPuT" متعدد و outPut واحد.

"ImPortant information"

InPuT	مدخل		
outPuT	مخرج	locked key	مفتاح مغلق
0	low	luminous lamP	مصباح مضيء
1	High	lamP off	مصباح مطفئ
oPen key	مفتاح مفتوح		

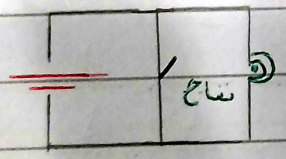
* "inverter gate" بوابة العاكس (NoT) $\neg A$ $\bar{A}$ $!A$ $>A$

"Draw"

تسمى بعاكس

InPuT = outPuT

VerificaTion Table

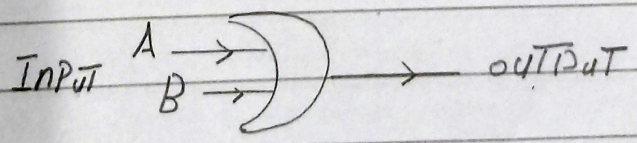


InPuT	outPuT
T	F
F	T

جميع البوابات المنطقية معتمد على الجبر الثنائى "Binary algebra"

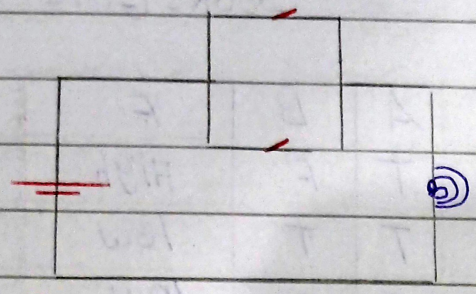
* "gates of choice" بوابة الأختيار وتسمى (OR) الجمع

"Draw"



الفرسان

جدول التحقق "Verification Table"



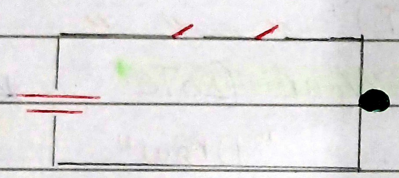
A	B	OUTPUT
T	T	T
T	F	T
F	T	T
F	F	F

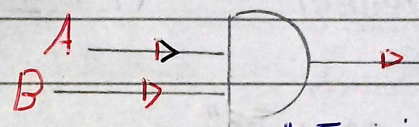
المعادلة المنطقية

$(A+B=F)$

$(A \vee B = F)$

* "Compatibility Gates" بوابة التوافق وهى "AND" الضرب





"Verification Table"

ملحوظة هامة

A	B	OUT
T	T	T
T	F	F
F	T	F
F	F	F

لمعرفة عدد احتمالات الجدول هو يجب أن يأتى 2^n اى انه (n) عدد المدخلات

مثال لو عدد المدخلات 3 ∴ $2^3=8$

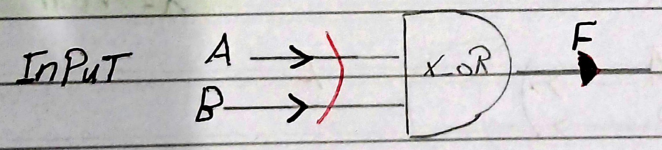
المعادلة المنطقية

$F = AB$

$A \wedge B, A * B, A.B, AB, A \& B$

* "X-OR Gate" بوابة X-OR

"Draw"



"logical equation"

$F = A \oplus B$

$F = \bar{A}B + A\bar{B}$

تسمى هذة البوابة "Asymmetry" عدم التكافؤ

الفرسان

SUBJECT:

DATE: / /

Verification Table — X-OR Gate *

$n = 2^2 = 4$ احتمالات

A	B	F
T	F	High
T	T	low
F	F	low
F	T	High

• عند التحويل للنظام العشري "Decimal" الارقام الناتجة تعني الترتيب العشري للارقام

القاعدة (rule) المداخل المتشابهة low

المداخل المختلفة "High"

$F = A\bar{B} + \bar{A}B$

"Save of Product"

لو عدد المداخل (T) زوجي 0 (F)

" " " (T) فردي 1 (T)

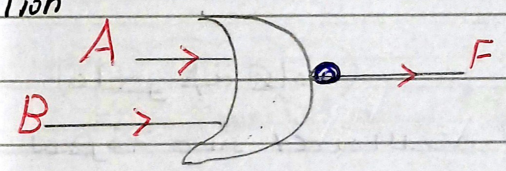
"Nour Gate" *

$(A \downarrow B)$

"logical equaction"

$F = \overline{A+B}$

"Draw"

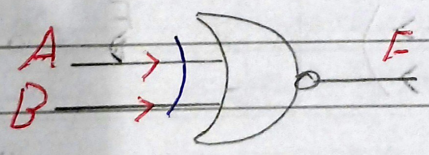


عكس (OR)

A	B	F
0	0	1
0	1	0
1	0	0
1	1	0

إذن بوابة Nour تعطي واحد عندما يكون كل الدخل بإشارة low

"X-Nour Gate" *



"logical equation"

المعادلة المنطقية

$F = A \odot B$

$F = AB + \bar{A}\bar{B}$

بوابة التكافؤ

عدد الاحتمالات $N = 2^2 = 4$

الفرسان

SUBJECT:

