Algorathms:-

Example :-

write an algorithem(pseudo coude) to find the sum of any two given numbers.

طرق الحل :-

- sequence . خطوات
- repetition (selection) تتفرع بعدة خيارات
- condition (loops) تكرار .
- if , such . خيارات الشرط .

note:-

لغة البرمجة هي اداة لتتيح الكمبيوتر طريقة الحل و اداء العملية بعد شرحها .

input
- Ask user to enter first number.
- read number and save as num 1
- Ask user to enter second number.
- read number and save as num 2.

processing [Add num 1 to num 2 and save result as Sum.

output [print sum to screen.

- يفضل أن تكون الجمل قصيرة .
- أغلب البرمجة تكون لل processing .
- اطلاق تسمية على المدخلات لتسهيل العملية فيما بعد

باقي القسمة (mood)

⇒ processing
- Arithmtic (+, −, *, /, %)
- logical op (>, <, =)

note :-

← نتيجة عدد صحيح على آخر بالبرمجة بعطيني رقم صحيح .

Ex:-

(1) 5%2 = 1 — 2)5 = 2, 4, 1

5/2 = 2

(2) 3/4 = 0 — 4)3 = 0, 3

3%4 = 3

732.

10)732 = 73, (2)

10)73 = (7), (3)

10)7 = 0, 7

→ كل برنامج يجب وجود وتعريف ال main "مثل الحرير".

أول سطر بكل برامج لغة C . # include <stdio.h> عمليات بسيطة .

(heder file) <math.h> ممكن نضع أكثر من واحدة

→ data taype :-

int , char , float , double . تم اختيار النوع المناسب للحفاظ على المساحة .

→ شروط تسمية المتغير المدخل :- لا يجوز أن تبدأ برقم .

- لا يجوز أن تحتوي على spaces ويمكن نوضع _

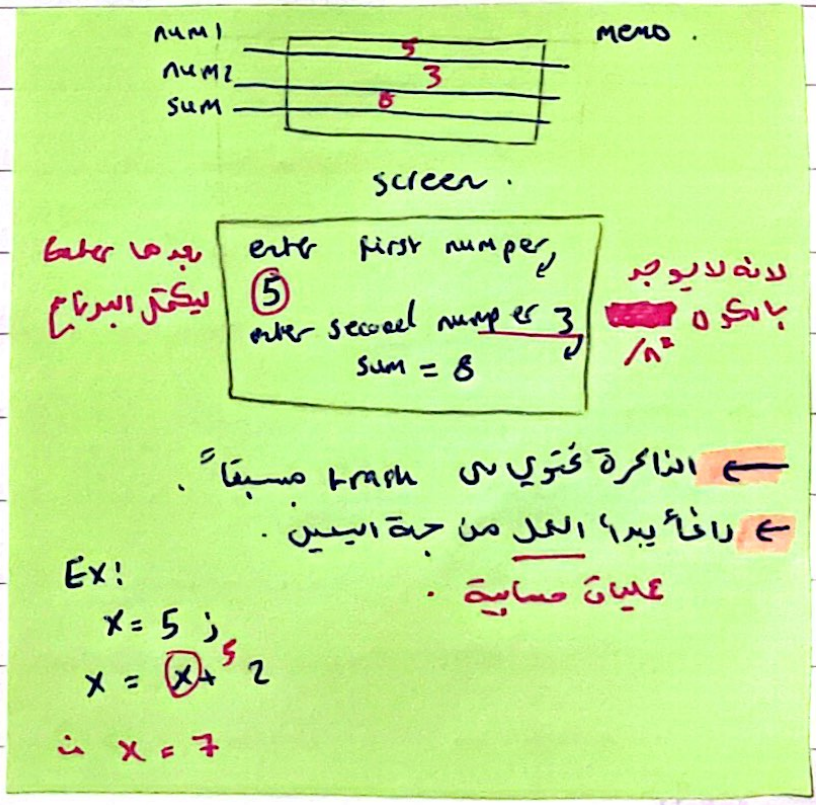
→ بعد ال data taype يجب عمل space . " int sum ; "

→ أي شيء يطلع منها ادخل إلا الذي يبدأ " / or " %"

→ printf like Ask user to .

From the Example:-

```
# include <stdio.h>      standart input and output.
int main ( )             حدد صحيح . نوع الخرج
{
   int num1, num2 ;      تعريف المتغيرات variables
   int sum ;             function
   printf ("enter first number \n") ;     انزل سطر Enter
   scanf ("%d", &num1) ;                  وظائف تساعدني تنفيذ العملية .
   printf ("enter second number") ;       بعد نهاية كل جملة .
   scanf ("%d", &num2) ;                  عنوان
   printf ("sum = %d", sum) ;             النوع . int integr .
   return 0 ;            success  يدل على نجاح العملية
}
```



note:-

compile (build)

run ▷ (execute).

خطأ أثناء التنفيذ .

signal errors . خطأ تركيبي

wrongs . خطأ بالتفكير بالمثال

Exampel :-

write an algoritham to find the area of a rectaagle given the lengthe and width.

Ask the user to enter the lenght.

Read the lenght and save as num1.

Ask the user to enter the width.

Read the width and save as num2.

the area equal num1 cross num2 and save the riselt as area.

Print the area to screen.

to find max between 2 value :-

n_1 n_2.

if $(n_1 > n_2)$

print n_1 to the screen

else

print n_2 to the screen.

End if

← إذا كان هناك أكثر من شرط نستخدم Else if

← نستطيع أن نأخذ من المستخدم الرقم الأول على أنه الأكبر والثاني هو الأصغر "أكثر من قيمتين".

(n_1, n_2, n_3).

- max = n_1

if $n_2 >$ max.

max = n_2

endif

if $n_3 >$ max

max = n_3

endif

=> n_1 n_2 n_3 لايجاد القيمة المتوسطة "ليست الاكبر ولا الاصغر".

if $n_2 > n_3$ and $n_2 > n_1$

print n_2 max.

if $n_3 < n_2$ and $n_3 < n_1$

print n_3 min.

n mid = $(n_1 + n_2 + n_3) - (max + min)$.

Constants + floats.

```
#include <stdio.h>
#define PI 3.14 space capital letters تكون constant لانها
{
    int rad;
    float area, circum;
    printf("enter radius \n");
    scanf("%d", &rad);
    area = PI * r*r;
    circum = 2* PI* rad;
    printf("area = %f circum = %f", area, circum);
    return 0;
}
```

Data tayfe:-

- int x = 3;
  - printf ("x = %d", x);
  - scanf ("%d", &x);
- float x = 3.2;
  - printf ("x = %f", x);
  - scanf ("%f", &x);
- double x = 3.2;
  - printf ("x = %f", x);
  - scanf ("%Lf", &x);
- char c = 'f';
  - printf ("result = %c", gender);
  - scanf ("␣%c", &gender);

Example:-

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

space.

```
char x;
x = 'b';
```

```
int x, y = 3;
نفس الحالة x = 6;
x = y;
```

```
char x, y = 'c';
x = 'y';
x = y;
    يطبع y بدل x
يطبع y بدل x
```

Airthimetric Expresciun + formula.

* , / , % . العمليات الثنوية "الاولوية".

+ , − العمليات الاضعف .

Example:-

x= 5 + 2*3 /7 - 2;

= 5 + 6/7 - 2

= 5 + 0 - 2

= 3

x = (5+2)* 3 / 7-2 ;

= 7* 3 /7-2

= 21 /7 - 2

= 1

→ $x = \frac{y+2}{2(x-3*4)}$ = (y+2) /(2*(x-3*4)) .

Type casting:-

int x;

x= 5/2; (2).

x= 5.0/2; (2)

float x;

x= 5/2; (2)

x= 5.0/2;

x= 5/2.0; → (2.5)

x = 5.0/2.0;

• عند ال % يكون فقط integr.

int x=5 , y=3;

float z;

z = 4/6;

z = 4.6/6;

=> z= (float)x /y %o

= x /(float)y ;

= (float)x/(float)y;

• z= x %y; (x,y int)

z = (int)x %y;

when x float "type casting".

Error Type:

1) syntax (compile, build) .
2) run-time (vnte) (error) .
3) logical error.

Ex:-

1) تكليات مثل (; ، ' ،) ناقصة .
2) z = x-y;

b = 3/(z); when z = 0.

