

Lecture 1

Computer:- A device that takes data as input
Process it and produces information as output

Difference between DATA and INFORMATION:-

Data: Facts : البيانات التي تدخل وهي ليست مفيدة
انها لا يمكن اتخاذ قرار

Information:- يمكن اتخاذ قرار عليها



Hardware: Physical parts

Software:

logical parts
or set of instructions
that tell hardware
what to do

As a summary : A hardware consists of :-

- 1) CPU : Central Processing unit (Brain of computer)
 - ↳ ALU: Arithmetic / logic units
 - ↳ CU: Control Unit
 - ↳ Register :- CPU الذاكرة
- 2) Storage
 < Primary
 secondary
- 3) Input / output

To explain it more :- ① CPU

a) ALU

Arithmetic :

حساب يستطيع القيام بهيات جمع فقس
 جمع = ضرب = قسمة = طرح

Logic :- > or < or = : Basic logic

b) CU : controls everything

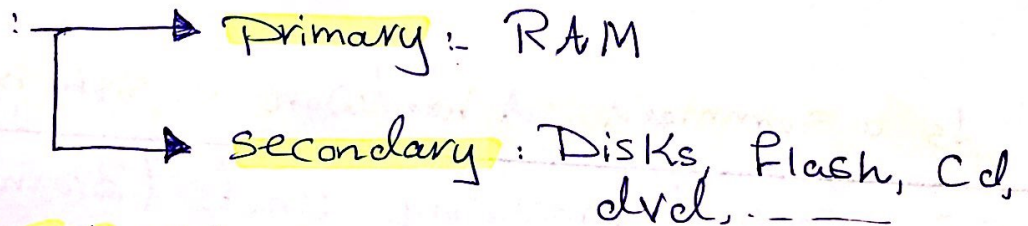
c) Register :- you put things that you want to reach it fast But it's not that big (it fits important things)

CPU = CU + ALU

The Machine cycle :- (4 processes)

- 1- Fetch (get) next instruction
- 2- decode : ترجمه : find what instruction is (changes instruction to commands)
- 3- Execute : run instructions
- 4- store in RAM

② Storage



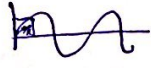
③ Input & output

- Input:
- Keyboard
 - Scanner
 - Microphone
- Output:
- Printer
 - Speakers
 - Screen
 - Plotter
- ← اذونات (طابعه كبرية)

Difference Between RAM & ROM:-	
RAM	ROM
① Read & write	Read only
② Volatile متغير = Temporary	Not-volatile غير متغير

Lecture 2

Data Representation, Data can be:-

- numbers $\begin{cases} \rightarrow \text{integers} \\ \rightarrow \text{floats} \end{cases}$
- Characters \rightarrow ASCII $A \rightarrow 65$ $a \rightarrow 97$
- Pictures \square
- videos
- sound \rightarrow 
- strings "ali"

Bin. To represent data, we change it to numbers $\left\{ \begin{array}{l} \text{in the Decimal system} \\ \text{and} \\ \text{in the Binary system} \end{array} \right.$

Those numbers we represent them as in the Binary System

1: on 0: off

Numbering systems

- binary :- (البنائي) 2 (0-1)
- Decimal :- (العشري) 10 (0-9)
- octal :- (الثماني) 8 (0-7)
- Hexadecimal (الست عشري) 16 (0-15)
(0, -9, A, B, C, D, E, F)

How to change from decimal to binary:-

Exp:- $(23)_{10} \rightarrow (10111)_2$

①

$$\begin{array}{r} 11 \\ 2 \overline{) 23} \\ \underline{22} \\ 1 \end{array}$$

③ $\begin{array}{r} 2 \\ 2 \overline{) 5} \\ \underline{4} \\ 1 \end{array}$

⑤ $\begin{array}{r} 25 \\ 2 \overline{) 1} \\ \underline{0} \\ 1 \end{array}$

②

$$\begin{array}{r} 5 \\ 2 \overline{) 11} \\ \underline{10} \\ 1 \end{array}$$

④ $\begin{array}{r} 25 \\ 2 \overline{) 10} \\ \underline{10} \\ 0 \end{array}$

How to change from binary to decimal

$$(10111)_2 \rightarrow (23)_{10}$$

$$\begin{array}{r} 2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0 \\ 16 + 0 + 4 + 2 + 1 \\ = 23 \end{array}$$

How to change from decimal to octal

$$(23)_{10} \rightarrow (27)_8$$

$$\textcircled{1} \quad \begin{array}{r} 8 \overline{) 23} \\ \underline{16} \\ 7 \end{array}$$

$$\textcircled{2} \quad \begin{array}{r} 8 \overline{) 27} \\ \underline{24} \\ 3 \end{array}$$

How to change from octal to decimal

$$(27)_8 \rightarrow (23)_{10}$$

$$\begin{array}{r} 8^1 \quad 8^0 \\ 16 + 7 \\ = 23 \end{array}$$

We change between any systems in the same way

As an example:- $(23)_{10} \rightarrow (113)_4$

$$\begin{array}{r} 4 \overline{) 23} \\ \underline{20} \\ 3 \end{array} \rightarrow \begin{array}{r} 4 \overline{) 23} \\ \underline{20} \\ 3 \end{array} \rightarrow \begin{array}{r} 4 \overline{) 23} \\ \underline{20} \\ 3 \end{array}$$

$$(113)_4 \rightarrow (23)_{10}$$

$$\begin{array}{r} 4^2 \quad 4^1 \quad 4^0 \\ 16 + 4 + 3 \\ = 23 \end{array}$$

How to change from Hexadecimal to decimal

$$(29)_{10} \rightarrow (1D)_{16}$$

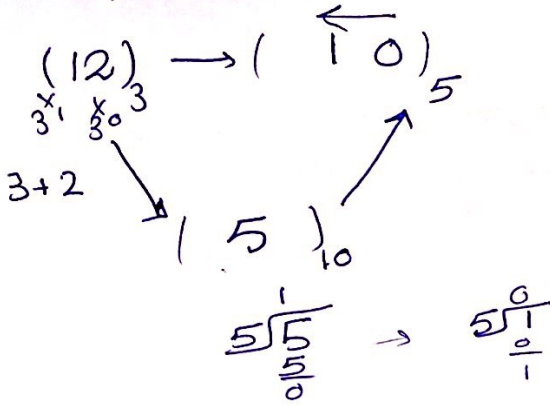
$$\begin{array}{r} 16 \overline{) 29} \\ \underline{16} \\ 13 \end{array}$$

Note:-
 10 → A
 11 → B
 12 → C
 13 → D
 14 → E
 15 → F

$$\begin{matrix} (1D)_{16} & \rightarrow & (29)_{10} \\ \begin{matrix} 16^1 & 16^0 \end{matrix} \end{matrix}$$

$$16 + 13 \rightarrow 29$$

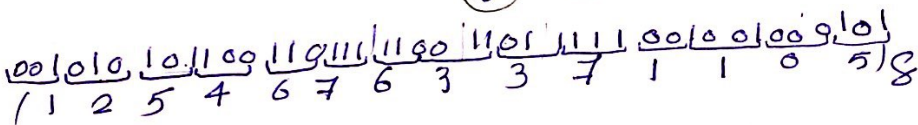
Examples: Between 2 systems that we don't know



Examples

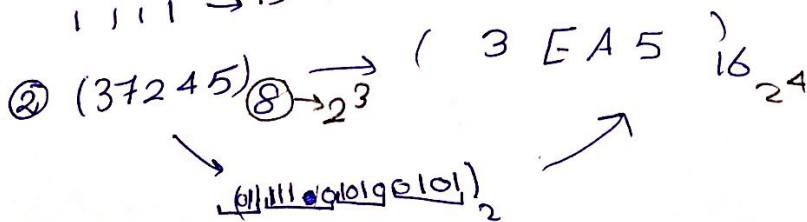
① (A B 3 7 C D F 2 4 5) \rightarrow ()
 (16) $\rightarrow 2^4$

(8) $\rightarrow 2^3$



\rightarrow This works only when the system can be divided on 2^2 like 8, 16, 32, ...

- Note:
- 0000 $\rightarrow 0$
 - 0001 $\rightarrow 1$
 - 0010 $\rightarrow 2$
 - 0011 $\rightarrow 3$
 - 0100 $\rightarrow 4$
 - 0101 $\rightarrow 5$
 - 0110 $\rightarrow 6$
 - 0111 $\rightarrow 7$
 - 1000 $\rightarrow 8$
 - 1001 $\rightarrow 9$
 - 1010 $\rightarrow 10$
 - 1011
 - 1100
 - 1101
 - 1110
 - 1111 $\rightarrow 15$



$$(2.5)_{10} \rightarrow (10.1)_2$$

$$0.5 \times 2 = 1.0$$

$$(2.25)_{10} \rightarrow (10.01)_2$$

$$0.25 \times 2 = 0.5$$

$$0.5 \times 2 = 1.0$$

وكان الجواب
مكرر 1.0
لأخترنا 0.01

lecture 3

$(52)_8 \rightarrow (222)_4 \rightarrow (46)_9$
 $(52)_8 \rightarrow (1010101)_2 \rightarrow (42)_{10} \rightarrow (46)_9$
 $(42)_{10} \rightarrow (46)_9$

To make sure that the answer is right we change the octal to decimal, then we change the decimal to decimal & if the answer is 42 for all then my answer is right!

Examples:-

① $(13.125)_{10} \rightarrow (11101.001)_2 \rightarrow (13.125)_{10}$
 (مخطط الأعداد)
 $0.125 \times 2 = 0.25$
 $0.25 \times 2 = 0.5$
 $0.5 \times 2 = 1.0$
 $8+4+0+1 = 13$ $(0+0+\frac{1}{8}) = 0.125$

To change from 2 to 8:-

$(001101.1001)_2 \rightarrow (15.1)_8$

from 2 to 16
 $(1101.0010)_2 \rightarrow (D.2)_{16}$
 $(16^1 \ 16^0)$
 $= (13.125)_{10}$

Binary operations

$$\left(\begin{array}{r} 101 \\ 10 \\ \hline 111 \end{array} \right)_2 \left(\begin{array}{r} 5 \\ 2 \\ \hline 7 \end{array} \right)_{10}$$

• Rules:- جمع رقمين فقط

$$\begin{array}{r} 0 \\ 0 \\ \hline 0 \end{array}^+, \begin{array}{r} 0 \\ 1 \\ \hline 1 \end{array}^+, \begin{array}{r} 1 \\ 0 \\ \hline 1 \end{array}^+, \begin{array}{r} 1 \\ 1 \\ \hline 10 \end{array}^+, \begin{array}{r} 1 \\ 1 \\ \hline 11 \end{array}^+$$

Examples:-

$$\textcircled{1} \begin{array}{r} 1011 \\ 10000 \\ \hline 11011 \end{array}^+$$

$$\textcircled{2} \left(\begin{array}{r} 11011 \\ 11100 \\ \hline 100111 \end{array} \right)_2 \left(\begin{array}{r} 11 \\ 28 \\ \hline 39 \end{array} \right)_{10}$$

$$\textcircled{3} \begin{array}{r} 111 \\ 111 \\ \hline 1110 \end{array}^+$$

$$\textcircled{4} \left(\begin{array}{r} 11.01 \\ 101.10 \\ \hline 1000.11 \end{array} \right)_2 \left(\begin{array}{r} 3.25 \\ 5.5 \\ \hline 8.75 \end{array} \right)_{10}$$

$$\textcircled{5} \left(\begin{array}{r} 11.11 \\ 10.10 \\ \hline 110.01 \\ \hline 6.25 \end{array} \right)_2 \left(\begin{array}{r} 3.75 \\ 2.5 \\ \hline 6.25 \end{array} \right)_{10}$$

Note:-

إذا كان الرقم يبدأ بـ 0 فهو موجب: 0101
 إذا كان الرقم يبدأ بـ 1 فهو سالب: 1011

Two's Complement

negative $\leftarrow 101 \rightarrow$ ✗ It's -3
 positive $\leftarrow 0101 \rightarrow$ +5 ✓

• 1's complement

101
 010

• we put 0 instead of 1 and we put 1 instead of 0

• 2's complement

010
 1+
 011
 +3

• we add 1 to the number

↳ so 101 is -3

Ex: 0110 \rightarrow +6

1001
 1+
 1010 \rightarrow -6

0101
 1+
 0110 \rightarrow +6

5 - 2
 = 5 + (-2)
 0101 1110
 ↓
 0101
 1110+
 ✗0011

• 2 is (0010) \rightarrow +2

to get -2 we use 2's complement :-

0010
 1+
 1101
 1110

• لا تأخذوا على المسألة البسار لأنه القليلة
 كانت بين رعيته أكبرها تنبؤه من أربع خانات
 وبالتالي الجواب يكون من أربع خانات
 ويستثنى الخانات من اليسار

Ex: $(-7) \times 1$
 $\downarrow \quad \downarrow$
 $(1001) (0001)$

$+7$ من 0100 ↓
 $\begin{array}{r} 0100 \\ 1000 \\ \hline 1000 \end{array}$

$\cdot -7 +$
 $\frac{1}{-6}$

7 من

$\begin{array}{r} 1001 \\ 0001 \\ \hline 1010 \end{array}$ → -6 ✓

Ex: $\frac{6}{-5-1}$
 $\leftarrow \quad \rightarrow$
 $1011 \quad 1111$

$\cdot +5$ من 0101
 $\begin{array}{r} 0101 \\ 1010 \\ \hline 1 \end{array}$
 $-5 \leftarrow 1011$

$\begin{array}{r} 1111 \\ 1011 \\ \hline 1111 \end{array} +$
 $\frac{1111}{11010}$ → -6

$\cdot +1$ من 0001
 $\begin{array}{r} 0001 \\ 1110 \\ \hline 1 \end{array} +$
 $-1 \leftarrow 1111$

Ex: -5×4
 $\swarrow \quad \searrow$
 $1011 \quad 1100$

$\cdot +4$ من 0100
 $\begin{array}{r} 0100 \\ 1011 \\ \hline 1011 \\ \hline 1 \end{array}$
 $\frac{1}{1100}$ → -4