DATE: / /

* المتشابة يخرج "High"
* المختلف يخرج "low"
* عدد فردى "low"
* زوجى "High"

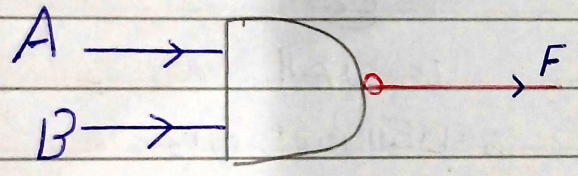
A	B	F
0	0	High
0	1	low
1	0	low
1	1	High

الاعداد الفردية 1, 3, 5, 7, 9, 11

الاعداد الزوجية 2, 4, 8, 6, 10, 12, 14

Draw الرسم

NAND Gate (A/B)



$\therefore F = \overline{A \cdot B}$

هى هى بوابة (AND) لكن الاختلاف إلا احنا بنعكس الخرج إلا الخارج بالا هو (الخرج OUTPUT)

لو بوابة (AND) ذات ثالثة مداخل نستنتج أن * إذا كانت كل مداخل T هخارج (F) والباقى كلها (Ture)

A	B	$\overline{A \cdot B}$
T	T	F
T	F	T
F	T	T
F	F	T

الفرسان

أوامر TERMINAL ← linux

$ code mario.c → بتقوم بأنشاء ملف

$ rm mario.c → بتزيل الملف تم أنشاءه (remove)

$ ls → بيقولك عندك كام ملف

$ mv ملف, الجديد → أعادة تسمية ملف معين

$ mkdir الفولدر → أنشاء فولدر جديد

$ rmdir الفولدر → حذف فولدر

$ cd الفولدر → لو عاوز تدخش جو فولدر

لو طلبت ارجع خطوة إلى الخلف / ..

← ازاى بقا نعرف functions في language C

```
# include <stdio.h>
void meow (int n);        الفيكشن
int main (void) {
.... meow(10);
}
void meow (int n)         الفيكشن
{   for (int i=0; i<n; i++)    التكرار
{   printf ("meow\n");        جملة الطباعة
}   }
```

هناك امر يسمى clang

ازاى بقا نستخدم clang -o hello hello.c

الاداة — machine code — Source code

علشان نحول ملف هذه لغة C إلى ملف معمول بلغة machine code

STEPS: Preprocessing → compiling → assembling → linking

→ bugg خطء
→ debugg علاج الخطاء

(help50 make bugg) علشان أبحث عن الخطاء
syntax errors

(debug50 ./ buggy) → Runtime errors

Syntax Errors | Runtime Errors | logical Errors

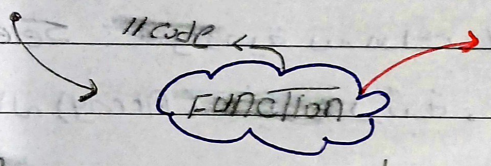
← (Array) هو مكان أقدر أخزن فيها الداتا بشكل متتالي []

Example: --> int main (void) { int scores [3] = {72, 3, 33};

← \0 ← null ← اول ما الكمبيوتر ما يشوفها بيوقف ولا عاجة بعدها مش هتنطبع

"HI! \0 ahmed" ; → output HI!

```
int main(void) {
    //code
}
```




```
int fun(void) {
    //Code
}
```

./file عن طريق

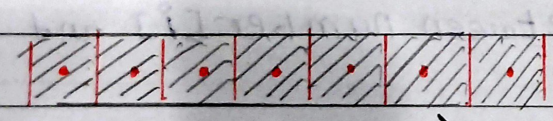
func (123 ...) ;

Array هو بيعد عدد المدخلات اللي داخلك وهو "Argument".

من نوع "String".

← الخوارزميات : هي أبسط طريقة وأسهل اللي عنده خلاله بنوصل إلى حل مشكلة معينة.

Input → Algorithms → Output



→ linear search

array [1000]

```
For i from 0 to n-1
    If number behind doors [i]
        Return true
Return false
```

IF numbers [i] and numbers [i+1] out of order swap Them

← 4	2	6	3	7	→
← 2	4	3	6	7	→

$O(n^2)$

$\Omega(n)$ ... أكثر ذكاء

for i from 0 to n-2

تعالى نفهم حاجة جديدة (recursion)

```
void fun (int n)
{
    // code
    fun();  (n-1)
}
```

```
int main (void)
{
    fun (3);
}
```

• هو يعتمد على مبدأ الترتيب على مبدأ التقسيم و التجميع : "Merge sort" ←

بباسطة احنا بنفكك عناصر (array) و

5	2	7	4	3	1

ترتيبة بعد كدا عن طريق التقسيم و التجميع

هبدأ في الأخر هدمجهم لكن

بشكل مرتب.

5 2 7 ↓ 5 2 7 ↓ 5 2 7

4 3 1 ↑ 4 3 1 ↑ 4 3 1

$O(n \log n)$ n.n

هي اسرع بكتير لكن أكيد لية ميزة ولية عيبه لكن دا في أفضل حالة

selection sort ← Buble sort ← Merge set