• (5 int) , (5.0 float) .

z = 5/2.0
z = 5.0/2 } = (2.5) if z float

z = 5/2 — (2) if z int .

• 4/3 *(—————)

دائما من أخرى float

يكون الناتج float .

out put formatting:

Example:-

```
int x=12467 , y=3 ;
printf ("%d ␣ %d", x,y);   (space)
```

it's be like : 12467 _ 3

7 _ 304

1234567 _ 21

ال width تحدد ...

→ printf ("%7d %4d", x,y);

_ _12467 _ _ _ _ _ _ _3

20 514

704 21

→ printf ("%-2d ␣ %5d", x,y);

12467 ␣ _ _ _ _3

```c
float x= 2463.25497 , y= -3.619;
printf("%-8.1f ␣ %07.4f", x,y);
```

2463.3 _ _ ␣ -3.6190

"%-8.1f"
- البدء من اليسار -
و اليمين +
التقريب خانة عشرية واحدة.

Text Files (intro):

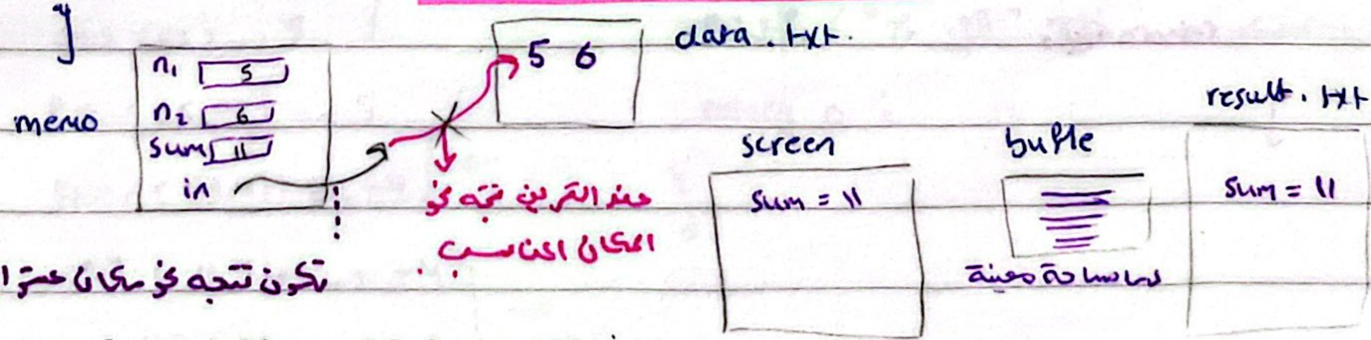
```c
#include <stdio.h>
int main( )
{
    int n1, n2, sum;
    FILE *in, *out;
    in = fopen("data.txt", "r");
    out = fopen("result.txt", "w");
    fscanf(in, "%d %d", &n1, &n2);
    sum = n1 + n2;
    printf("sum =%d", sum);
    fprintf(out, "sum= %d", sum);
    fclose(in);
    return 0;
}
```

the mode write or read. (w) (r)

مكان تخزين الملف. (in)

مكان إظهار النتيجة. (out)

يتم تخزين البيانات مؤقتاً في ال bufle وبعد ذلك يتم نقلها إلى مكان التخزين "File".



تكون تتجه نحو مكان عشوائي.

عند وضع fclose(in); تنكسر الرابطة، تكون نحو الاتجاه العشوائي.

Function:-

code reuse :- اداة استخدام اكثر من مرة

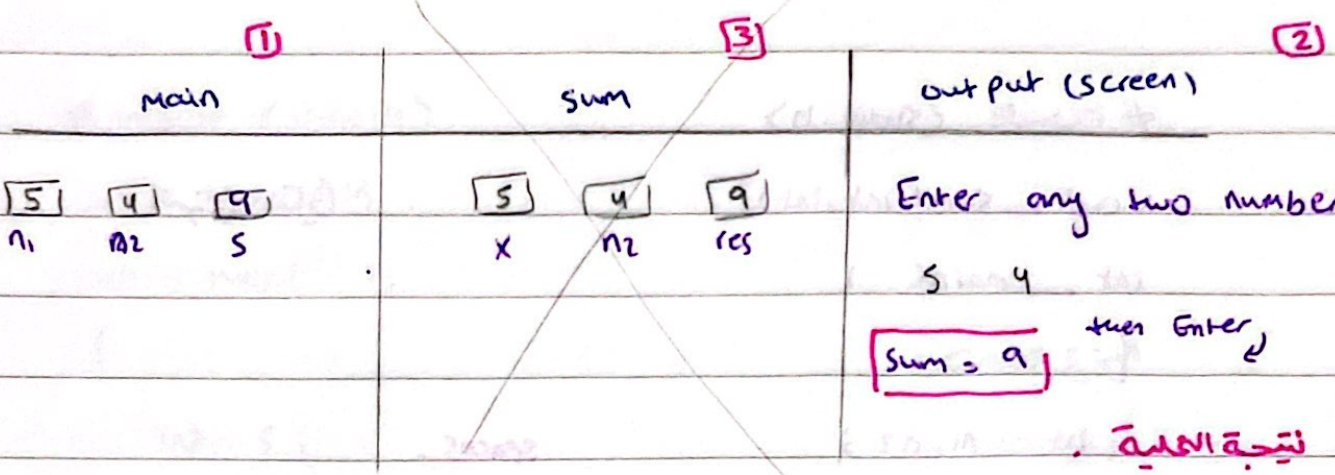
modularity . لو كتبته مرة واحدة فقط وسيتخدم بعد ذلك اكثر من مرة

```
# include <stdio.h>
int sum(int,int); Function prototype.
int main()
{
    int n1, n2, S;
    printf("Enter any two number\n");
    scanf("%d%d", &n1, &n2);
    S = sum(n1, n2); ⇒ Function call.
    printf("sum = %d", S);
    return 0;
}
int sum(int x, int n2)
{
    int res;
    res = x + n2;
    ⇒ return res;
}
```

Function definition.

النوع الاول :-

يمكن تكون الاعداد متساوية، يمكن لا.



نتيجة العملية .

الـ return 0s تعني حذف العملية الأخيرة وإعادة النتيجة إلى الشاشة .

=> abs : لا تأخذ الا int إذا أردنا إدخال float نستخدم fabs

System defied Function:-

#include <math>	
sqrt(_)	floor
pow(_ , _)	ceil
abs	cos
fabs	sin

Ex:-

x = sqrt (16); → 4

x = pow (4, 2); → y^2

abs (-5); → 5] قيمة مطلقة

abs (5); → 5]

floor (5.7); → 5]

floor (-3.4); → -4] ←

ceil (-4.7); → -4]

ceil (2.6); → 3] →

like:

```
# 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 ); pow (y,y)
  printf ("%.2f", z);
```

التمرين لاثنين .

```
  return 0 ;
}
```

```
#include <stdio.h>
void sum(int,int);
int main()
{
  int n1, n2;
  printf("Enter n1 n2 \n");
  scanf("%d%d", &n1, &n2);
  sum(n1, n2);
  return 0;
}
  void sum(int x, int n2);
{
  int res;
  res = x + n2;
  printf("sum = %d", res);
}
```

البرنامج التالي :

spaces.

=> Function call.

main	sum	out put
4 5 n1 n2	4 5 9 x n2 res	Enter n1 n2 4 5 ↵ enter sum = 9

```
#include <stdio.h>
int sum( );
int main( );
{
    int s;
    s = sum( );
    printf("sum = %d", s);
    return 0;
}
int sum( )
{
    int n1, n2, res;
    printf("Enter n1 _ n2 \n");
    scanf("%d%d", &n1, &n2);
    res = n1 + n2;
    return res;
}
```

السؤال الثالث :-

مايخرج برنامج

main	sum			output
9 s	4 n1	5 n2	9 res	Enter n1 _ n2 4 _ 5 Sum = 9 ↵ Enter

```
#include <stdio.h>
Void sum ( );
int main ( );
{
   sum( );
   return 0;
}
Void sum ( )
{
  int n1, n2, res;
  printf ("Enter n1_n2 \n");
  scanf ("%d %d", &n1, &n2);
  res = n1+n2;
  printf (" sum = %d ", res);
}
```

السؤال الرابع :-

لديباخذ دلايبرجع .