$-5 \neq$
 -4
 $\ominus 9$

$\begin{array}{r} 1011 \\ 1100 \\ \hline 10111 \end{array}$ → $\oplus 7$

• الجواب خاطئ بسبب الأرقام التي دخلت في العملية

تحتاج إلى أربع خانة لتمثيلها

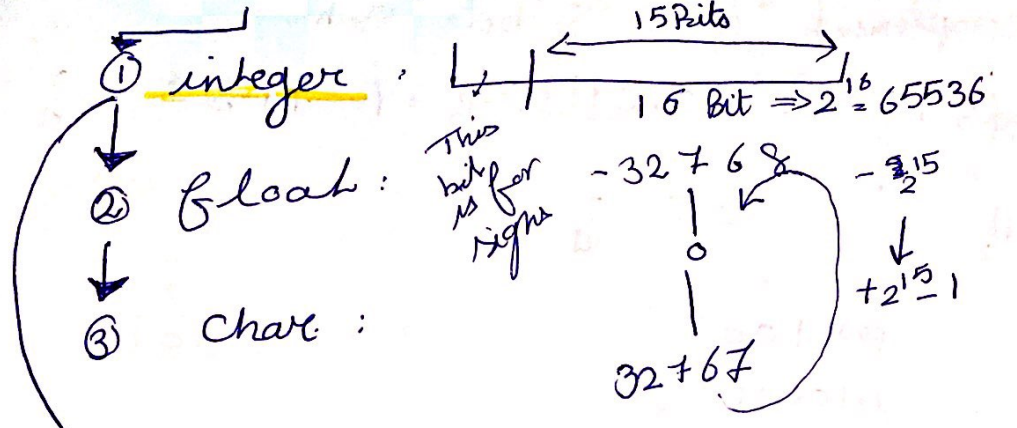
ولكن الجواب (-4) وهو ليس صحيحاً يحتاج إلى خمسة خانة لتمثيله

• لذلك الجواب ليس صحيحاً كغيره لكي يتأكد من الجواب فيكون عدد

الخانات ~~6 أو 7~~ أو 128 أو عدد أكبر من الخانات

• عندما يكون الجواب غريباً أو خاطئاً نزيد عدد الخانات للتأكد

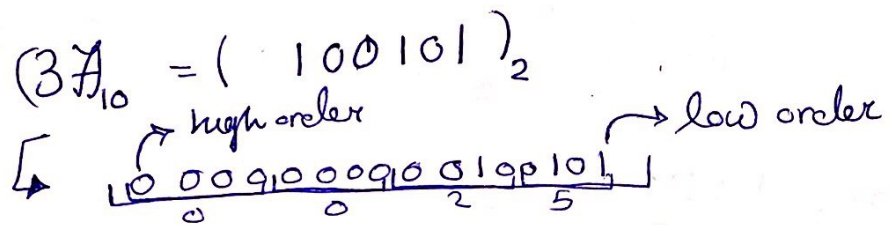
Data Presentation:



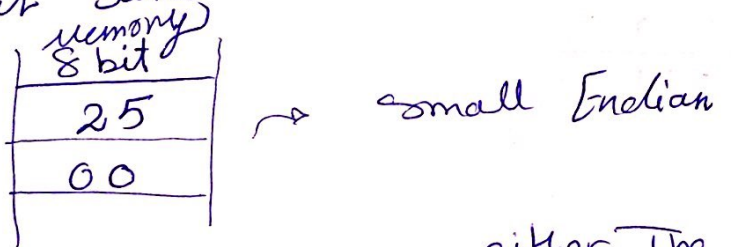
• 2's complement is used with integers

if I had one bit I can put either 1 or zero so $2^1 = 2$

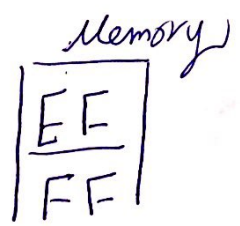
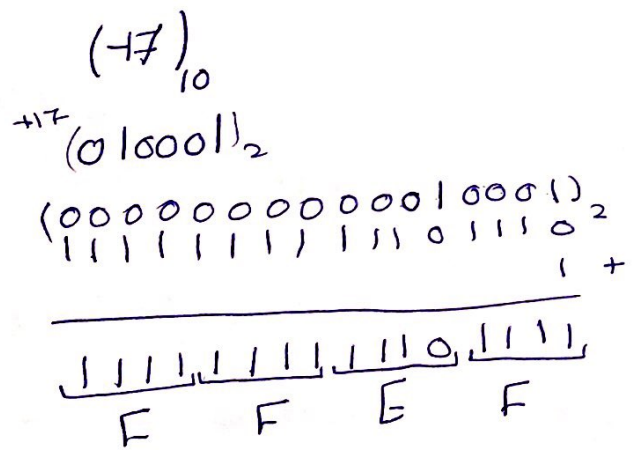
if I had 2 bits I can put 4 diff things $2^2 = 4$



• How is it saved in the memory?



• There is 2 ways: either the Big Endian or the small Endian



3) Char.

ASCII → Extended ASCII → Unicode
ASCII → Univ. code

a-z, A-Z, 0-9, ?, #, ---

• ASCII (128)

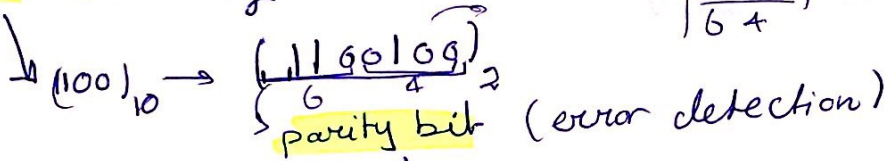
→ American standard code for information interchange

A - 65

d - 100

a - 97

d → memory → using even or odd parity



if using odd: 1100100
 There is 3 ones (it's an odd no)
 so we put 0 in the parity bit

if using even: 1100100
 There is 3 ones (it's an odd no)
 so we put 1 and it becomes 4 which is a parity bit

As an example: a h m a c l :- a → h → m → a → c → l →

- a - 97
- b - 98
- c - 99
- d - 100
- e - 101
- f - 102
- g - 103
- h - 104
- i - 105

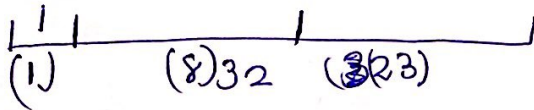
② float :-

$$(-4.25)_{10}$$

$$\left((100.01)_{2} \right)$$

• Scientific notation

$$= 1.0001 \times 2^2$$



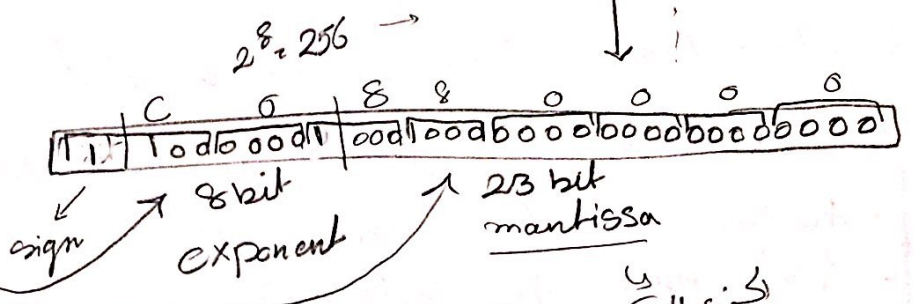
5247.32
The Scientific Notation
is 5.24732×10^3

Quiz 5

Floating Point representation

$(-4.25)_{10}$

- ① 100.01
- ② ①0001 x 2²



كبره الكسريه صالقم

$\Rightarrow 127 + 2 = 129$

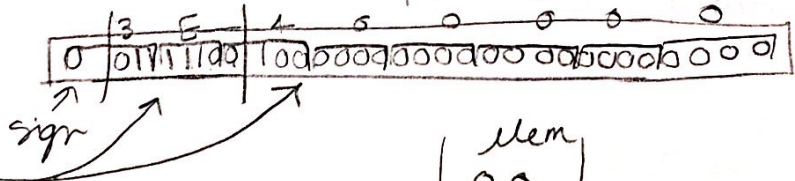
Memory	
00	
00	
88	
C0	

- $(0.2) \times 2 = 0.4$
- $(0.4) \times 2 = 0.8$
- $(0.8) \times 2 = 1.6$
- $(0.6) \times 2 = 1.2$
- $0.2 \times 2 = 0.4$

$127 + (-3) = 124$

2+2⁰=3 افه نسف 127 قبل

- ① 0.00110₋₃
- ② 1.10 x 2⁻³
- = 1.① x 2^③



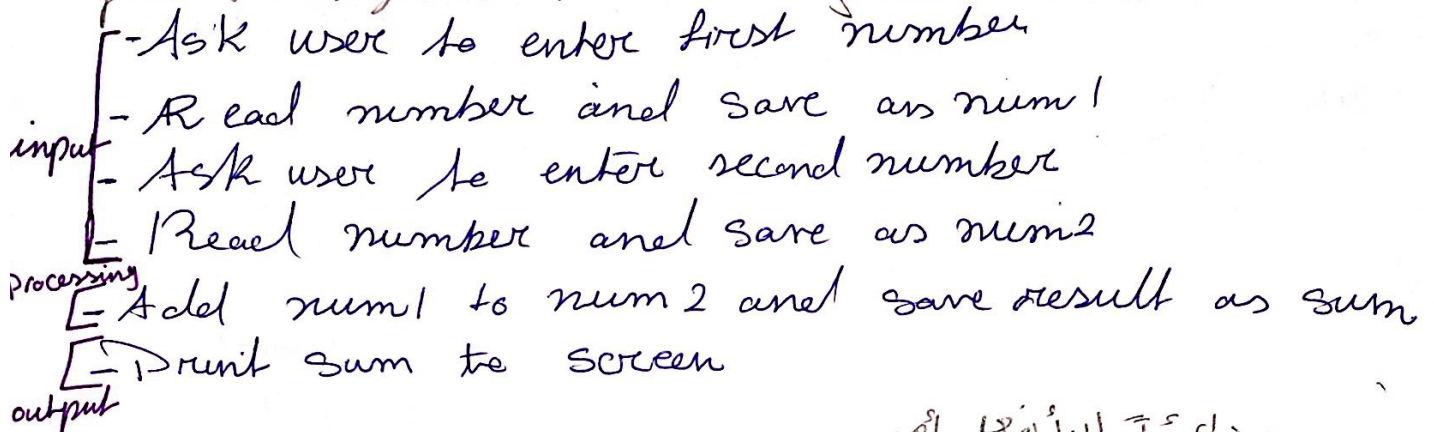
Mem	
00	
00	
40	
3E	

Algorithms

طرق الحل

- Sequence
- Conditional
- Repetition

• In Algorithms there is an input, an output & process
find the algorithm to sum any two numbers



- ذلك يبدأ بفعل أمر
- المحلة يجب ان تكون شرطاً (او اكثر) كذا اعلى
- عكس الاعمال التي نوع من الكيفيات بشرط ان يكون لغرض كامل ووسع
- الترتيب مهم أحياناً

In the Top-down design :-

- 1- get number → 1.1 get first number → 1.1.1 Ask user
→ 1.2- get second number → 1.1.2 Ask Read --
- 2- first sum
- 3- Print sum

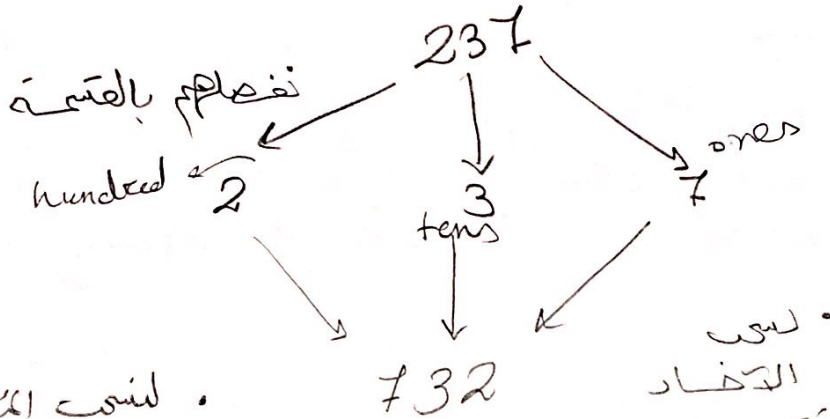
Lecture 6

Write an algorithm (Pseudo-code) :-

- to reverse any given three digit number

$$237 \rightarrow 732$$

$$102 \rightarrow 201$$



$$5 \% 2 = 1$$

ما هو الباقي

$$2 \% 8 = 2$$

But

$$5 / 2 = 2$$

لنسب المئات

$$\begin{array}{r} 100 \overline{) 237} \\ \underline{200} \\ 37 \end{array}$$

لنسب
الآحاد
نقسم على 10

$$\begin{array}{r} 23 \overline{) 237} \\ \underline{230} \\ 7 \end{array}$$

لنسب العشرات هناك أكثر من طريقة :-

أخذ آخر خانتي 37 ونقسم على 10 ونأخذ الباقي

$$\begin{array}{r} 3 \overline{) 37} \\ \underline{30} \\ 7 \end{array}$$

أو نأخذ أول خانتي 23 ونقسم على 10 ونأخذ الباقي

The Algorithm:-

- 1- Ask user to Enter any three digit number
- 2- Read number and save as num
- 3- Divide num by a hundred and save result as hundreds
- 4- Divide num by ten and save remainder as ones
- 5- Divide num by a hundred and save remainder as temp
- 6- Divide temp by ten and save result as tens

- 7- Multiply ones by a hundred and save result to rev
- 8- Multiply tens by ten and add result to rev
- 9- Add hundreds to rev
- 10- Print rev to screen

$$\text{rev} + = \text{ten} * 10$$

$$\text{rev} = \text{rev} + (\text{tens} * 10)$$

We Try :- 352

$$\text{num} = 352$$

$$\text{hundreds} = 352 / 100 = 3$$

$$\text{ones} = 352 \% 10 = 2$$

$$\text{temp} = 352 \% 100 = 52$$

$$\text{tens} = 52 \% 10 = 5$$

$$\text{rev} = 2 * 100 = 200$$

$$\text{rev} = 200 + (5 * 10)$$

$$= 250$$

$$\text{rev} = 250 + 3$$

$$= 253$$

253

Example :- In-out

Ex:

$$\begin{cases} 5123 \% 10 = 3 \\ 5123 / 10 = 512 \\ 512 \% 10 = 2 \\ 512 / 10 = 51 \\ 51 \% 10 = 1 \\ 51 / 10 = 5 \\ 5 \% 10 = 5 \\ 5 / 10 = 0 \end{cases}$$

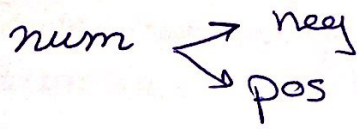
```
x = num % 10 ;
num = num / 10 ;

x = num % 10 ;
num = num / 10 ;

x = num % 10 ;
num = num / 10 ;
```

We use the loop

Conditional (selection)



>
<
==

X=5 put 5 in X
X=25 does 5 equal X.