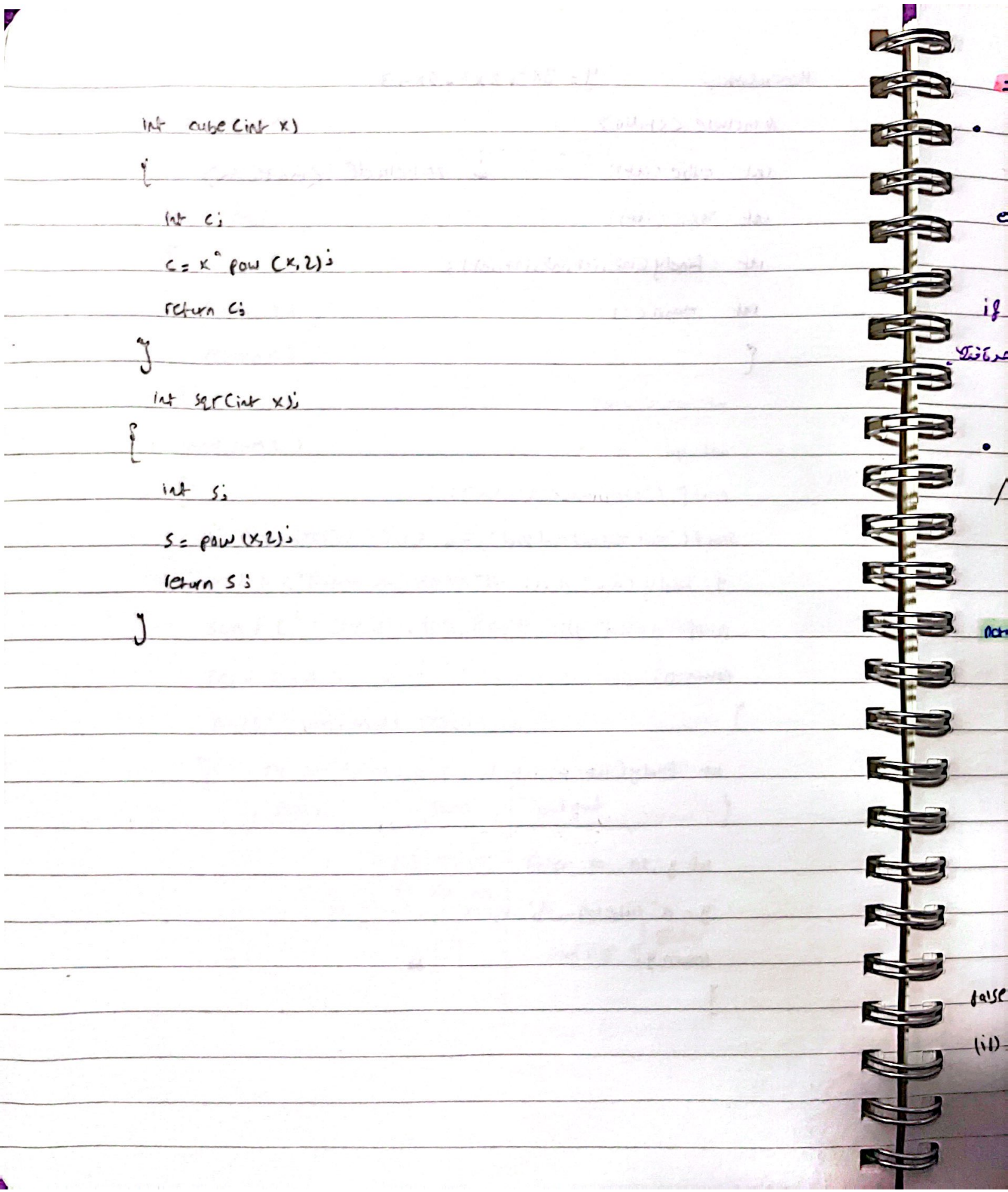
main	sum	output
	4 5 9 n1 n2 res	Enter n1_n2 4_5 sum = 9 ↵ Enter

Homework:- $y = 3x^3 + 2x^2 - 5x + 3$

```
#include <stdio>
int cube(int);
int sqr(int);
int findy(int, int, int, int, int);
int main()
{
  int a,b,c,d,x;
  int y;
  printf("enter a,b,c,d, x\n");
  scanf("%d %d %d %d %d", &a, &b, &c, &d, &x);
  y = findy(a,b,c,d,x);
  printf("y = %d", y);
  return 0;
}
  int findy(int a, int b, int c, int d, int x)
  {
   int y;
   y = a*cube(x) + b*sqr(x) + c*x + d;
   return y;
  }
```

#include <math.h>

```
int cube(int x)
{
  int c;
  c = x * pow(x,2);
  return c;
}
  int sqr(int x);
{
  int s;
  s = pow(x,2);
  return s;
}
```



relational operation:-

1) < les than.

2) > greater than.

3) == equal.

4) <= les or equal.

5) >= greater or equal.

6) != not equal.

Selection:-

```
• if (grade >= 60)
      printf ("pass");
  else
      printf ("fail");
```

• مهم: دخل { } بعد جملة الشرط الخاصة بـ if وبعد else لأنها كثيرا ان كلمها في جملة احدهم

• comentice:-

```
/* ____          or     // ____
   ____
   ____                 // ____
          */
```

← تكون حل للمستخدم أو قارئ الكود ولا يقرؤها البرنامج أثناء التنفيذ.

note:-

• x = 5;
if (x = 5) ✓

• x = 5;
if (x = 4) ✓

• x = 3;
if (x)

• if (x %2)

• if (x).

في هذه الحالات يقوم البرنامج بتطبيق الكود ويعطي (true) لهذا يوجد مشكلة في الكود.

• x = 0;
if (x = 0)

• x = 0
if (x == 0) ⇒ True.

في هذه الحالة يتم التطبيق الكود ويعطي true حيث أن الـ (==) تعني مقارنة فيقارن ما بداخل الـ (if) مع ما هو معرف فوقه فتكون مقارنة ناجحة.

في هذه الحالة يتم التطبيق في اعطاء false ولا يطبق الكود لأنه يجب أن لا يكون ما في الـ (if) يساوي (0)

Ex:-

write a program to change a given marks (0-100) to a letter grate (A= 90-100 , B = 80- 89 , C= 70-79 , F= 0-69) .

```
int mark ;
printf(" Enter your mark (0-100) ");
scanf(" %d", &mark);
if (mark >=90)
    printf(" grate is A ");
else if (mark>=80)
    printf(" grate is B");
else if (mark>69)
    printf(" grate is C");
else
    {
    printf(" grate is F ");
    printf(" see u next semester");
    }
print f (" bye" );
```

logical operation:-

1) ! Not
2) && and
3) || or

صوبات تحن صبت الاقوى ،

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

EX:-

```
int age = 20;
char gender = 'f';
float average = 80.3%;
if ((age > 20) && !(gender == 'm') || (average >= 60))
```

(F && !F || T)

(F && T || T)

(F || T)

T

Short cricait :-

x=5 , y=7 ;

- if ((x>2)||(y==3))

T

- if((x>3) && (y==6))

T && F = F

- if((x<3) && (y==6))

F

- y=2 ;
- if((x>3) && (y=6)) T يطبع y=6
- if((x<3) && (y=6)) F y=2 =

- اذا استبنج البرنامج الناتج دون اكمال العملية يعمل short cricait ولا يكمل كتابة الجملة .

switch :-

```
if (Val==1)
    printf("one\n");
else if (Val == 2)
    printf("two\n");
else
    printf("unknown\n");
```

```
switch(Val)
{
case 1: printf("one\n");
break;
case 2: printf("two\n");
break;
default : printf("unknown\n");
}
```

note :-

- يجب وضع break بعد كل case باستثناء ال switch ليكون بمثابة الخيار الاخير ولكن لا يكمل البرنامج بقية الجملة، حيث لو انه اكمل لقام بطباعة كل الجمل .
- نكد جملة الـ switch أفضل من جمل if لعدة أسباب:

1) لا تكون اكثر من عمليتان المقارنة من نوع مساواة .