• If statements

if num is less than zero

Print "num is negative" to screen

Else

Print "num is positive" to screen

END IF

Conditional (Selection)

Lecture 7

- Write an algorithm to decide whether a given number is odd or even

Ask user to enter any number

Read number and save as num

Divide num by two and save remainder as rem
If rem equals zero

Print "num is even" to screen

Else

Print "num is odd" to screen

End IF

- Write an algorithm to change marks to letter grades such that (A=90-100, B=80-90, C=70-79, D=60-69, F=0-59)

Ask user to enter mark

Read mark and save as mk

IF mk is greater than or equal to ninety

Print "grade is A" to screen

Print "good job" to screen

Else IF mk is greater than or equal to eighty

Print "grade is B" to screen

Else IF mk is greater than or equal to sixty nine

Print "grade is C" to screen

Else IF mk is greater than or equal to sixty

Print "grade is D" to screen

~~Else IF mk is greater than Print "grade is F" to screen~~

~~Print "see you next time semester" to screen~~

~~Print "see you next time semester" to screen~~

Else

Print "grade is F" to screen

Print "see you next semester" to screen

End IF

To save the grades and use them in another thing we type after every Else IF

Set grade to 'A'

$X = A$ and $X = 'A'$ is different

This means

That A is a variable

The computer will search for a

(A is a constant)

This means That

• IF _____

Else IF _____

IF _____

Else _____

IF _____

• End IF
• اذا كلفه الشرط اجل كذا
• اذا لم يلقه الشرط اجل كذا

• End IF
• اذا لم يلقه الشرط
• يخل على IF آخرى
• ويرى ان كلفه الشرط ام
• 8 ثم ينفذ

Note :-

* Conditions can have : And , OR , Not in it

Example:	X	Y	X and Y	X or Y	Not X
	T	F	F	T	F
	F	T	F	T	T
	T	T	T	T	F
	F	F	F	F	T

- If mkt is greater than eighty nine and mkt is less than or equal to a hundred

Print "greater is A" to screen
End IF

Repetition :- (loops)

There are three types of loops :-

- while
- do/while
- for

- Write an algorithm (pseudo-code) to find & print the average grade for a class of ten students

- Set sum equal to zero
- Set count equal to zero

one vsu word

Two things
change

- sum
- count

Lecture 8

- Write an algorithm to calculate & print the average grade for a class of ten students

Set sum equal to zero
 Set **Count** equal to zero → **initial value**
 While count is ~~ten~~ less than ten → **final value (condit**
 Ask user to enter grade → Read grade and save as **grade**
 Add grade to sum
 → **change** (التغيير)

→ **Increment** count by one → **change** (التغيير)

END while

Divide sum by ten and save result as avg

Print avg to screen

How does the computer work :-

$count = 0$ ² < 10 yes
 $gr = 50$
 $sum = 0 + 50 = 50$
 $count = 1$ < 10 → yes
 $gr = 70$
 $sum = 70 + 50 = 120$
 $count = 2$

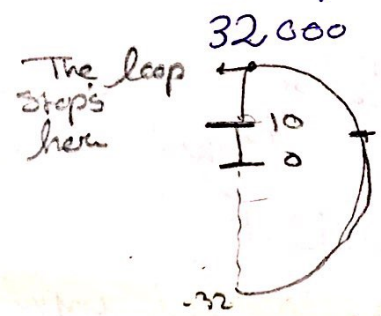
$count = 9$
 $gr = 6$
 $sum = 770 + 6 = 776$

$count = 10$

$avg = \frac{776}{10}$
 $= 77.6$

Note

- if you put increment by 1 it makes a loop of



• Write an algorithm to find the avg grade for a class with an unspecified number of students (0-100).
 Range

set sum equal to zero
 set count equal to zero

~~Sentinel~~

• sentinel:

التي تتوقف عن العمل

• when we don't know the times of loop we use away called priming the pump

Ask user to enter grade or -1 to stop

Read grade and save as gr

while gr is not equal to -1

add grade to sum

Increment count by one

Priming the pump

Ask user to enter grade or -1 to stop

Read grade and save as gr

END while

If count is greater than zero

set avg equal to sum divided by count

Print avg to screen

Else

Print "No grade entered" to screen

ENDIF

Range of input called Sentinel

3

Write an algorithm to find the average grade for a class with an unspecified number of students
Set sum equal to zero
Set count equal to zero

Ask user to enter whether to continue or not (y/n)

Read answer and save as ans → user initial value

While ans is not equal to 'n' → final value

Ask user to enter grade

Read grade and save as gr

Add gr to sum

Increment count by one → change

(The rest is the same)

End while

Range input sentinel *

ITU the computer :-

```
#include <stdio.h>
```

```
int main( )
```

```
{  
  int n1, n2;
```

```
  int sum;
```

Lecture 9

- 1 Ask user to enter first number
- 2 Read number and save as num1
- 3 Ask user to enter second number
- 4 Read number and save as num2
- 5 Set sum equal to num1 plus num2
- 6 Print sum to screen

* include <stdio.h> → Standard input/output header file

int main ()

{
 int num1, num2; → data type
 int sum; → variables

always exists you have to know them by heart

1 printf ("Enter first number \n");

2 scanf ("%i", &num1);

3 printf ("Enter second number");

stops the program

4 **scanf** ("%d", &num2);

• عنوان ذاكرة الذي سوف نضع فيه القيمة

5 sum = num1 + num2;

important Search for Camel Notation

6 printf ("sum = %d", sum);

Assignment statement

return (0); → success

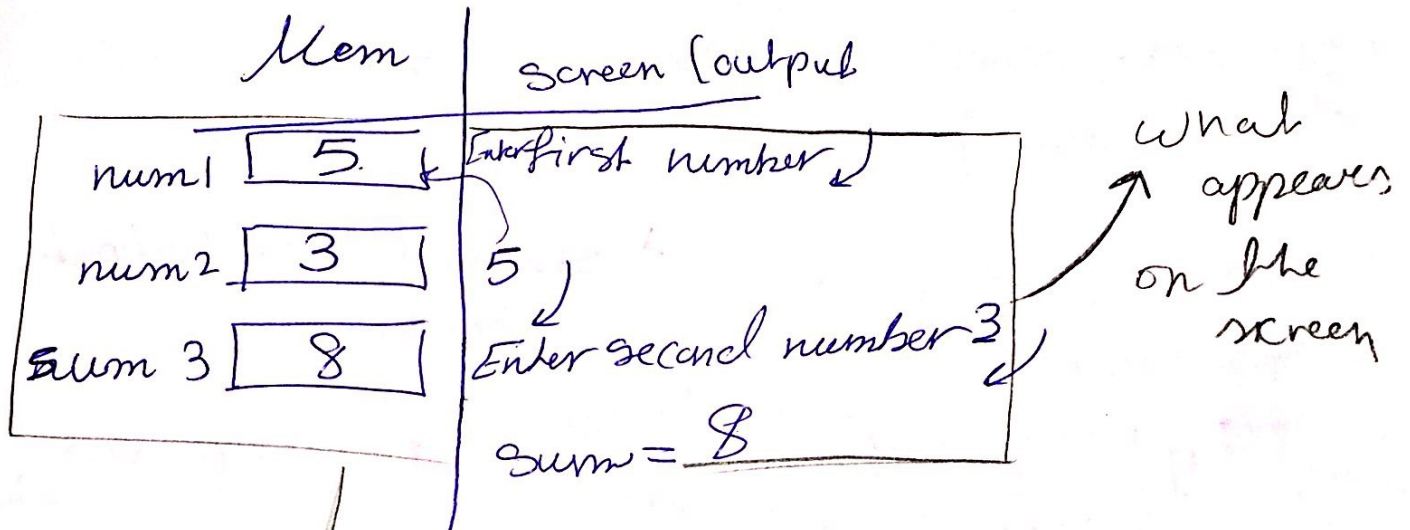
}

Notes :- 1- Since num1 and num2 is the same type we can put them in one line and we end it with (;). If they were different we split them with (;)

2- Variables name can include letters + numbers + underscore But it can't start with a num
 s.a 2nx → wrong / fir_nm → right

→ you can't name variables with key words (reserved words) such as main, int ... (words that exist in the basic sentences of the program)

In The Memory :-



what happens in the memory (we don't see it)

On the computer :-

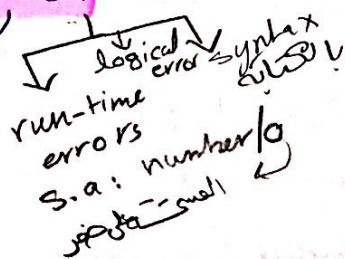
Compile (build)

run (link)

Code Block

• If u made a mistake it either shows an error or a warning

you can't run the program so you should have 0 errors



Try to make them warnings
- sometimes they don't affect your program

But you should correct it Bcz sometimes it's dangerous to leave them

Examples of errors

① $X = 5;$

$$y = X - 4;$$

$$y = y - 1;$$

$$z = 3/y; \Rightarrow \text{run-time error}$$

② if the program is to sum and i put - instead of + \Rightarrow logical error

③ $\text{avg} = (g_1 + g_2)/2;$ \rightarrow you should put ()

$\text{avg} \rightarrow 100 \Rightarrow$

Lecture 10

```

• Constants
#include <stdio.h>
#define PI 3.14
int main()

```

- constants
- usually we use All capitals for constant
- Constant space its value (No =)

```

{ int rad;
  float area;
  printf("Enter radius\n");
  scanf("%d", &rad);
  area = PI * rad * rad;
  printf("Area=%f", area);
  return 0;
}

```

Types

To read and write X :-

if x is	int scanf("%d", &x); printf("%d", x);	float scanf("%f", &x); printf("%f", x);	double scanf("%lf", &x); printf("%lf", x);	Char ^{space} scanf("%c", &x); printf("%c", x);
---------	---	---	--	---

Example

```

int age;
char gender;
printf("Enter age and gender\n");
scanf("%d %c", &age, &gender);

```

When you enter you put a space if the 2 numbers are float

we put a space so it doesn't take the char char "space"

• binary operations
Arithmetic operations

- + $3+2=5$
- $3-2=1$
- * $3 \times 2=6$
- / $3/2=1$
- % $3\%2=1, 6\%10=6$

Precedence Rules

$$X = 5 + 2 * 3 + 4 / 2 * 5$$

$$= 5 + 6 + 2 * 5$$

$$= 11 + 10 = 21$$

* / % } stronger
 + - : weaker

$$X = [(5+2) * (3+4)] / (2*5)$$

$$= (7 * 7) / 10 = 4.9$$

• 1 type casting

```
int x=3, y=2; a;
float z=2.4;
```

- $X = 3/2; \rightarrow 1$
- $Z = 3/2; \rightarrow 1.0$
- $X = 3/2; \rightarrow 1$
- $Z = 3/2.0; \rightarrow 1.5$

لدي فصل على الجوان الكسر
 ① بعد انه يكون النسبة او اللقام
 او الالفت
 مثل 2.0
 ② بعد انه يكون العكس الزرى
 صف فيه كواب float
 مثل Z

$$a = x/y; \rightarrow 1$$

$$b = x/y; \rightarrow 1.0$$

$$a = x/y.0; \rightarrow$$

to make it float for once we use type casting

$$\Rightarrow a = (\text{float}) x/y; \rightarrow 1$$

$$z = (\text{float}) x/y; \rightarrow 1.5$$

$$z = x / (\text{float}) y; \rightarrow 1.5$$

$$z = (\text{float}) x / (\text{float}) y; \rightarrow 1.5$$

$$z = (\text{float}) (x/y); \rightarrow 1.0$$

int to float



Note: • we can use $z = x;$ without type casting

• But $x = z;$;
 ↓
 int

you have to explain that you want z to become an integer

$$x = z \% y;$$

 ↪ int

Example: $V = \frac{4}{3} \pi r^2 h$

$$= \frac{4}{3} * \pi * r * r * h$$

↳ you have to put a zero

• output formatting

52_ 3_ 12345

2305 232

2_ 345_ 67

for a float

By default

$x = 73.624927$ $y = 5.261$ $z = 324.52$

Print f ($\%10.2$ f $\%7.1$ f $\%8.4$ f) $\% \dots$ f
 width ← $\%10.2$ (عبر الخانات (تقريب))
 , x, y, z) width number of digits

• الرقم يجب ان يكتب صحيحه ، اذا لم يبي الخانات يكتب الرقم خارج الخانات

• Comments To write them there is 2 ways

 int _____
 {
 ① /* _____ */

or ② `sum = x + y; // add x to y`

files

#include <stdio.h> يعني هذا البرنامج يعتمد على Data موجوده في فايل اسمه data.txt

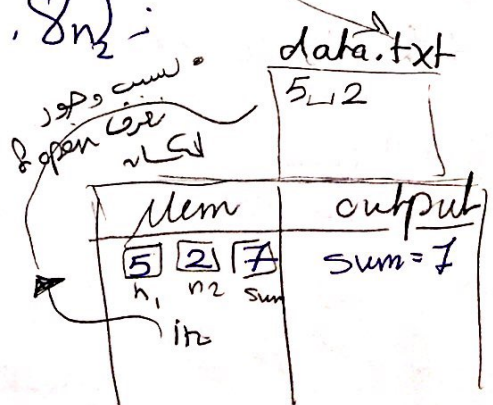
int main()
 { int n1, n2, sum;
 FILE * in; يعني هم مؤشر على فايل (Pointer)

in = fopen ("data.txt", "r"); mode of the opened file
 fscanf (in, "%d %d", &n1, &n2); يعني هذا الملف افتحه للقراءة

sum = n1 + n2;

Print f ("sum = %d", sum);

fclose (in);
 return 0;

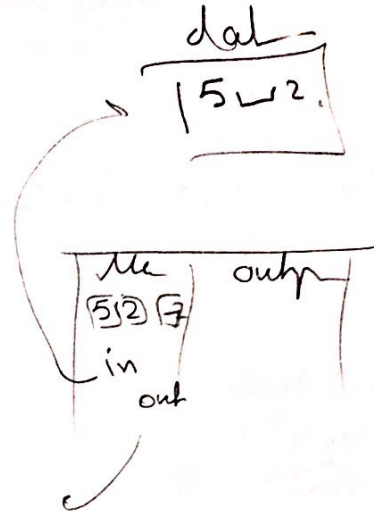


```

#include <stdio.h>
int main ( )
{
  int n1, n2, sum;
  File *in, *out;
  in = fopen ("data.txt", "r");
  out = fopen ("rest.txt", "w");
  fscanf (in, "%d%d", &n1, &n2);
  sum = n1 + n2;
  printf ("sum %d", sum);
  fprintf (out, "sum=%d", sum);
  fclose (out);
  fclose (in);
  return 0;
}

```

مہمہ
 یعنی اذاکہ
 السیر غیر
 مکتوب
 اکتبہ و اعقرو
 بیونہا بکونہ لکھ فارغ
 او معلومات قدری