2) لا تقبل كل انواع data type في تقبل فقط من نوع
   - int
   - char

   ولا يجوز الا في الحالات
   - float
   - double

Ex:-

```c
char letter;
printf('Enter letter\n');
scanf(" _%c", & letter);      space.
if (letter == 'a' || 'o' ||'u' ||'e' || 'i')      it's wrong word ←
pBarf( "letter %c is a vowel \n"), letter);
else
    printf( "letter %c is not a vowel ", letter);
```

→ نتيجة داخل جملة شرط if بالمختصر تعطيني True أي تستطيع أن كل الأحرف vowel ((F|T)||T||T||T||T) .

it's should be

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

- switch (letter)

```c
{
case 'a': case 'o': case 'i': case 'e':
case 'u': printf ( "letter %c is a vowel", letter);
default: printf(" letter %c is not a vowel ", letter);
```

boolean function:-

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

```
return !(n%2);
```

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

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

this function gives just true or false

If it's true (1) , if it's false (0) .

⇒ 1) if(isEven(num)==1)

2) if(isEven(num)) الترجمة من أول خيار.

```
{

}
```

compaund + nested if :

- int x=3 (x=13), y=2 (y=12), z=5 ;

```
if (x<5)
{
printf ("one \n");
if (y<10)
printf ("two\n");
}
else
printf ("three \n");
printf ("four");
```

- one
- two
- four
- three
- four

- one
- two
- four
- three
- four

- int x=13, y=2, z=5;

```
if (x<5)
{
if (y<10)
printf ("two");
else
printf ("three");
printf ("four");
}
```

- four
- ____

لا يقوم بطباعة شيء.

```
int n1, n2 ;
int choice , result;
printf("Enter two number");
scanf("%d %d", &n1, &n2 );
Printf("1-add \n 2-subbract\n 3-multiply\n");
Scanf("%d", &choice );
switch (choise)
{
   case 1 :  result = n1 + n2 ; break;
   cas 2 :  result = n1 - n2 ; break;
   cas 3 :  result = n1 * n2 ;
}
printf("result = %d ", result);
```

```
Enter two number
5  6
select action :
1 - add .
2 - Subbract .
3 - multiply .
```

الأقوى

loops: . while , for , do/while .

```
(x) 5;     ] initial value .
while ((x)<=10) ] condition (final value)
{
 Printf("(x) = %d", x );
 x++ ;     ] change .
}
printf ("bye \n");
```

- x = 5
- x = 6
- x = 7
- x = 8
- x = 9
- x = 10

← قيمة x اصبحت 11

bye .

post increment : x++ ;

pre inc : ++x ;

post dec : x-- ;

pre dec : --x ;

- x=2;

  x++;

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

  x++ → 3

  ++x → 3

- x=2;

  y=x++; ++x;

  y=2 ← → y=3

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

  x++ → 3

  ++x → 3

- x=2; هنا بيتأكد x=3 وبالتالي لايدخل اللوب.

  while (++x < 3)

  {

  }

  يمان باللغة الاصلية (x=2)

  يدخل اللوب بـ (x=3)

- x=2;

  printf("%d", x++); 2 3

  printf("%d", x); 3 3

- x=x/y;

  x/=y;

تقرأ أول مرة فقط لشرط الحلقة.

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

```
for (x=1; x<5; x++)
  printf("hi \n");
```

تستخدم عندما لا نعلم البداية والنهاية (while)

عندما نعلم البداية والنهاية (for)

Ex:

n! , 5! -> 1*2*3*4*5

5*4*3*2*1

لو كانت عملية جمع يكون 0.

لأنه عنصر محايد في الضرب.

```
• int result = 1
  n
  j=1;
  while (j<=n)
  {
   result = result * j;  or  result *= j;
   j++;
  }
```

```
• int result = 1;
  j=n;
  while (j>0)
  {
   result = result * j;
   j--;
  }
```

• res = 1*1 = 1
= 1*2 = 2
= 2*3 = 6
= 6*4 = 24
= 24*5 = 120

```
• for (j=1; j<n; j++)
     result *= j;
```

(i+=2) تم إضافة 2

EX:

x^y , $2^4 = 2*2*2*2$

```
int result = 1;
i = 1;
while (i<=y)
{
    result *= x;
    i++;
}
```

res = 1*2 = 2 (i=1)
= 2*2 = 4 (i=2)
= 4*2 = 8 (i=3)
= 8*2 = 16 (i=4)

EX: divisor . n=20 → 1/2/5/4/10/20 .

(int sum=0;) divisor ل sum ازاد!

```
for( i=1; i<n; i++)
    if((n%i)==0)
    ~~printf("%d\n",i);~~
    sum = sum + i;
    printf("sum = %d", sum);
```

1
2
3 → (n%3 != 0)
4
5
10
20

Ex:

```
int mark, sum=0, i;
float avg;
i = 1 ;
while (i<=5)
{
  Printf ("Enter mark\n");
  Scanf ("%d", &mark);
  sum += mark;
  i++ ;
}
  avg = sum / 5.0 ;
```

- برنامج يقرأ درجات (5) طلاب ويحسب المعدل avg
فما إذا كان عدد أكبر أو أصغر لا يعطي avg

=) • sentinel — يقرأ مدخل إذا أدخله المستخدم يعطي الـ avg ويخرج مباشرة (0)

• priming the pump.

- while (mark != -1)

```
{
    printf ("Enter mark\n");
    scanf ("%d", &mark );
    sum = sum + mark;
    count ++ ;
```

- ```
    sum = 0;
    cont = 0;
    printf ("Enter mark or -1 to stop \n");
    scanf ("%d", &mark);
    while (mark != -1)
    {
    sum = mark;
    count ++ ;
    printf ("Enter mark or -1 to stop");
    scanf ("%d", mark);
    }
    if (count > 0)
    {
      avg = (float) sum / count;
      printf ("avg: %.2f", avg );
    }
else   printf ("no mark enter\n");
```

- Enter ...
  60
  sum = 0 + 60 = 60
  count = 1
  Enter ...
  70
  Sum = 60 + 70 = 130
  count = 2
  Enter ...
  -1
  avg = 75.00
```

- **do / while :**

```
int num = 32462, count = 0;
while (num>0)
{
  count ++;
  num/=10;
}
printf(" count = %d ", count);
```

. count 1 2 3 4 [5]

```
int num = 32462, count = 0;
do
{
  count ++;
  num /= 10;        = num=num/10;
}
while (num>0);
printf(" count = %d", count);
```

note:-

do / while : تنفذ مرة واحدة على الأقل دائماً

وتكون قبل تنفيذ جملة while يعني عند الدخول وتم لا يحقق شرط الـ while ولكنه يعتبر (count) متغير .

- **break and continue :**

```
x=2;
while (x<6)
  {
  printf ("hi\n");
  if (x==4)
  break;     continue;
  x++;
  }
printf (" done \n");
```

إذا الشرط تحقق اطلع [break]

break :	continue :
hi	hi
bye	by
hi	hi
bye	by
hi	hi
done.	hi
	hi
	⋮

. continue بعد الـ (x++) لن تأتي ⇐ infinit loop.

```
x=3;
while (x++ <=10)
{
  printf("%d", ++x);
  if (x==8)
  break;   continue;
  printf("%d\n", x++);
  }
  printf("bye\n");
```

break;	continue;
5	5
5	5
8	8
bye.	10
	10
	bye

. here x=12 ← لو دخل ال loop يكون

Nested Loops:

```
int num;
int i;
printf("Enter any number\n");
scanf("%d", &num);
for (i=2; i<num; i++)
    if (num%i==0)
        printf("it's not prime");
    else
        printf("it's [redacted] prime");
```

. no ~~a~~ prime يطبع أرقام مرة

```
int num, i;
int prime =1;
printf("Enter any number");
scanf("%d", &num);
for(i=2; i<num; i++)
if (num%i==0)
{ prime=0;
  break; }
if (prime)
  printf("%d is prime", num);
else
  printf("%d is not prime", num
```

نفس البرنامج السابق لكن باستخدام ال function .

```
{                           int isprime (int) ;
int num, i ; not important in the main
printf("Enter any number\n");
scanf ("%d", &num);
if (isprime (num))
    printf("%d is prime", num);
else
    printf("%d is not prime", num);
return 0;
}

int isprime (int n)
{
int i ;
for (i=2; i<n; i++)
    if (n%i ==0)
        return 0;
return 1;
```

تمرين ال function

```
{ int num ;
for (num=1 ; num<=100 ; num++)
if (isprime (num))
printf("%d\n", num);
:
:
```

لا يجد ال prime قبل 100 رقم .

Drawing a one dimensional n stars (*)

```
for (i=1; i<=n; i++)
```

printf("*"); or printf("%c",'*'); تختلف بعض البرامج بين بعض

Drawing a two dimensional nXn stars (*)

- for (i=1; i<=n; i++) تختص بالصفوف
- {
- for (j=1; j<=n; j++) تختص بالاعمدة أو داخل الصفوف.
  printf("*"); نجمة تتم طباعة i و j
  printf("\n");
- }

Triangles:

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

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

* * * *
* * *
* *
*

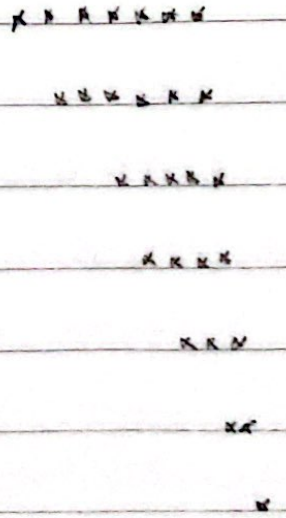
* *
* * *
* * * *

n=4 لو فرض

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

Homework:

```
int i, j, k, n=7;
for(i=1; i<=n; i++)
{
    for(k=1; k<i; k++)
        printf(" ");
    for(j=i; j<=n; j++)
        printf("* ");
    printf("\n");
}
```



```
• int i, j, k;
  int n = 4;
  for (i=1; i<=n; i++)
  {
      for (k=i; k<n; k++)
          printf("  ");
      for (j=1; j<=i; j++)
          printf("%d", i);   ("%d", J);
      printf("\n");
  }
```

```
   1
  22
 333
4444
```

```
   1
  12
 123
1234
```

Using loops to Read from files (EOF char):

scanf + fscanf : return Value.

- int status; اذا كان %c %d لا تقرأ %c بيكون 1 , اذا كان %c لا يحمل الزائد يكون 0

2 تكون status = scanf("%d %d", &x, &y);

printf("%d", status);

- Write a program to read an unspecified number of grades from a file and find and print the average grade

```
{
    int status;          // scanf / fscanf الدالي بيرجع رقم
    int grade, sum=0, count=0;
    float avg;
    FILE *in;
    in = fopen("grades.txt", "r");          // 70
    status = fscanf(in, "%d", &grade);      // 60
    while (status != EOF)                   // 60
    {                                       // => Avg grade = 70.00
      sum += grade;
      count++;
      status = fscanf(in, "%d", &grade);
    }
    avg = float(sum) / count;
    printf("Avg grade = %.2f", avg);
  return 0;
}
```

- for i=2, j=7; (i<5 && j>3); i++, j--) .

  printf("hello\n");

```
i=2;
j=7;
while(i<5 && j>3)
    printf("hello\n");
i++;
j--;
```

- `j=5;` تقرأ مرة واحدة فقط بداية الـ loop .

```
while (j>9)
{
  i=2;
while(i<7)
{ =
  ==
  ===
  i++;
}
j++;
}.
```

`i=2;` تقرأ دوماً بداية اللوب .

وتعاد قراءتها عند الخروج من اللوب والدخول إليها مرة أخرى كجزء شرط (i)

pointer :- dynamic addreses. عنوان متغير .

int x=5;

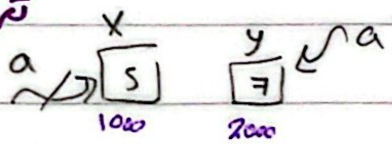
x [5]   قيمة المتغير (x)

مكان بالذاكرة   • printf("%d", &x);   • نقل بس طباعة عنوان x بالذاكرة

تعريف الـ pointer   or we put (%p).   وليس قيمته .

=> int *a = &x ; int y=7, int x=5;   - static . ثابتة

FILE *in; File . بس تعريف pointer   - dynamic متغير

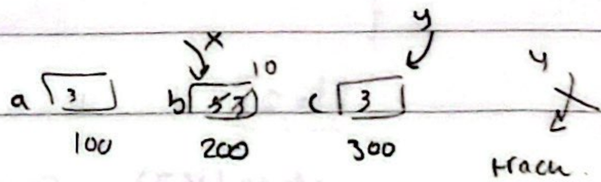


⋮

a = &y;

=> ing a=3, b=5, c;

مختلفة . ing *x = &b, *y; . تعريف



printf("%d", &b);   200

printf("%d", x);   200

printf("%d", *x); ما يوجد داخل (x)   5

c=a;   inderction * pointer القيمة الذي يؤشر عليها ال

y=&c;   note:

printf("%d", *y);   3   if we put *x=10;

*x = *y;   سيقوم بحذف (5) ويضع بدلا منه

*x = *x + 7;   10   10 لأن ال pointer متغير وليس

printf("%d", b);   10   ثابت مثل ال (&) .

=> int x=7, z=2, y;

int *b=&y, *a;

Printf ("%d", &y); 3000

a = &x;

Printf ("%d", (*a)++); 7

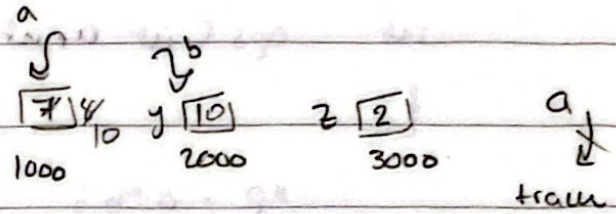
y= *a + 2;

Printf ("%d", (y+x)); 18

printf("%d", *b); 10

*a = *a +2;

printf ("%d", x); 10



=> #include <stdio.h>

void ~~int~~ ops (int, int, int *, int *); int *;

int main()

{

int x=5, y=3;

int sum, prod, diff;

sum = ops (x, y, &prod, &diff); &sum

print f ("sum = %d prod = %d diff = %d");

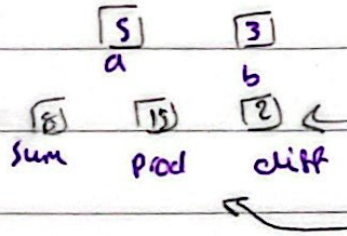
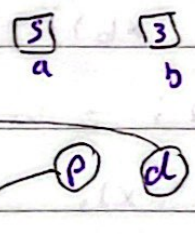
return 0;

}

output parameters.

.void ;(3>*

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

main	ops	output.
		sum = 8 prod = 15 diff = 2

لا pointer تشير إلى مكان بالذاكرة، بالتالي تستطيع أن تشير على مكان

بـ main مع إنها داخل الـ function .

```
=> void ops (int , int , int * , int *);

int main( )
{
    int sum , prod, x=5, y=7;
    ops (x, y, &sum, &prod);
    printf(" Sum = %d  prod = %d ", sum, prod);
    return 0;
}

void ops int a, int b, int *s, int *p)
{                                   int tmp;
    *s = a+b;                       tmp = *s =
    *p = a *b;                      *s = tmp+a;
}                                   *p = a *tmp;

=> void ops (int *, int *);

int main( )
{ int x=5, y=7;
    int prod;
    ops (&x, &y);
    printf ("sum = %d prod = %d "), x, y);
    return 0;
}
```

```
void Ops int *s , int *p)
{
    int tmp , tmp2 ;
    tmp = *s ;
    tmp 2 = *p;
    *s = tmp + tmp 2 ;
    *p = tmp * tmp 2 ;
}
```

swap function:

```
x = y;
y = x ;
```
] it's wrong.

you must use a temp value as follows:

```
tmp = x ;
x = y;
y = tmp ;
```

```
void swap(int ,int );
int main ( )
{
int   x = 5, y = 7 ;
swap ( x, y) ;
printf(" X = %d . y = %d ", x, y) ;
return 0;
}
void swap(int a, int b) ;
{ int temp ;
  temp = a ;   a = b ;
  b = tmp ;
}
```

. pointer لا يعمل لأنها

(printf) تتعامل مع الذاكرة عكس

. وبكده هذا حل غير فعال

=> void swap (int*, int*)