Lecture 12

• functions

Built starts here.
sum starts here.

include <stdio.h>

int sum (int, int);

• function Prototype

نوع المتغيرات التي
تقبلها (نوع المتغيرات التي
تقبلها)
int, int, sum: int

int main ()

int x, y, s;

printf ("Enter x and y \n");

scanf ("%d %d", &x, &y);

s = sum (x, y);

function call

printf ("sum = %d", s);

return 0;

}

int sum (int a, int b)

{
int result;

result = a + b;

return result;

}

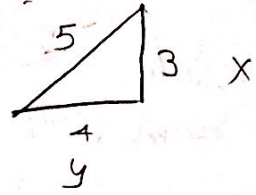
a small program to explain sum

definition

main	sum	output
3 2 5 x y s	3 2 a b	Enter x and y
	5 result	3 2
		sum = 5

Two Types

- 1) system defined functions
- 2) user defined functions



$$z = \sqrt{x^2 + y^2}$$

1) S.d.f.

```
#include <stdio.h>
#include <math.h>
```

```
int main()
{
    int x, y;
    float z;
    printf("Enter x and y\n");
    scanf("%d%d", &x, &y);
    z = sqrt(pow(x, 2) + (y*y));
    printf("z = %.2f", z);
    return 0;
}
```

Include <math.h>

```
Pow (double, double) gives → double
sqrt (double) → double
floor (double) → int
ceil (double) → int
abs (int) → int
fabs (double) → double
```

Si أكبر عدد صحيح
أخذ الرقم الصحيح الذي
يقوم الأبر
2.5 → 3
القيمة المطلقة
int double

$\sin(\text{double}) \rightarrow \text{double}$
 $\cos(\sim) \rightarrow \sim$
 $\tan(\sim) \rightarrow \sim$
 $\sec(\sim) \rightarrow \sim$
 $\cos(\sim) \rightarrow \sim$
 $\cot(\sim) \rightarrow \sim$

2) U.d.f

... $\text{sum}(\text{int}, \text{int}) \rightarrow \text{int}$ ✓

$\text{sum}()$ → int

$\text{sum}(\text{int}, \text{int}) \rightarrow (\text{Nothing}) \text{ void}$ ✓

$\text{sum}()$ → (Nothing) void ✓

Lecture 13

void: لا شيء

```
void
#include <stdio.h>
int sum(int, int);
```

if we put void (لا شيء)
 (int) sum(int, int);

```
int main( )
```

```
{
  int x, y, s; // we cancel s
  printf ("Enter x and y\n");
  scanf ("%d %d", &x, &y);
  sum(x, y);
  printf ("sum = %d", s);
  return 0;
```

```
} void
int sum(int a, int b)
```

```
{
  int result;
  result = a + b;
  return result;
  printf ("sum = %d", result);
}
```

Question for void :-

2 numbers and it prints the two numbers (No output) (output is done by sum Not main)

for the first (No void)

main	sum	output
2 4 5 x y s	2 4 6 a b	Enter x and y 2 4

main	sum	
2 4 x y	2 4 6 a b res	Enter x & y 2 4 sum = 6


```

    ( )
    int sum ( ) ;
    int main ( )
    {

```

مقوم
 Processing
 & input
 هو الذي
 مقوم
 output

() : لا يوجد
 main ()

```

    int s;
    s = sum ( );
    printf ( " sum = %d", s);
    return 0;
}

```

```

int sum ( )
{
  int x, y, result;
  printf ( " Enter x and y \n");
  scanf ( "%d %d", &x, &y);
  result = x+y;
  return result;
}

```

main	sum	output
6 s	2 4 x y 0	Enter x and y sum = 6

```

void + ( ) :-
void sum ( );
int main ( )
{
  sum ( );
  return 0;
}

```

```

void sum ( )
{
  int x, y, result;
  printf ( " Enter x & y \n");
  scanf ( "%d %d", &x, &y);
  result = x+y;
  printf ( " sum = %d", result);
}

```

$$poly = 3x^3 - 2x^2 + 5$$

\swarrow \swarrow \downarrow \searrow
 a b c d

y lip u, x, a, b, c, d نوناغ برخل

```

** Include <stdio.h>
int find_y (int, int, int, int, int);
int cube (int);
int sqre (int);
int main ( )
{
  int a, b, c, d, x, y;
  Printf (" Enter a, b, c, d, x \n");
  scanf ("%d %d %d %d %d", &a, &b, &c, &d, &x);
  y = find_y ( a, b, c, d, x);
  Printf (" y = %d", y);
  return 0;
}

int find_y (inta, intb, intc, intd, intx)
{
  inty;
  y = a * cube(x) + b * sqre(x) + c * x + d;
  return y;
}
  
```



```
int cube (int x)
```

```
{  
  return x * sqr(x) * X;  
}
```

$$2x^3 + 3x^2 - 5$$

```
int sqr (int x)
```

```
{  
  return X * X;  
}
```

main	find-y	cube	sqr	output
$\begin{matrix} 2 & 3 & 0 \\ a & b & c \end{matrix}$	$\begin{matrix} 2 & 3 & 0 \\ a & b & c \end{matrix}$	$\begin{matrix} 1 \\ x \end{matrix}$	$\begin{matrix} 1 \\ x \end{matrix}$	Enter a,b,c d,x
$\begin{matrix} 5 & 1 & 0 \\ d & x & y \end{matrix}$	$\begin{matrix} 5 & 1 \\ d & x \end{matrix}$		$\begin{matrix} 1 \\ x \end{matrix}$	y=0

lecture 14

Selection

```

if (x > 5)
    printf("good");
else
    printf("bad");
    
```

First :-

relational operators :-

- < less than
 - > more (greater) than
 - <= less or equal
 - >= greater than or equal
 - == not equal
- القارنة
 = لا =
 = لا =

Ex:.

```

if (x % 2 == 0)
    printf("even");
    
```

```

else
    printf("odd");
    
```

But if you write it like this:-

```

if (x % 2)
    printf("odd");
else
    printf("even");
    
```

النتج هنا اذا قسم الرقم على 2 وكان الجواب صفر فانه يتغير الى zero / else وينتج الى 1 اما اذا كان الجواب اي شيء غير الصفر فانه يتغير الى True وينتج الى odd

Note: This is wrong
 if (x = 5) should b ==

else

In C language

zero -> false

non-zero -> true

3] If you write:-
 x=2; x=0; x=5
 if (x=2)
 printf("good");
 else
 printf("bad");

It will print good for three cases

if you put:-

```

if (x=0)
    it prints bad
    because
    zero = false
    T.n C
    
```

second:- logical operator :-

- && and
- || or
- ! Not

Ex- If $(age > 20) \&\&(gender == 'f') \parallel (age > 85)$

* It's either all of it is positive or all of it is false

x	y	x && y	x y	!x
T	F	F	T	F
F	T	F	T	T
T	T	T	T	F
F	F	F	F	T

Ex ag=15, gender='f', avg=70

if (F && T || ! (F))

= (F && T || T) it starts with and

= (F || T)

= T

- !() is the most important
- && is more important than ||
- parenthesis () is the most important

if (F && { T || ! (F) })

(F && { T || T })

(F && T)

```

if (num >= 0);
    printf ("%d is positive", num);
else
    printf ("%d is negative", num);

```

```

Ex: if (mark >= 90); && (mark <= 100)
    {
        printf ("grade is A\n");
        printf ("good job\n");
    }

```

* This means all the conditions should be true to take the result

```

else if (mark >= 80)
    printf ("grade is B\n");

```

• It's a one if so it goes to each condition and if one is true it stops.

```

else if (mark >= 70)
    Print ----- is C
else if (mark >= 60)
    ~ ----- is F

```

Short circuits

if (X=5; y=10; && (y < 20))

if (X > 6) _F good _T y = 10; _T (y < 20)

else bad

printf ("%d\n", y);

if X=15

Now if it was y=0 it is false so T && F = F it prints bad and 0

It will print good and 20

* a program to know whether a letter is a vowel or not:-

```
char letter;
```

```
printf("Enter letter\n");
```

```
scanf("%c", &letter);
```

```
if (letter == 'a' || 'o' || 'i' || 'u' || 'e')
```

```
printf("%c is a vowel, letter);
```

```
else
```

```
printf("%c is not a vowel, letter);
```

Those are not vowels
so it's All T

This is wrong

you write :

```
if (letter == 'a' || letter == 'o' || letter == 'i' || letter == 'u' || letter == 'e')
```

Lecture 15

```

if (num == 1)
    printf ("one\n");
else if (num == 2)
    printf ("two\n");
else
    printf ("No such Number\n");
    
```

Questions:
 * Gives you switch and wants if for the same case or opposite

using switch

```

switch (num)
{
    case 1: printf ("one\n");
            break;
    case 2: printf ("two\n");
            break;
    default: printf ("No such number\n");
}
    
```

Notes
 • It works only with == (Doesn't work with < >)
 • ~ ~ ~ things that we know what (it doesn't work with float & string)
 Comes after or before
 ↳ s.a : integers & Ascci code (a,b,c...)

break is important to stop the program
 ↳ after default you can put break but no need bcz the program ended any way

Exp: Vowels using switches:

```

switch (letter)
{
    case 'a': case 'i': case 'o': case 'e': case 'u':
    
```

it doesn't have to be here break;
 ↳ printf ("%c is a vowel", letter);
 default: printf ("%c is not a vowel", letter);
 }

• x y
 if (x > y)
 printf ("%d is larger", x);
 else
 printf ("%d is larger", y);

• x y z a b

• we assume that the first number entered is the min

•

x	y	z	a b
5	2	4	3 2

↳ assuming $x = \text{min};$

if (y < min)

min = y;

if (z < min)

min = z;

• you can't use switch in this case

• you need a loop

Ex: A program to print the number that is between

→

x	y	z
2	1	7

 it has to print 2

if ((x > y) && (x < z)) || ((x > z) && (x < y))
 printf ("%d", x);

else if ((y > x) && (y < z)) || ((y > z) && (y < x))
 printf ("%d", y);

else print ("%d", z);

Ex: Enter a formula & get the answer
 input → 5+2
 output 5+2=7

```
printf("Enter formula");
scanf("%d %c %d", &n1, &op, &n2);
```

```
switch(op)
{
    case (+) : result = n1+n2;
              break;
    case (-) : _____
              |
              |
}
printf("%d %c %d = %d", n1, op, n2, result);
```

on the screen
 Enter formula
 5+3
 5+3=8

Nested if:-

```
x=5, y=3;
if (x > y);
    if (x == 5)
        printf("good\n");
    else
        printf("bad\n");
else
    printf("bye\n");
{
```

adding
 else for the first if

• on the screen after addy

good	x=5, y=3 good	x=5, y=3 good bye	x=5, y=3 good bye
-	x=5, y=8 bad bye	x=5, y=8 bye	x=5, y=8 bad bye
-	x=3, y=5 bad bye	x=3, y=5 bye	x=3, y=5 bad bye
bad bye	-	x=7, y=5 bad bye	x=7, y=5 bye

Lecture 16

Boolean function 2

لا ترجع ^{عن} function
و لا ترجع جواب زرع أو لا
true or false

int isEven(int n) ← it's like asking is this even?

```
{
  if (n%2==0)
    return 1;
  else
    return 0;
}
```

```
main ||
{
  int x=5;
  if (isEven(x))
    printf("Even");
}
```

Another way

```
int isEven(int n)
{
  if (n%2)
    return 0;
  else
    return 1;
}
```

The shortest way :-

```
int isEven(int n)
{
  return ! (n%2);
}
```

* Loops :-

while
for
do/while

• 3 Basic Components of loops :-

Ex: int x=2

while (x<=5)

{ printf("hi\n");

x++;

}

1- initial value

2- final value

3- change

2 hi

3 hi

4 hi

5 hi

6 ?

Ex1 X=5;

X++;

printf ("%d", X); it prints 6

Ex2 X=5;

++X;

printf ("%d", X); it prints 6

Ex3 X=5;

Y = ++X

→ Add 1 to X and Then do everything

printf ("%d", X); it prints 6

But y = 6

Ex4 X=5;

Y = X++

→ Do everything Then add 1

printf ("%d", X); 6

But y = 5

Ex5 X=5;

printf ("%d", X++);

it prints 5 Because he does the print then he adds one.

Ex6 X=5;

printf ("%d", ++X);

it adds one then it prints
So it prints 6

you can use the ++, -- way for operations : s.a:-

$$X = X + 7; \equiv X += 7;$$

$$X = X * 2; \equiv X *= 2;$$

$$X++; \equiv X += 1;$$

Ex:- factorial :-

n!
5 → 5! ⇒ we need a loop!

$$5 \times 4 \times 3 \times 2 \times 1$$

$$\text{or } 1 \times 2 \times 3 \times 4 \times 5$$

i = n (initial value)

In case your going down

while (i >= 1)

```

{
  result = result * i; // result *= i;
  i--;
}

```

- result = 1
i = 5
- result = 1 * 5 = 5
i = 4
- result = 5 * 4 = 20
i = 3
- result = 20 * 3 = 60
i = 2
- result = 60 * 2 = 120
i = 1
- result = 120 * 1 = 120

• in case your goal is up

```

i = 1
while (i <= 5)
{
    result = result * i; // result *= i;
    i++;
}

```



Ex: x^y

```

result = 1;
i = 1;

```

$2^3 = 2 * 2 * 2$

```

while (i <= y)
{

```

```

    result = result * x;

```

```

    i++;
}

```

$3^4 \Rightarrow x=3, y=4$

```

i = 1
result = 1 * 3 = 3
i = 2
result = 3 * 3 = 9
i = 3
    = 9 * 3 = 27
i = 4
    = 27 * 3 = 81

```

• if i wanna make it as a function

```

int myPow (int x, int y)

```

```

{
    int result = 1;
    int i = 1;
    while (i <= y)

```



```

{
    result = result * x;
    i++;
}
return result;
}

```

int z = pow(5, 7);



```

sum = 0;
i = 0;
while (i < 10)
{
    printf ("Enter grade \n");
    scanf ("%d", &grade);
    sum += grade;
    i++;
}

```

avg = (float) sum / 10;

```

printf ("Enter grade or -1 to stop \n");
scanf ("%d", &grade);
while (grade != -1)
{
    sum += grade;
}