```
int main()
{
 int x=5, y=7;
 swap (&x, &y);
 printf("x=%d y=%d", x, y);
 return 0;
}
void swap(int *a, int *b)
{
 int tmp;
 printf("a=%d b=%d\n", *a, *b);
 tmp = *a;
 *a = *b;
 *b = tmp;
 printf("a=%d b=%d\n", *a, *b);
}
```

Home work :-

do the program with this function :-

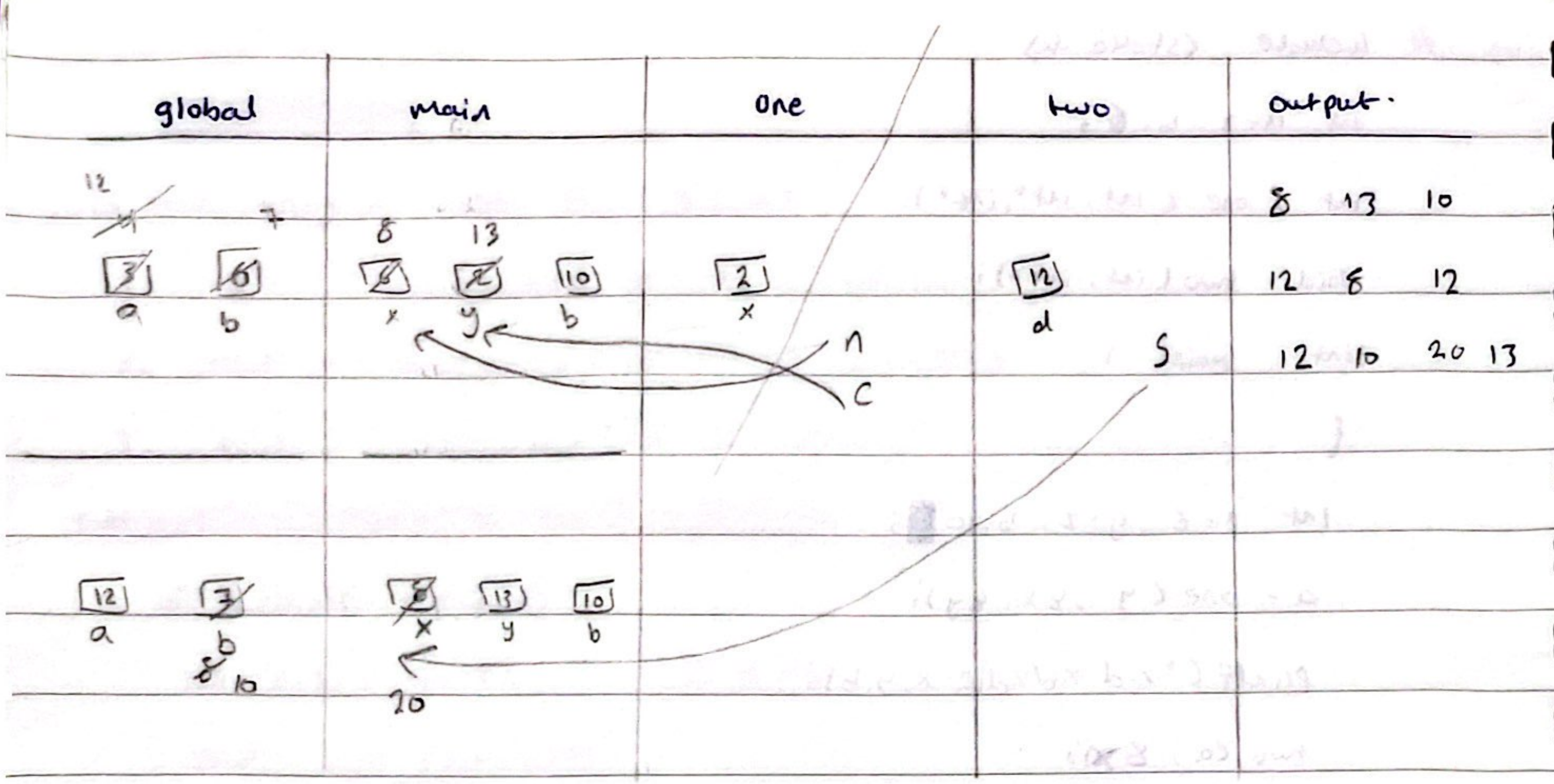
1. int main - max (int , int, int , int*);
2. Void main - max (int , int , int , int* , int*);
3. Void main - max (int, int* , int*);

Global Variables Vs local Variables :-

```
#include <stio.h>
int x=10 , y= 7 ;          // global
Void doit (int, int);
int main( )
{   int a= 2, b=3 ;        // local .
    doit (a,b);
    x++ ;                  // x=11
    printf ("%d %d %d %d", a,b, x,y);
    return 0; }            2 3 11 8
Void doit (int c, int d)
{   int x ;                // local.
    x= c+d;                // x = 2+3 = 5
    y++ ;                  8
    printf ("%d", x); }
                           5
```

⇒ # include <stdio.h>

```
int a=3, b=6;
int one (int, int*, int*);
void two (int, int*);
int main ()
{
int x=6, y=2, b=10;
a = one (y, &x, &y);
printf ("%d %d %d %d", x, y, b);
two (a, &x);
printf ("%d %d %d %d", a, b, x, y);
return 0;
}
int one (int x, int *n, int *c)
{
  *n = *c + b++;
  *c = *n + 5;
  a++;
  return a + *n; }
void two (int d, int *s)
{
  *s = d + *s;
  printf ("%d %d %d \n", d, ++b, a);
}
```



Array :

int nums [5];

0	1	2	3	4
		10	16	7

subscript (index) from 0 -> (n-1)

nums[2] = 10;

nums[3] = nums[2] + 6;

nums [1+3] = 17;

could be eqation like (i + j + 3).

int A[4] = {5, 6, 2, 10}; → | 5 | 6 | 2 | 10 | A

int A[4] = {3, 4}; → | 3 | 4 | 0 | 0 | A

int A[4] = {0}; → | 0 | 0 | 0 | 0 | A

int A[4] = {1, 5, 2, 4, 7}; Error (X)

int A[ ] = {2, 3}; → | 2 | 3 | A

STUDENTS-HUB.com

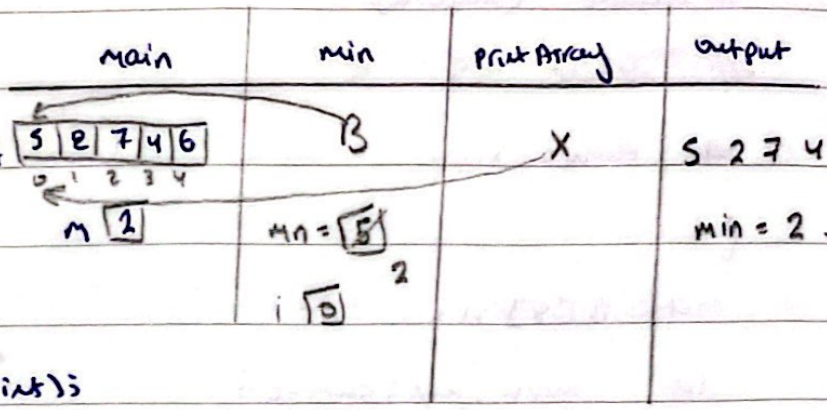
```
#include <stdio.h>
#define S 5
int main()
{
    int B[S], i;
    int mark, min; sum=0;
    float avg;
    for (i=0, i<S, i++)
    {   printf("Enter next marks\n");
        scanf("%d", &B[i]); }
    for (i=0; i<S; i++)
        sum += B[i];
    avg = (float) sum / S;
    min = B[0]
    for (i=0; i<S; i++)
        if (B[i] < min)
            min = B[i];
    for (i=0; i<S; i++)
        printf("%d___", B[i]);
    printf("avg = %.2f", avg);
    printf("min = %d", min);
    return 0;
}
```

60
70
90
30
100

60	70	90	30	100
0	1	2	3	4

B

Arrays in function :




```
#include <stdio.h>
#define S 5
int min (int [], int);
int printArray (int [], int);
int main ( )
{   int A[5] = {5, 2, 7, 4, 6};
    int m;
    m= min ( A, 5);        &A[0]      . [0] خلي ال عنصر اول عنصر
    printf ("min = %d\n", m);
    return 0; }
int min (int B[ ] , int n)
{   int mn = B[0];          function ال Pointer تعوض ك
    int i;                  It's can be *B .
    for (i= 0; i<n; i++)
        if (mn > B[i])
            mn = B[i];
                            should be void.
    return mn; }
int printArray (int X[ ], int n)
{ int i;
  for(i=0; i<n; i++)
  printf ("%d  ", X[i]); }.
```

linear search :

10	5	70	3	16
0	1	2	3	4

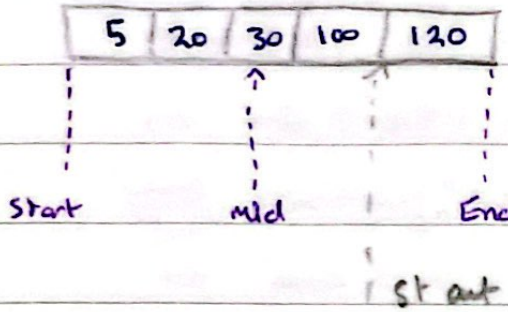
ids

```
#  include <stdio.h>
#  define  S  4
int  insearch (int [ ], int, int);
int  main ( )
{     int  key, A[S] = {20, 100, 3, 17};
      printf ("Enter key\n");
      scanf ("%d", key);
      pos = insearch (A, S, key);
      if (pos == -1)
          printf("%d no such key\n", key);
      else
          Printf ("%d is at pos %d", key, pos);
      return 0; }
int   insearch(int B[ ], int n, int k)
{     int i;
      for (i=0; i<n; i++)
          if(k == B[i])
              return i;
      return -1; }
```

binary serch:

• note!

- يجب ان تكون ال Arrays مرتبة تصاعدياً او تنازلياً .



$$mid = \frac{Start + End}{2} = \frac{0+4}{2} = 2.$$

Start .

the Bobble sort:

• A [5 | 2 | 7 | 20 | 3] → A [2 | 5 | 7 | 3 | 20] → A [2 | 5 | 3 | 7 | 20] ← A [2 | 3 | 5 | 7 | 20]

0 1 2 3 4

• عدد عناصر الـ Array = عدد المرات التي يقوم البرنامج بالمقارنة بها .

EX:

A [20 | 10 | 5] → A [10 | 5 | 20] → A [5 | 10 | 20]

0 1 2

```
for ( i =  ؛      ؛i++)        عدد الصفوف .
    for (j =  ؛   ؛ j++)       عدد الاعمدة .
```

Multi Dimensional Array:

int A[2][3];

rows colums

	0	1	2
0		5	
1			7

A[0][1] = 5;

A[1][2] = 7;

```
=> #include <stdio.h>
#define R 2
#define C 3
int main ( )
{  int A[R][C] , i, j, sum=0;
   for (i=0; i<R; i++)
      for(j=0; j<C; j++)
      { printf ("Enter value \n");
        scanf("%d", &A[i][j]);
      }
   for (i=0; i<R; i++)
      for (j=0; j<C; j++)
      sum += A[i][j];
}
```

note:

```
#define S 10
void reverstArray (int A[], int n);
{ int B[n];
```

يمكن تمرير نفس الـ Array أكثر من

• for min value :-

```
min = A[0][0];
for (i=0; i<R; i++)
    for (j=0; j<C; j++)
    { if(A[i][j]< min)
        min= A[i][j];
    }
```

• to print value :-

```
for (i=0; i<R; i++)
{   for(j=0; j<C; j++)
    ->printf("%d ", A[i][j]);
    printf("\n");   }
```

at the end of 2nd loop.

1 2 3

5 6 7

=>

```
#include <stdio.h>
#define R 2
#define C 3
void printArray(int[][C], int, int);
int main()
{   int A[R][C] = {{5,6,7},{2,3}};
    printArray(A, R, C);
    return 0;
}
void printArray(int A[][C], int r, int c)
{   int i,j;
    for(i=0; i<r; i++)
    {   for(j=0; j<c; j++)
            printf("%d ", A[i][j]);
        printf("\n"); }
}
```

```c
# include <stdio.h>
# difine   R 2
#difine   C 3
Void   printArray ( int [ ][C], int, int) ;
Void   add Array ( int [ ][C], int[ ][C], int[ ][C], int, int);
{  int   A [R][C] = { {5,6,7}, {2,3} };
      int  B [R][C] = {{1,5,2}, { 3,4}};
      int  D [R][C];
      - - - - - - - - but here.
      Print Arlay (D,R,C);
      add Arlay ( A,B,D,R,C);
      return 0;
}
```

note!

تم تخزين الـ Array في الذاكرة على شكل خطي .

1	2	3	4	5	6

(column indices above: 0 1 2 0 1 2; row indices below: 0, 1)

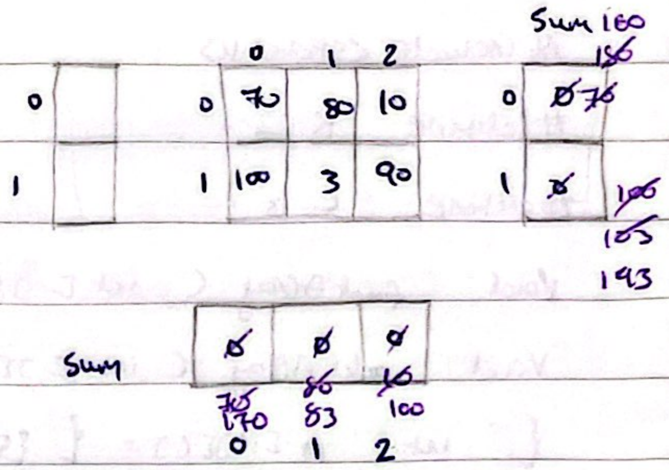
```c
   Void   PrintArray ( int  D[ ][C], int c, int r)
   {
         like  the  last  examper.
   }

Void  add Array ( int A[ ][C], int B[ ][C], int D[ ][C], int r, int c)
{    int  i, j;
      for( i=0 ; i<r ; i++)
           for (j=0; j<c ; j++)
                D[i][j] = A[i][j] + B[i][j] ;
}
```

• for (i=0; i<R; i++)

for (j=0; j<C; j++)

sum[i] += A[i][j];



sum[j] += A[i][j];

Strings.

char name[5] = "ali";

→ null char.

a	l	i	\0	
0	1	2	3	4

⇒ • char name[10] = "ahmad";

a	h	m	a	d	\0				
0	1	2	3	4	5	6	7	8	9

name.

• char name[ ] = "ali";

a	l	i	\0

• char name[5] = {'a', 'l', 'i'};

a	l	i		

a	l	i	\0	

{'a', 'l', 'i', '\0'}

⇒ • printf("%s", name); → ahmad.

• printf("Enter name \n");

suha

scanf("%s", name);

or

&name[0]

s	u	h	a	\0

```
char name[10];
int i;
printf("%s", name);
int cont=0;
i=0;
while (name[i] != '\0')
{   if (name[i] == a)
        cont++;
    i++; }.
printf(" cont = %d", cont);
```

Enter name.

asad

a	s	a	d	\0					
0	1	2	3	4	5	6	7	8	9

String function:

```
char name1[7] = "ali";
char name2[10];
name2 = name1 ;        ×   long word.
if (name1 == ali)      ×
if (name1 > "ali")     ×
```

we use #include <string.h> for string function.

1. strcpy (_,_);
2. strncpy (_,_,_);
3. strcat (_,_);
4. strncat (_,_,_);
5. strcmp (_,_);
6. strncmp (_,_,_);
7. strlen (_);
8. strtok (_,_);

(1) char S1[10] = "ahmad";
char S2[20];
strcpy (S2, S1); →

ينسخ S1 في S2 ويجب أن تكون S2 حجمها بقدر الحرف نسخه زائد 1

(2) strncpy (S2, S1, 2);
S2[2] = '\0';
إذا اردنا وضع النهاية

نسخ أول 2 char

وكيفية النهاية تلقائياً

(3) S1 a b \0 b c \0
S2 a b c \0
strcat (S1, S2);

(4) strncat (S1, S2, 1);
S1 a b a \0

first
last
strcat (first, 0);
strcat (first, last);

(5) strcmp → 0 / -1/-2, .. / +

if (strcmp (S1, S2) == 0)
(strcmp (S1, S2) > 0) S1 > S2
(strcmp (S1, S2) < 0) S1 < S2

6) strncomp (S, "ali");

7) strlen ("ali"); → 3

8) strtok — the most important one.

- char *token;
  char sent[100];
  - - - - - - - - - - ;
  gets(sent); ← printf("Enter any sentence\n");
  token = strtok(sent, " ");
  while (token != NULL)
  {
  token = strtok(NULL, " ");
  }

يقوم على فصل الكلمة بناءً على اي امر تريده من space.

Notes:

- char name[20];

Enter name.

Ahmad Nawaz.

a	h	m	a	d	\0	

So we use gets instead of scanf.

- gets(name);
- fgets(name);

from file.

- char names[10][20]; It's 19 letter but we put 20 because we have (\0) null.