```

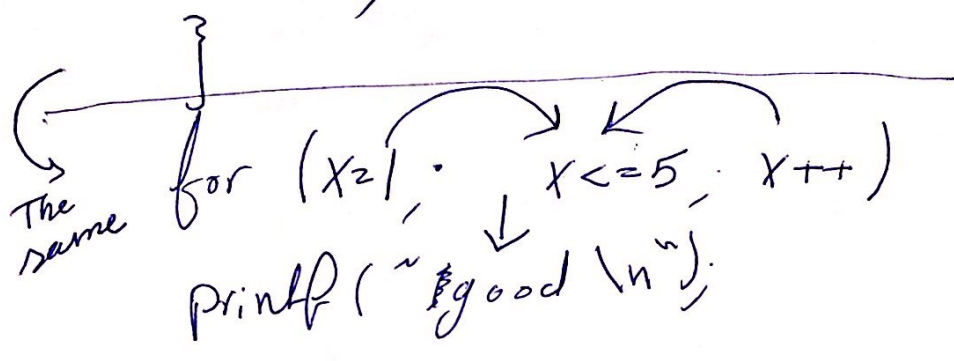
Lecture 17

for loops

```
x = 1;
while (x <= 5)
{
    printf ("good\n");
    x++;
}
```

Types of loops

- while
- for
- do/while



• If more than one variable controls the loop

```
x = 1; y = 10;
while ( (x <= 5) && (y >= 5) )
{
    printf ("hi\n");
    x++
    x++;
    y--;
}

```


in the same way:-

```
for ( x=1, y=10; ((x<=5) && (y>=5)) ; x++, y-- )  
    printf ("hi\n");
```

X^y

```
result = 1;  
i = 1;  
while (i <= y)  
{  
    result *= x;  
    i++;  
}  
printf ("result = %d", result);
```

in the same way:-

```
for ( result = 1, i = 1; i <= y; i++)  
    result *= x;  
printf ("result = %d", result);
```

```
Ex: for (i=1; i <= 1000; i++)
```

```
    printf ("good\n");
```

• في هذا طبع good مرة واحدة

• Pause للتحقق من الحالة التي نريتها في
مرة واحدة من كل output

this means
don't do anything

do/while

$X = 12547;$

Sum = 0;
Count = 0;

while (X > 0)

{ digit = X % 10;
X = X / 10;

Count ++;

Sum = Sum + digit

}

printf ("Count = %d", Count);

~ printf ("Sum = %d", Sum);

برنامج حَسِب عدد الخانات

• How it works

$X = 12547$
Count = 0

$X = 1254$
Count = 1

$X = 125$
Count = 2

$X = 12$
Count = 3

$X = 1$
Count = 4

$X = 0$
Count = 5

if you want to add digits

Using do while :-

$X = 12547;$

Count = 0;

do

{ $X = X / 10;$
Count ++;

} while (X > 0);

• نضعه في الشرط
في آخر الحلقة

• do/while is the only one that

you have to enter the loop for at least once

• Ex

$i = 1;$
while ($i \leq 100$)

or ($i < 100$)

• برنامج يطبع فواصل العدد 100


```

{
  if (n % i == 0)
    printf ("%d is not prime\n", n);
  i++;
}

```

```

i = 2;
while (i < n)
{

```

```

  (n % i == 0)
    printf ("%d is not prime\n", n);
  else
    printf ("%d is prime\n", n);
  i++;
}

```

برنامج تصفية الأعداد الأولية.

we use a technique called: flag technique
 ↳ you don't print till you make sure

```

Prime = 1; (True)

i = 2;
while (i < n)
{
  if (n % i == 0)
    Prime = 0;
  i++;
}

```

Lecture 18

break & continue.

```

X=3;
while (X<7)
{
    printf ("%d\n", X);
    if (X==5)
        break; continue
    X++;
}
printf ("bye\n");

```

لا تتوقف عند الشرط.
 it means
 انما تتوقف عند الشرط
 اوقف ال loop
 it prints 3 4 5 5
 infinite loop
 = it prints 3 4 5 bye

Example:-

```

X=3;
while (X++<10)
{
    printf ("%d\n", ++X);
    if (X==7)
        break; continue;
    printf ("%d\n", X);
}
printf ("%d\n", ++X);

```

break - it prints	* continue - it prints
5	5
5	5
7	7
<hr/> 8	<hr/> 9
	9
	11
	<hr/> 11
	13

```

int i, prime, n;
prime = 1;
for (i=2; i < n; i++)
    if (n%i == 0)
        {

```



```

prime = 0;
break;
}

```

```

if (prime)
    printf ("%d is prime", n);
else
    printf ("%d is not prime", n);

```

Versteel loop :- (added by)

• They don't have to be the same loop

Prime function :-

```

#include <stdio.h>
int isPrime (int);
int main ( )
{
    int i;
    for (i = 1; i <= 10000; i++)
        if (isPrime(i))
            printf ("%d\n", i);
    return 0;
}

```

```

int isPrime (int n)
{
    int i = 2;
    while (i < n)
    {
        if (n % i == 0)
            return 0;
        i++;
    }
}

```

```

return 1;
}
}
for (i = 2; i < n; i++)
    if (n % i == 0)
        return 0;
return 1;

```

ampli: Give me the perfect numbers (number = $2^k - 1$)

$6 = 1 + 2 + 3$ مثال
 $22 = 1 + 2 + 4 + 7 + 14$

Example:-

$n = 4;$

```

*
* *
* * *
* * * *
    
```

To print those:-
 for (i=1; i<=n; i++)
 printf ("%*s", i, "*");

To print

```

* * * ← Rows
* * * ←
* * * ← ←
↑ ↑ ↑
columns
    
```

```

for (i=1; i<=n; i++) i
    for (j=1; j<=n; j++)
        printf ("%*s", j, "*");
    printf ("\n");
}
    
```

```

i=1
* * *
i=2
* * *
i=3
* * *
    
```


To print *
* *
* * *

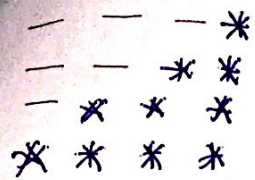
(Added by)

To print * * *
* *
*

To print - - *
- * *
* * *

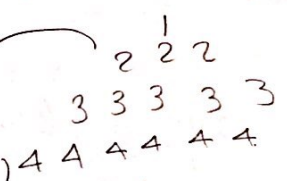
```
for (i=1; i<=n; i++)  
{ for (j  
    printf (" " " ");  
    for (k  
        printf (* * " ");  
    printf ("n" );  
}
```

Lecture 19

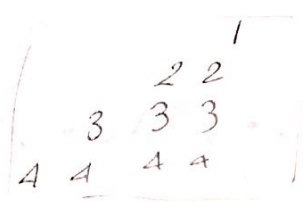
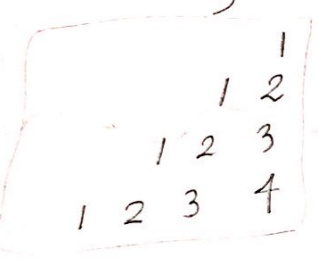


```

for (i=1; i<=n; i++)
{
    for (j=n; j>1; j--)
        printf(" ");
        printf(" ");
        for (k=1; k=i; k++)
            printf(" *");
        printf("%d", i);
        printf("\n");
    }
}
    
```



you print k



you print i

printf("%d", i) instead of printf(" *")

files and loops

```

int status;
status = scanf("%d", &x);
    
```

• you can use scanf to stop (when something is odd and not known)

```

-1
status = fscanf(FILE, "%d", &x);
    
```




```
float avg;
int g1, g2, status;
```

• برنامج يستقبل علامات من File
و يتوقف عندما يطلب منه ذلك

```
FILE * in;
```

```
in = fopen("data.txt", "r");
```

```
status = fscanf(in, "%d%d", &g1, &g2);
```

• الملف فارغ
او غير موجود
يرجع EOF

```
while (status != EOF)
```

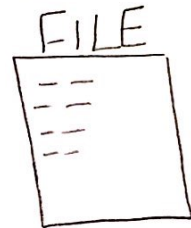
```
{
```

```
avg = (float)g1 / g2;
```

```
printf("avg = %.2f", avg);
```

```
status = fscanf(in, "%d%d", &g1, &g2);
```

```
}
```



Pointers := dynamic addresses

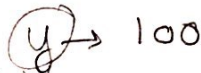
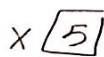
العنوان

```
int x=5, z=30
```

```
int *y = &x;
```

نوع Pointer

في الذاكرة: لها نفس عنوان مكان
و عنوان



```
printf("%d", &x);
```

طبع address وليس

ليس القيمة

و هنا يطبعها 100

```
y = &z;
```

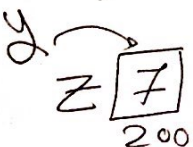
• لو كتبت
هنا تغير

• و تؤثر على z

• نحن في متغير

متار

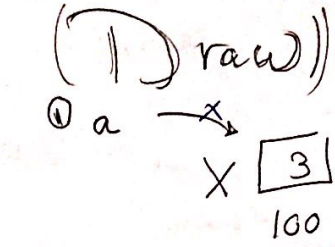
• الفرقة
ثابتة
ولكن في متغير



• يطبع 200

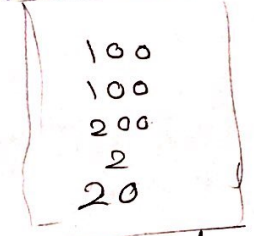
What is the output? ②

```
int x=3, y=2;
int *a = &x, *b;
```



```
printf("%d\n", a);
printf("%d\n", &x);
b = &y;
printf("%d\n", b);
```

• what it prints (answari)



```
a = b;
printf("%d\n", *b);
*a = 20;
printf("%d\n", y);
```

• كمنطق
عن * int
تسمى قيمة
indirection
• رجوع العنصر
• ما نؤشر عليه
b
• رجوع لطبع 2
• address الرجوع 8

Notes:-
int
a = x; خطأ
لأن a و x ليسا
نفس النوع

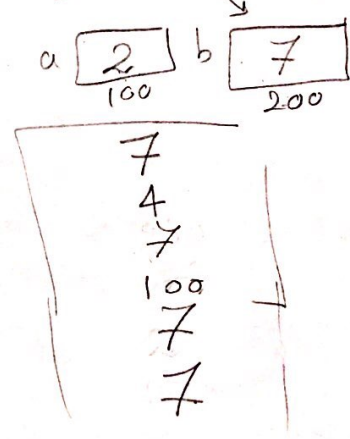
```
a = &x
```

رجوع للـ لايشين
Pointers

What is the output?

```
int a=2, b=7;
int *c = &b, *x;
printf("%d\n", *c);
*c = a+2;
printf("%d\n", b);
x = &a;
*x = *x + 5;
printf("%d\n", a);
printf("%d\n", x);
```

```
a = b + a;
printf("%d\n", *x);
```



lecture 20

توزيع ال pointers

function return

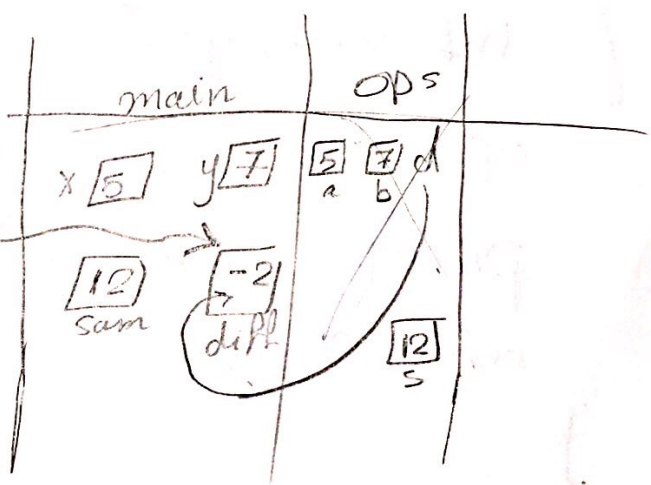
```
#include <stdio.h>
int ops ( int, int, int * );
int main ( )
{
    int x = 5, y = 7, sum, diff;
    ops ( x, y, &sum, &diff );
    printf ( "sum = %d\n diff = %d", sum, diff );
    return 0;
}
```

واحد
دسكن تستطيع الاعطاء
اكثره من
عن طريق ال pointers
سكن ترجع كتر من
(بالرجوع)

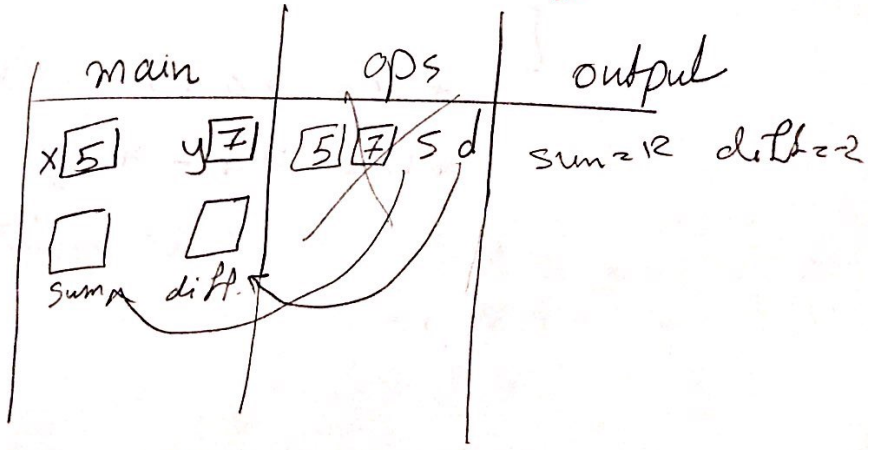
arguments

ops (x, y, &sum, &diff)

```
void ops ( int a, int b, int *d, int *s )
{
    int s;
    s = a + b;
    *d = a - b;
    return s;
}
```

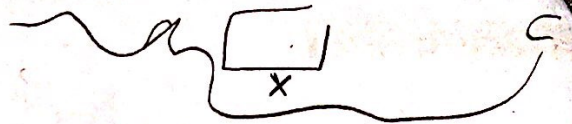


الغالبية كذلك



scanf :-

```
int scanf ( —, int *c)
```



• إذا ما حظت $\&$ ليحل ال Trash Pointer ، حين لا يؤخذ ما داخل المربع بل ليذهب الي مكان غير معروف في الذاكرة

• مع $\&$: يؤشر على المربع ، اذا حظت 5 صحتها في المربع

• There is a Void * C (Void pointer) علامة لتسجيل البعد

```
#include <stdio.h>
Void ops (int *, int *);
int main ( )
{
    int x=5, y=7;
    ops (&x, &y);
    printf (" sum = %d \n diff = %d \n", x, y);
    return 0;
}
```

Important

main	ops	output
		sum = 12 diff = -2

```
Void ops (int *a, int *b)
```

```
{
    *a = *a + *b ; wrong
    *b = *a - *b;
    int x = *a , y = *b;
    *a = x + y;
    *b = x - y;
}
```


What is the output?

```
#include <stdio.h> :-
```

```
int one ( int, int *, int * );
```

```
void two ( int *, int );
```

```
int main ( )
```

```
{  
  int x=5, y=2, z;
```

```
  z = one ( x, &y, &x );
```

```
  printf ( "%d %d\n", x, y );
```

```
  two ( &z, x );
```

```
  printf ( "%d %d", x, z );
```

```
  return 0;
```

```
}
```

```
int one ( int x, int * a, int * y )
```

```
{
```

```
  *a = x + *y;
```

```
  printf ( "%d %d\n", *a, x );
```

```
  (*y) ++;
```

```
  return *a + *y;
```

```
}
```

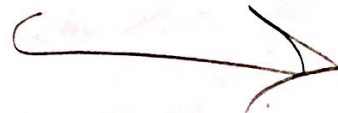
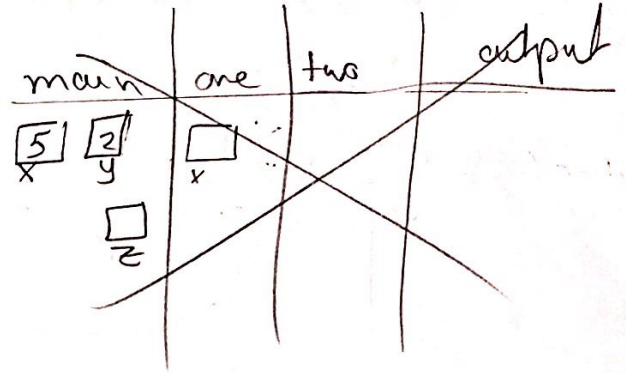
```
void two ( int * a, int * b )
```

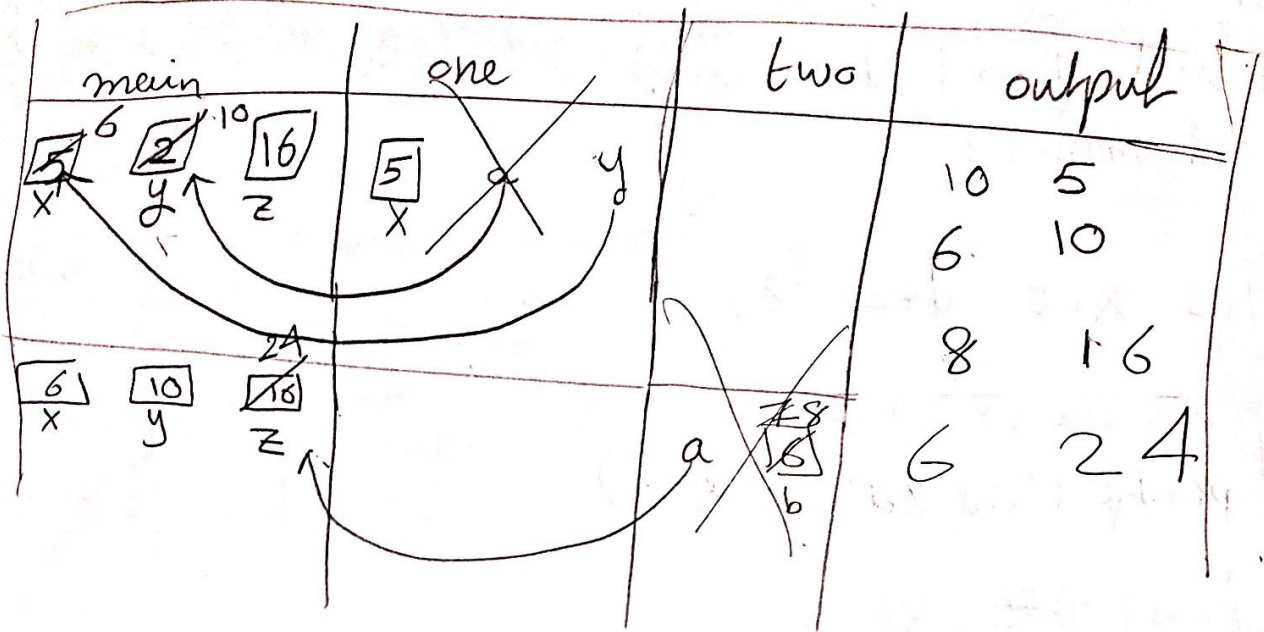
```
{
```

```
  b ++;
```

```
  printf ( "%d %d", ++*b, *a );
```

```
  *a = *a + b;
```





lecture 20

* global vs local variables

```

#include <stdio.h>
int x = 5;
int one (int);
int main ( )
{
    int y;
    y = x;
    ;
}

```

global variable

```

int one (int b)
{
    int x;
    x = 10;
}

```

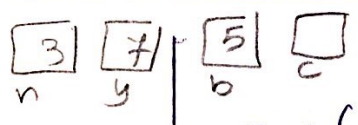
• لو في متغير اسمه
الكل (x) متغير
فانه يتغير عليه ويستعمل هو
• لو ما كان في ، ستعمل الـ global
• لو x int ما كانت موجودة
تغير قيمة الـ global وتطلع
10 من 5

what is the output?

```

#include <stdio.h>
int n = 3;
int first (int *, int);
void sec (int *, int, int *);
int main (.)
{
    int y = 7, b = 5, c;
    c = first (&b, n);
}

```



```

19
sec (&y, first (&c, c),
    &b);
printf ("%d %d %d %d",
    n, y, b, c);
return 0;
}

```

```
int first(int *a, int b)
```

```
{
  int n = *a;
  b++;
  *a = n + b;
  return b + n;
}
```

```
void sec(int *x, int y, int *z)
```

```
{
  *x = ++n;
  *z = *x + y;
  printf("%d %d %d\n", y, n, *z);
  (*z)++;
}
```

global	main	first	sec	output
<div data-bbox="76 1265 183 1332" style="border: 1px solid black; padding: 2px;"> 3 4 n </div>	<div data-bbox="87 1332 391 1489"> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 2px;">7</div> <div style="border: 1px solid black; padding: 2px;">3</div> <div style="border: 1px solid black; padding: 2px;">9</div> </div> <div style="display: flex; justify-content: space-around; margin-top: 5px;"> y b c </div> </div>	<div data-bbox="486 1288 710 1534"> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 2px;">3</div> <div style="border: 1px solid black; padding: 2px;">4</div> </div> <div style="display: flex; justify-content: space-around; margin-top: 5px;"> a b </div> <div style="border: 1px solid black; padding: 2px; margin-top: 10px; width: 40px;">5</div> <div style="text-align: center; margin-top: 5px;">n</div> </div>	<div data-bbox="774 1321 853 1433" style="border: 1px solid black; padding: 2px; width: 40px;">19</div> <div style="text-align: center; margin-top: 5px;">d</div>	<div data-bbox="1053 1265 1500 1422"> 19 5 24 5 5 25 19 </div>
<div data-bbox="55 1624 167 1736" style="border: 1px solid black; padding: 2px;">4</div> <div style="text-align: center; margin-top: 5px;">n</div>	<div data-bbox="183 1624 422 1736"> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 2px;">7</div> <div style="border: 1px solid black; padding: 2px;">9</div> <div style="border: 1px solid black; padding: 2px;">9</div> </div> <div style="display: flex; justify-content: space-around; margin-top: 5px;"> y b c </div> </div>	<div data-bbox="486 1601 710 1848"> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 2px;">9</div> <div style="border: 1px solid black; padding: 2px;">10</div> </div> <div style="display: flex; justify-content: space-around; margin-top: 5px;"> a b </div> <div style="border: 1px solid black; padding: 2px; margin-top: 10px; width: 40px;">9</div> <div style="text-align: center; margin-top: 5px;">n</div> </div>		
<div data-bbox="95 1892 183 1982" style="border: 1px solid black; padding: 2px;">5</div> <div style="text-align: center; margin-top: 5px;">n</div>	<div data-bbox="191 1848 422 2004"> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 2px;">5</div> <div style="border: 1px solid black; padding: 2px;">24</div> <div style="border: 1px solid black; padding: 2px;">19</div> </div> <div style="display: flex; justify-content: space-around; margin-top: 5px;"> y b c </div> </div>	<div data-bbox="742 1848 965 1982"> <div style="border: 1px solid black; padding: 2px; width: 40px;">19</div> <div style="text-align: center; margin-top: 5px;">z</div> </div>		

Pointers

Dynamic address

How to define it?

$\text{int } *y = \&x;$
 ↑ نوع المؤشر ↑ الموقع الذي يؤشرون عليه

5

x

Notes:
 if a is a pointer and x is a variable
 Then you can't say $x = a$
 but you can say $a = \&x$

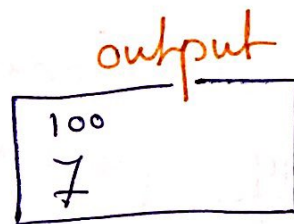
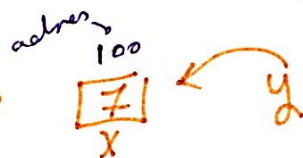
What is the Output?

If you have a pointer $*y =$

$\rightarrow \text{printf}("%d", y);$ → it prints the address that y is pointing to it.

$\rightarrow \text{printf}("%d", *y);$ → it prints the value inside the address

Ex $\text{int } x = 7;$
 $*y = \&x;$



$\text{printf}("%d", y);$
 $\text{printf}("%d", *y);$

How to use pointers in functions?

Remember that a function can return one value only
 But we can use pointers to return more than one value from a function.

How?

first

The function should be void / Example: function to calculate sum/multiplication/division & subtraction
 $\text{void ops}(\text{int}, \text{int}, \text{int}^*, \text{int}^*, \text{int}^*, \text{int}^*)$

You can add as many as you want

Second :-

When Calling the function :-

ops (x, y, &sum, &multi, &div, &sub)

Third :-

In the function itself :-

```
ops (int a, int b, int *d, int *s, int *f, int *e)
{
    *d = a + b;
    *s = a * b;
    *f = a / b;
    *e = a - b;
}
```

Note

When it comes to printing when you have many functions
; let's say we have 2 functions : sum and sub

```
Z = sum(x, y, s);
printf("%d", Z, x);
```

```
E = sub(M, F);
```

```
printf("%d", M, F);
```

you work out Z
if there is another
print in the function
it comes first

— This This

same

lastly this

Global vs local variables

- If you define a variable before main function, it's a Global variable

```
Ex #include _____  
    int x=5  
    {  
    }  
    int main( )
```

- in functions, if there is a variable same as a global variable then :-

```
int one (int b)
```

```
{  
  int x
```

```
  x=10;  
}
```

لو كانت متش موجودة
لنأخذ قيمة global

You take this value.
But if it didn't say its value
then you take x=5

lecture 26

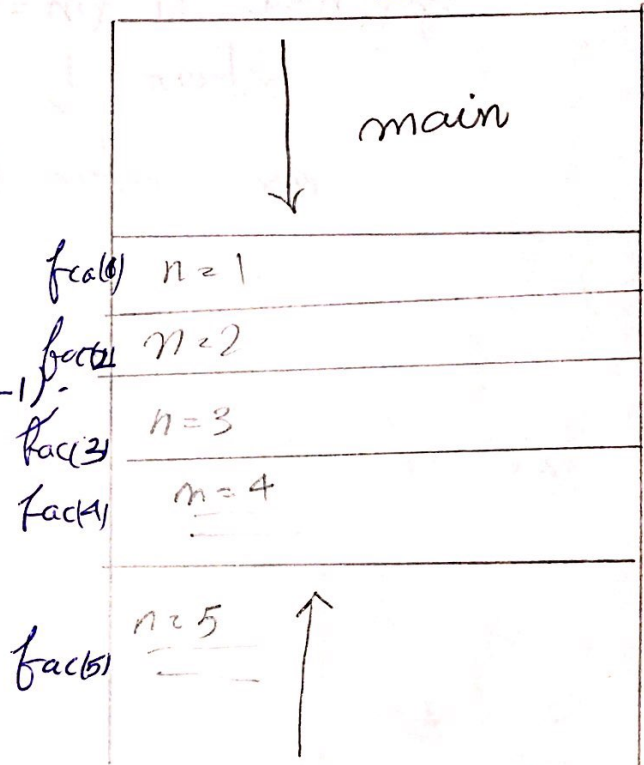
Recursion

$$\begin{aligned}
 \text{fac}(5) &= 5 * 4 * 3 * 2 * 1 \\
 &= 5 * \text{fac}(4) \\
 &= 4 * \text{fac}(3) \\
 &= 3 * \text{fac}(2) \\
 &= 2 * \text{fac}(1) \\
 &= 1
 \end{aligned}$$

Stack: مكتبة
تحتوي على 3 من فوق
حتى 3 من تحت
LIFO
last in First out

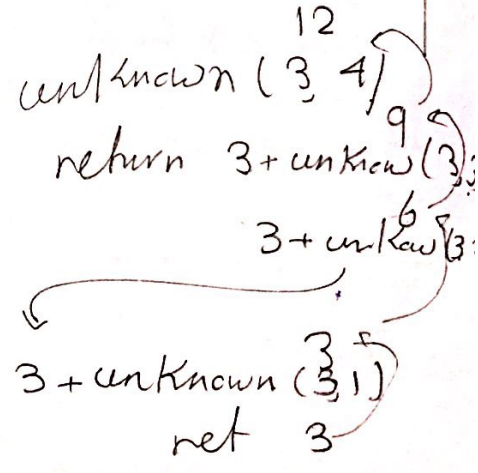
```
int fac (int n)
```

```
{
  if (n == 1)
    return 1;
  else
    return n * fac (n-1);
}
```



```
int unknown (int x, int y)
```

```
{
  if (y == 1)
    return x;
  else
    return x + unknown (x, y-1);
}
```



fibonacci :-

	1	1	2	3	5	8	13	21	---
n=1	2	3	4	5	6	7	8	---	---

int fib(int n)

{

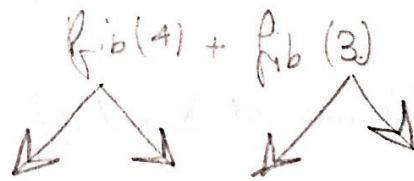
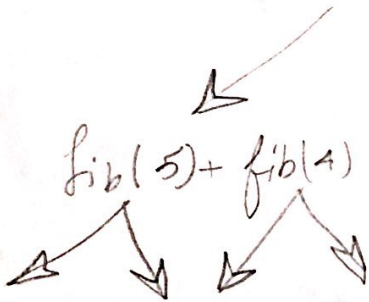
if ((n==1) || (n==2))
return 1;

else return (fib(n-1) + fib(n-2));

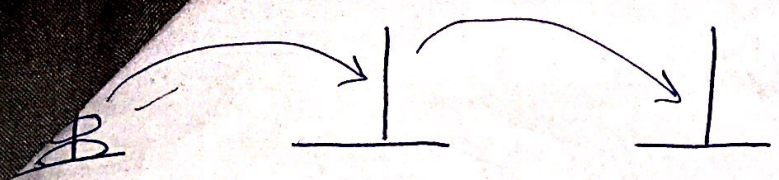
• لهذا الزناح بعض جواب، ولكن يأخذ وقت طويل

Ex:- fib(7)

fib(6) + fib(5)



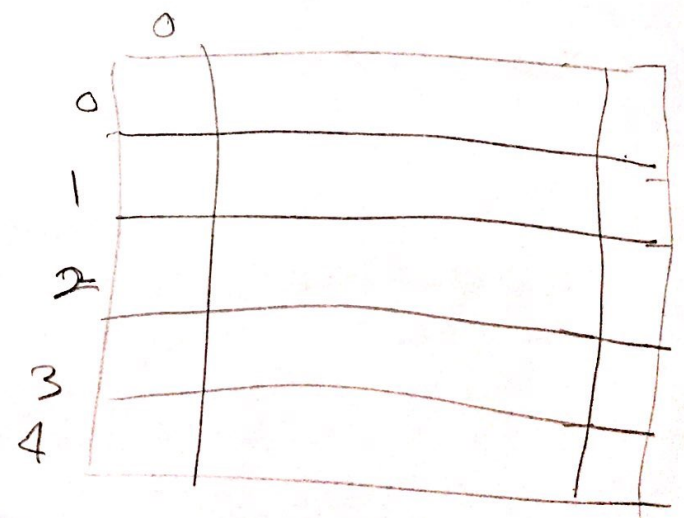
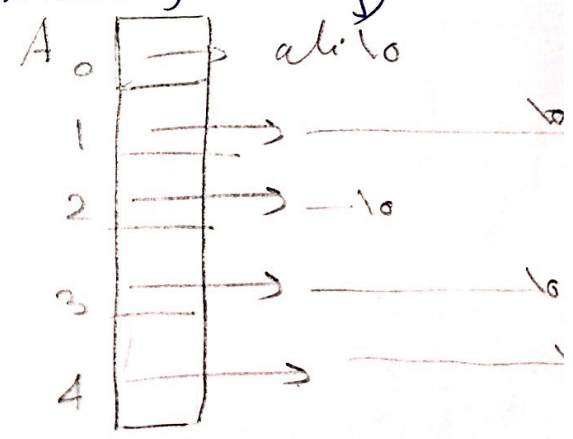
Recursion is loop ال .



5 2 3 4 16
 to print
 5 2, 3 4 16

• array of pointers

char * A[5] = {"ali", "Ahmad", ...}



Parallel Arrays :-

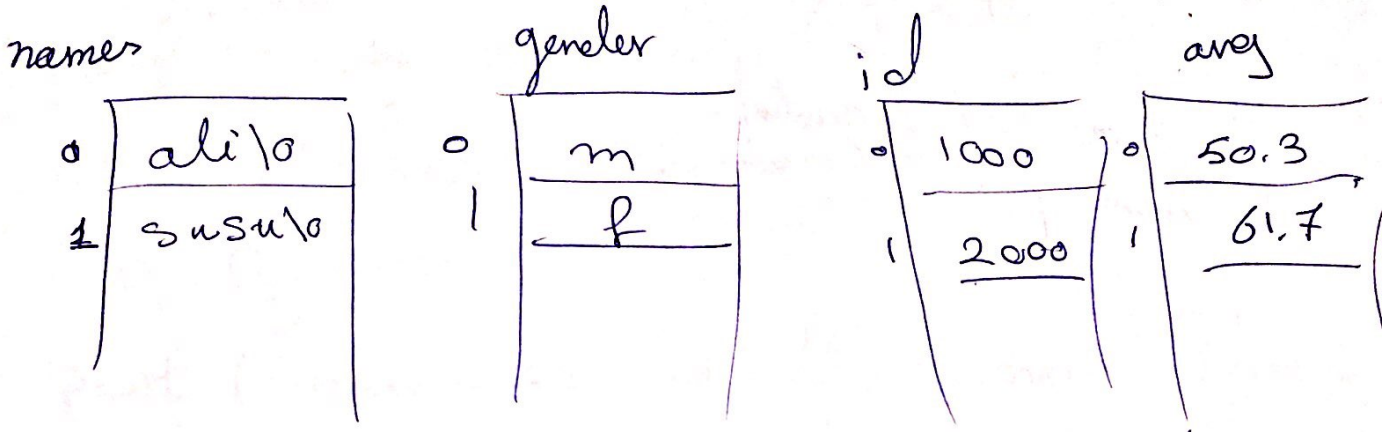
names	grades	ids	gender
0 Ali	70 30 100	1000	m
1 Saw san	60 90 10	2000	f
2			
3			
4			

avg (float)

6.35

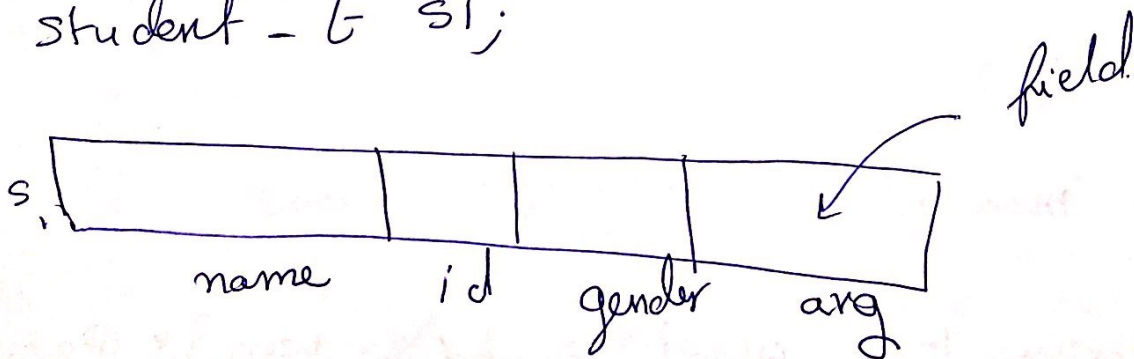
Lecture 27

Structures :-



```
typedef struct  
{  
    char name[10];  
    int id;  
    char gender;  
    double avg;  
}  
student_t;
```

```
student_t s1;
```



```
strcpy (s1.name, "ahmed");  
s1.id = 1000;  
s1.gender = 'm';  
s1.avg = 70.7;
```


Array of structures - students [10];

students

0	name	id	gender	-
1	name	id		

```
#include <stdio.h>
#define S 10
typedef struct
{
    char name[S];
    int id;
} person_t;
```

x y;

int main ()

```
{
    person_t p1 = { "ahmad", 1000 }; P2;
```

$P_2 = P_1$; ✓ You can do that

$\&P_1 == P_2$ ✗ You can't do that:

```
printf ("Enter name & id in");
```

```
scanf ("%s%d", P2.name, &P2.id);
```

```
P1 = P2;
```

```
printf ("name = %s id %d", P1.name, P1.id);
```

```
return 0;
```

```
}
```

P₁

Salem 10	2000
ahmad 10	1000
name	id

P₂

Salem 10	2000
name	id

output

Enter name _____

Salem 2000

Using a function

same definition

```
void print-person (person-t);
```

```
int main ()
```

```
{
```



```

person_t p1 = { "ahmed", 1000 };

```

```

print-person (p1);

```

```

return 0;

```

```

}

```

```

void print-person ( person_t p )

```

```

{

```

```

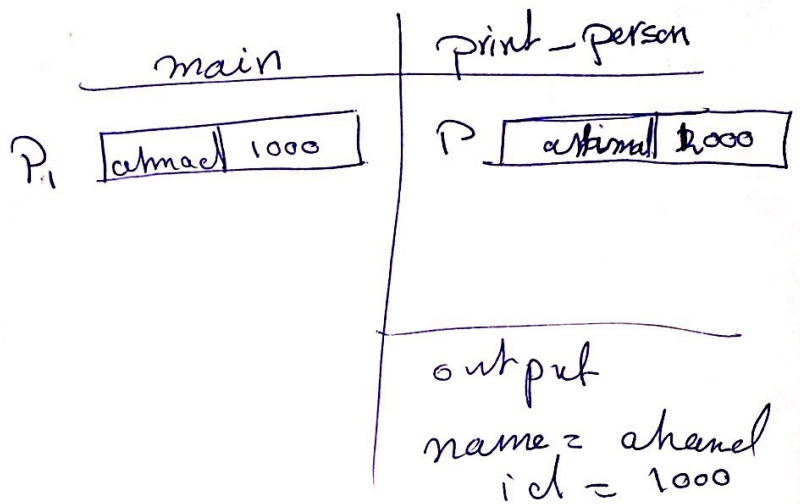
printf ( " name = %s id = %d\n", p.name, p.id );

```

```

}

```



```

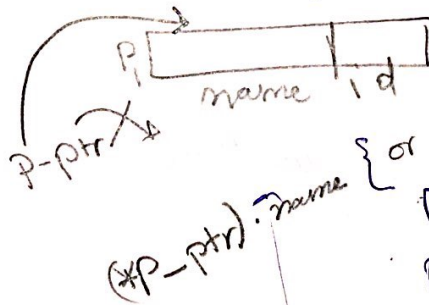
#include <stdio.h>
typedef struct
{
    char name[50];
    int id;
} person_t;

```

```

person_t get-person ();
int main ()

```



```

{
    person_t p1;
    P_ptr = &p1;
    P1 = get-person();
    return 0;
}

```

```

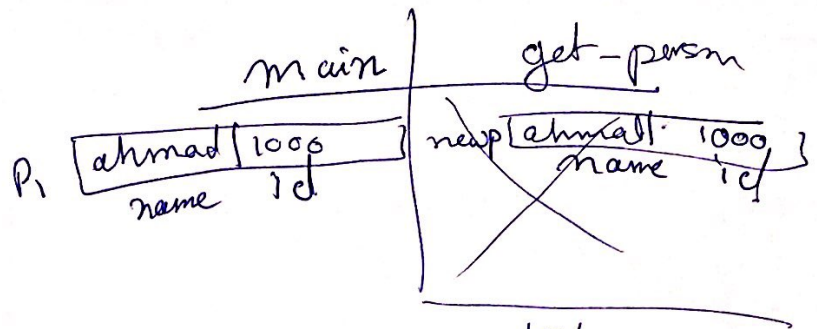
person_t get-person()
{
    person_t newp;
    printf ( "Enter name & id\n" );
}

```

```
if ("%s %d" , newp.name , &newp.id )
```

```
return newp;
```

```
}
```

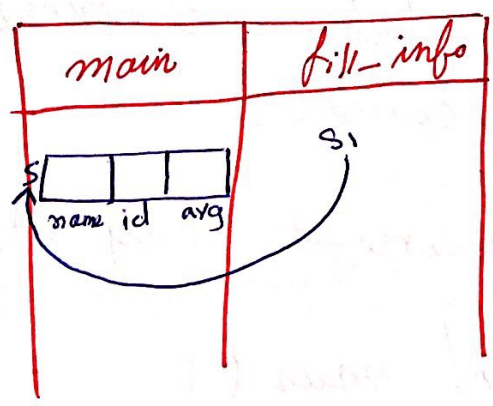


outph

Enter name & id
ahmad 1000
←

Lecture 28

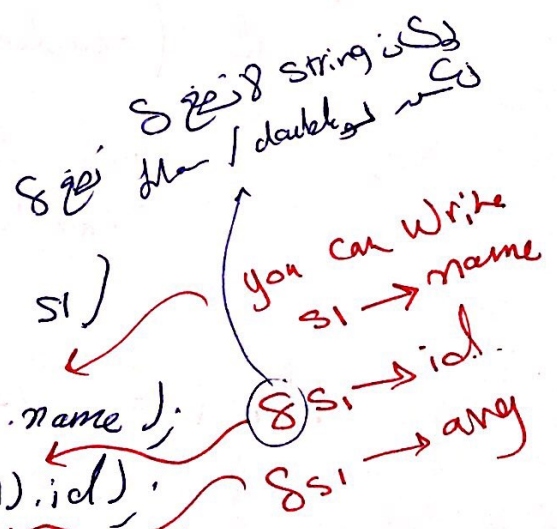
```
#include <stdio.h>
#define S 10
typedef struct
{
    char name [S];
    int id;
    double avg;
}
```



```
void fill-info (stud-t);
int main ()
```

```
{
    stud-t s;
    printf ("Enter name, id and avg for student\n");
    fill-info (&s);
    printf ("%s %d %.2f", s.name, s.id, s.avg);
    return 0;
}
```

```
void fill-info (student * s1)
{
    scanf ("%s", (*s1).name);
    scanf ("%d", &(*s1).id);
    scanf ("%lf", &(*s1).avg);
}
```



type def struct

```
{  
    char instructor [5];  
    int num;  
    stud. t students [20];  
}  
course - t  
course - t comp124;
```

```
int main ()  
{
```

```
    stud - t students [10], max - student, temp;
```

```
    int i;
```

```
    for (i=0; i<10; i++)
```

```
        fill info (& student [i]); → or
```

```
    max - student = student [0];
```

```
    for (i=0; i<10; i++)
```

```
        if ( student [i].avg > max - student . avg )
```

```
            max - student = student [i];
```

```
            printf ("max = %s", max - student . name);
```

```
    for (i=0; i<5-1; i++)
```

```
        for (j=0; j<5-1; j++)
```

```
            if ( student [j].id > student [j+1].id )
```

```
{  
    student [i].  
    scanf ("%s", )  
    scanf ("%d", & student  
    [i].id);  
    scan  
    }  
}
```

chur	20
son	30

max


```

temp = students [j];
students [j] = students [j+1];
students [j+1] = temp;
}
sum = 0, count = 0;
for (i=0; i<10; i++)
    if (strcmp (students [i].name, "ahmad") == 0)
        {
            sum += students [i].avg;
            count ++;
        }
avg_ahmad = (float) sum / count;

```

```
#include <stdio.h>
```

```
int equal_students (student_t, student_t);
```

```
int main ( )
```

```
{
    student_t s1 = { "ahmad", 20, 50.5 };
    student_t s2 = { "ahmad", 30, 50 74.3 };

```

```
if (equal_students (s1, s2) )
```

```
    printf (" same \n");
```

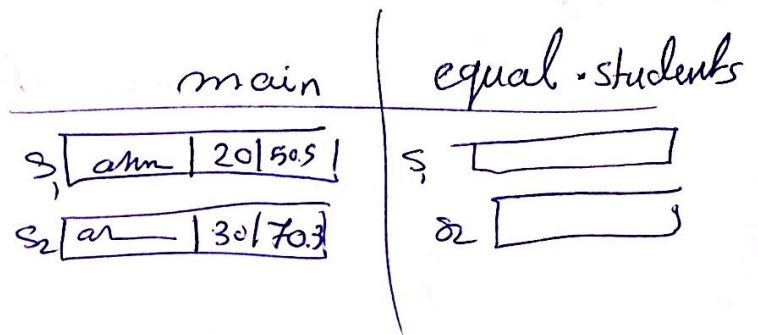
```
else
    printf (" diff \n");
return 0;

```

```

int # equal_students ( stud-t s1 , stud-t s2 )
{
    return ( strcmp ( s1.name , s2.name ) == 0
            && s1.id == s2.id
            && s1.avg == s2.avg )
}

```



Lecture 29

• Files :-

```
FILE *in;
in = fopen("data.txt", "r");
```

```
if (in == NULL)
{
    printf("cannot open file");
    exit(0);
}
```

تتوي كلسي عام

نقبرتق أي رقم exit()

```
FILE *in;
```

```
char filename[10];
printf("Enter file name\n");
```

```
scanf("%s", filename);
in = fopen(filename, "r");
```

```
if (in == NULL)
{
    printf("cannot open file %s", filename);
    exit(2);
}
```

هذا الرقم يعني انه خطأ
Error

Enter file name
- data
- Cannot open
- Enter file

data.txt لا يوجد

```
FILE *in;
```

```
char filename[10];
printf("Enter file name\n");
```

```
for (scanf("%s", filename); in = fopen(filename, "r") == NULL; scanf("%s", filename))
```

لا يوجد Loop ويطبق

```

}
printf ("cannot open file %s", filename);
printf ("Re enter file name\n");
}

```

output (again)

```

Enter
data
Connect
Re enter file
data.txt
name
Re
data.txt

```

disadvantages
 1- Not readable
 2- different between systems

Binary files :-

- disadvantages of text files:-
- 1- processing time
 - 2- precision problem
 - 3- space

```

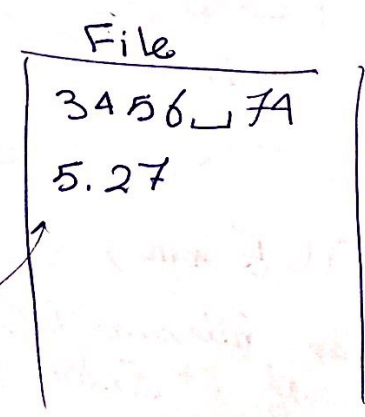
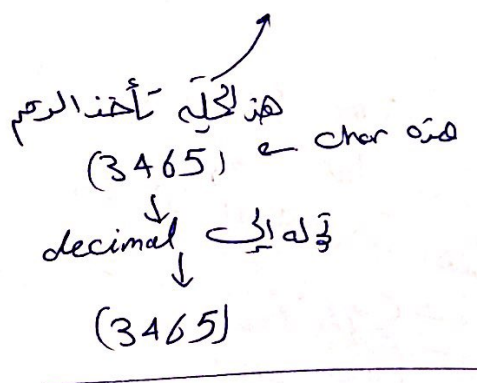
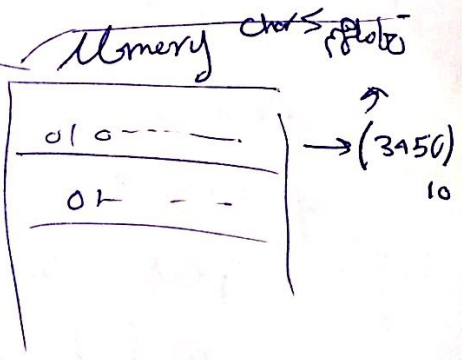
FILE *int;
int x=3456, y=74;

```

```

fprintf (out, "%d\n", x, y);
fscanf (in, "%d%d", &x, &y);

```



```

fprintf (out, "%.2f", x);

```


How to open Binary files?

```
FILE *in;
in = fopen ("data.bin", "rb");
fread (---, ---, ---, ---);
fwrite (---, ---, ---, ---);
fclose (in);
```

سكينة وفعواته مع

read binary
"rb"
write binary
"wb"

لا تعرف بيقيني في القراءة وليست
← القراءة نستعمل fread
← وليست مع fwrite

كيف اكتب اقران في file

```
int x=7, y; float a=3.5, b;
FILE *out;
out = fopen ("data.bin", "wb");
fwrite (&x, sizeof (int), 1, out);
```

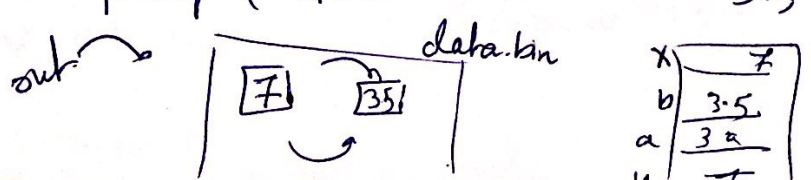
عنوانه x
اقصع بم
int في الذاكرة
memo على الشاشة

```
fwrite (&a, sizeof (float), 1, out);
fclose (out);
out = fopen ("data.bin", "rb");
```

8-bit → 2 byte
4 byte → float

```
fread (&y, sizeof (int), 1, out);
fread (&b, sizeof (float), 1, out);
printf ("x=%d . b=%f", x, b);
```

int n = sizeof (int); → 2
sizeof (float); → 4
sizeof (x) → 2



x=7, b=3.5

```
typedef struct
{
```

```
    char name [10];    → 4 byte
```

```
    int id;           → 2 byte
```

```
} stud_t;
```

```
stud_t s1 = {"ahmad", 10};
```

```
{ printf ("out", "%s %d", s1.name, s1.id);
```

```
{ scanf (" %s %d", myname, &myid);
```

```
{ write (&s, sizeof (stud_t), 1, out);
```

```
{ read (&s2, sizeof (stud_t), 1, out);
```

البيانات
من B إلى A

ما تعرف Array في الذاكرة مع سبوع

write س
read س

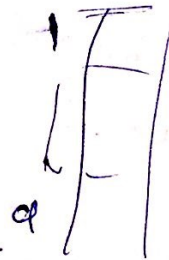
```
int B[10];
```

```
int A[5] = { 1, 2, 3, 4, 5 };
```

```
{ write (A, sizeof (int), 5, out);
```

```
{ read (B, sizeof (int), 5, out);
```

```
{ read (&B[5], sizeof (int), 5, out);
```



```
int n;
```

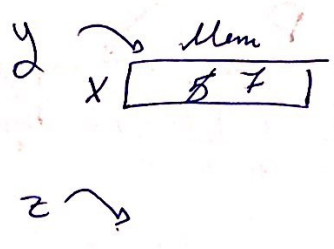
```
n = fread (A, sizeof (int), 1000, out);
```

File

Lecture 30

Dynamic allocation :-

```
int x=5;
int *y=&x;
int *z;
*y=7; ✓
*z=3; ✗
int A[1000];
int *A ; int A[ ];
```



• malloc (memory allocate)
 #include <stdlib.h>
 تنفع لأي نوع

```
int *x;
float *x;
x = (int *) malloc (sizeof(int));
```

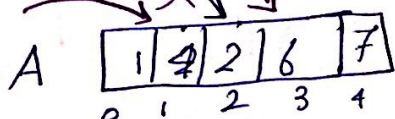
int ← تنوع في السعة الحجم من malloc وحواله
 void pointer
 x = (int *) malloc (sizeof(int) * 10)
 ← كجزء 10 int

calloc تستخدم لجزء مسطر

```
x = (int *) calloc (10, sizeof(int));
```

int A[5] = {1, 4, 2, 6, 7};

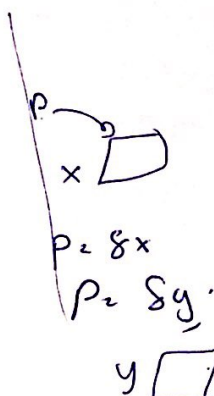
mit *p; P- P P++



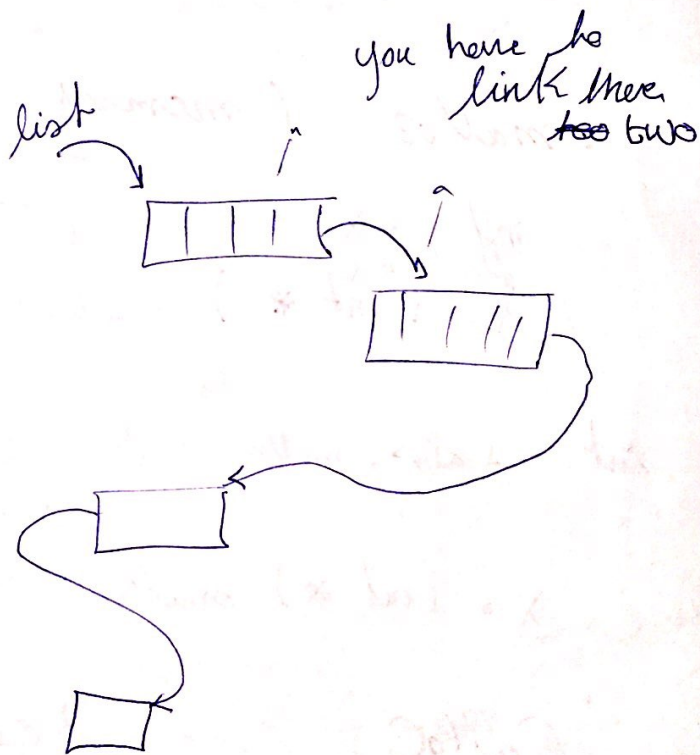
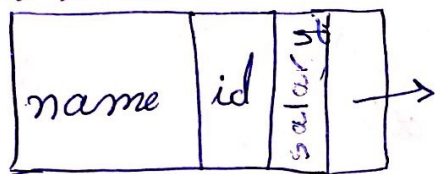
P = A;

P = &A[0];

P = &A[2];



• Structure \rightarrow linked list



```

type-def struct stud_a
{
    char name [10];
    int id;
    struct stud_a *next;
} stud_t;
    
```



```
int main ( )
```

```
{ char name[10]; int id;  
  stud-t *list;
```

```
  printf ("Enter name and id\n");
```

```
  scanf ("%s%d", name, &id);
```

```
  list = (stud-t * ) malloc (sizeof (stud-t));
```

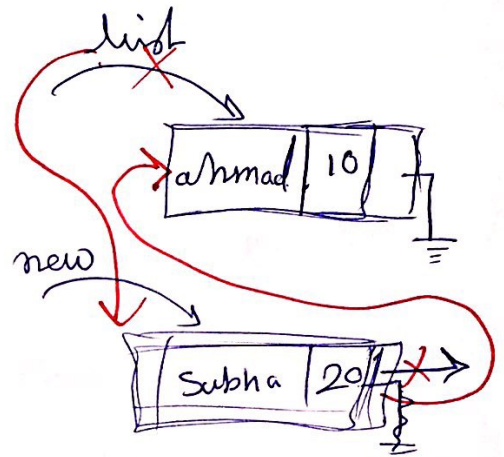
```
  strcpy (list->name, name);
```

```
  list->id = id;
```

```
  list->next = NULL;
```

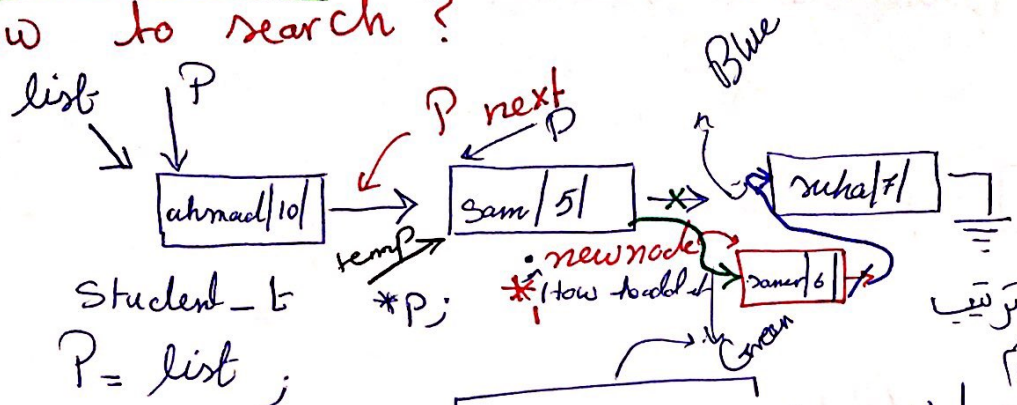
```
  You define new like list  
  new->next = list . linked
```

```
  list = new;
```



lecture 31

How to search?



```
Student_t
P = list;
```

لازم يكون في بهاد الترتيب
مصنوع ان نكتبه

```
while ( (P != NULL) && (P->id != x) )
```

we have
to add
this

```
P = P->next;
if (P != NULL)
    printf ("name = %s", P->name);
else
    printf ("%d No such id, x);
```

How to add?

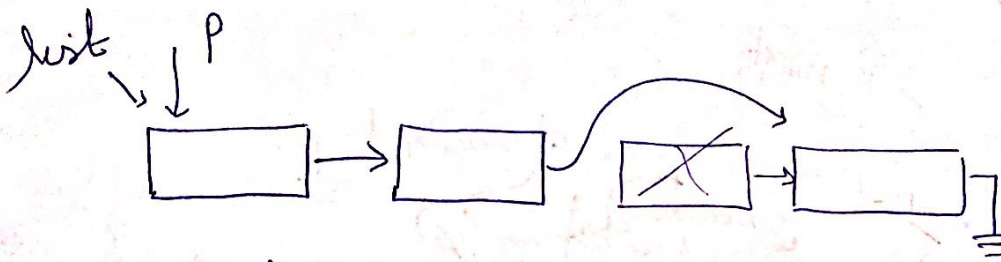
```
* :- newnode = (Student_t *) malloc (sizeof (Student_t));
newnode->next = P->next; This step (Blue)
P->next = newnode; This step (Green)
```

How to delete?

```
temp = P->next;
P->next = temp->next; (or you can say :- P->next = P->next->next)
free (temp);
```

function
العمل

How to print the list?



p = list;

while (p != NULL)

```

{
    printf ("%s\n", p->name);
    printf ("%d\n", p->id);
    p = p->next;
}
    
```

```

char temp [20];
char name [10][20];
char *name [10];
for (i=0; i<10; i++)
    
```

{ printf ("Enter name\n");

scanf ("%s", temp);

names [i] = (char *) malloc

(sizeof (char) * (strlen(temp)+1));

strcpy (names [i], temp);

names

0	a	l	i	o	
1					
2					
3					
4					

use

اسم