```
i = 0;
while (i<10)
{ printf("Enter name");
  scanf("%d", name[i]);
  i++ ; }.
```

	0	1	2	3	4	. . .	20
0	a	1	1	\0			
1	a	h	m	ad	d	10	
2							
3							
:							
:							
:							
9							

```
#include <stdio.h>
#include <string.h>
#define S 10
int main ( )
{
  char sent[S];        char save[S],
  char *token;
  printf("Enter any sentence\n");
  gets(sent);
  token = strtok(sent, " ");      strcpy(save, sent);
  while (token != NULL)
  {
     printf("word = %s\n", token);
     token = strtok(NULL, " ");
  }
  return 0;
}
                    printf("Sentence after = \n");
                    printf("%s", save);
```

```
char    sent[100], save[5];
char    *token;
char    words[0][20];
int     i=0, j;
printf("Enter any sentence\n");
gets(sent);
strcpy(save, sent);
token = strtok(sent, " ");
while(token != null)
{
  strcpy(words[i], token);
  i++;
  token = strtok(null, " ");
}
for(j=i-1; j>=0; j--)
    printf("%s", words[j]);
return 0;
}
```

words

	0	1	2	.	.	.	.	19
0	t	h	e	b				
1								
2								
.								
.								
.								
9								

```
#include <stdio.h>
#include <string.h>
#define S 100
#define R 10
#define C 20
int main()
{  char *tok, int i=0, int j;
   char sent[S], temp[S];
   char words[R][C];
   printf("Enter sentence");
   gets(sent);
   strcpy(temp, " ");
   while (tok != NULL)
   {  strcpy(words[i], tok);
      tok = strtok(NULL, " ");
      i++ }
   for (j=0; j<i; j++)
      if (strcmp(words[j], "girl") == 0)
         strcpy(words[j], "boy");
      else if (strcmpy(words[j], "boy") != 0)
         strcpy(words[j], "girl");
   for (j=0; j<i; j++)
      printf("%s", words[j]);
   return 0; }
```

the word:

"the girl is smarter than the boy"

recursion :

- int f(int x).
  { return x+3 ; }

like we know.

- f(x) = f(x-1) + 3.

f(3) = f(2) + 3
f(1) + 3
f(0) + 3.

recursion :
we need base case.
like f(0) = 3 here.
∴ f(3) = 16

- factorial ...
- int fac(int n)
  { if ((n==1)||(n==0))
    return 1;
    else
    return n*fac(n-1); }.

fac(5) = 5*4*3*2*1.
5 * fac(4). base case.
5 * 4 * fac(3). fac(1) = 1
5 * 4 * 3 * fac(2)
5 * 4 * 3 * 2 * fac(1) 1.
∴ fac(5) = 120.

main	fec	
n = 5	n = 1	1 * fac(0) (1).
return fac(n).	n = 2	2 * fac(1) = 2
= 120.	n = 3	3 * fac(2) = 6
	n = 4	4 * fac(3) = 24
	n = 5	5 * fac(4) = 120.

Homework:-

```
int unknown (int x, int y)        try with (5, 4).
{
  if (y == 1)                     يعمل في الدالة (x) مرتين (y)
    return x;
  else                            its will be xy.
    return x + unknown (x, y-1);
}
```

main	unknown	
x = 5		
y = 4		
unknown (5, 4) : 20		
	x=5 , y=1	return x + unknown (y-1) 10
	x=5 , y=2	return x + unknown (y-1) 15
	x=5 , y=3	return x + unknown (y-1) 20

1) base case to avoid base case: recurive من شرط ال
2) compount intrest rule.

اذا لم نكتب هذه loop لا تنصب ال recurive.

• fibonaii series:

0 1 1 2 3 5 8 13 . . .

1 1 2 3 5 8 13 . . . . ∴ ratio = 1.6 .

(13 / 8 ≈ 1.6)

• int fib (int n)

```
{     if ((n==0)||(n==1))
          return 1;
      else
          return fib(n-1) + fib(n-2);    }.
```

fib (6) .

fib (5) + fib (4) .

fib(4) + fib(3)   fib(3) + fib(2)

fib(3) + fib2

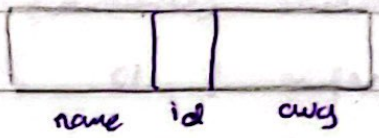
لا تحقق الشرط التالي ، كم مرات loop أشغل .

structwers:

name

0	a	l	i	\0
1				
2				
3				

ids	gender	avg
100	'm'	90.3
200	'f'	80.6
400'	'm'	
70		



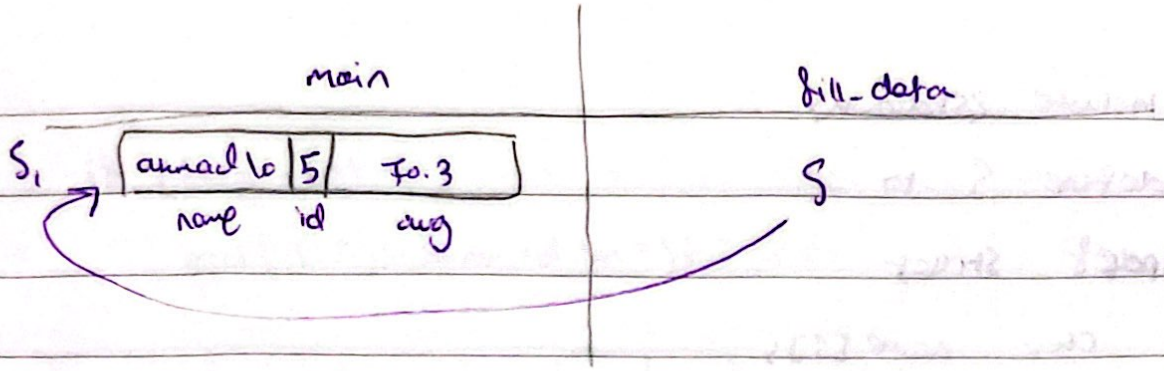

```
#include <stdio.h>
#define S 10
typedef struct { char name[S];
                 int id;
                 double avg; } Stud_t;
```


```
int main()                          void PrintStrc(Stud_t);
{ Stu_t S1; S2 = {"ali", "1000", "70"};
  printf("Enter name, id, avg of Student\n");
  scanf("%s %d %lf", S1.name, &S1.id, &S1.avg);   (string → no &)
  printf("name = %s\n", S2.name);                 PrintStrc(S1);
  printf("id = %d\n", S2.id);                     } can but PrintStrc(S2);
  printf("avg = %lf\n", S2.avg);
  return 0;
}
void PrintStrc(Stud_t Stu)
{ printf("name = %s\n", Stu.name);
  printf("id = %d\n", Stu.id);
  printf("avg = %lf\n", Stu.avg);
}
```

```
#include <stdio.h>
#define S 10
typedef struct
{ char name[S];
  int id[S];
  dauel avg[S];
} stud_t;
void stud_t fill_data (stud_t);
int main ( )
{ stud_t S1, S2 = { "ahmad", 5, 70.3 };
  S1 = fill_data (&S1);
  return 0;
}
stud_t fill_data ( )
{ stud_t temp;
  printf (" Enter name, id and avg of student \n");
  scanf ("%s %d %lf ", temp.name, &temp.id, &temp.avg);
  return temp;
}
void fill_data (stud_t *s)
{ printf (" Enter name, id and avg of student \n");
  scanf ("%s %d %lf", *s.name, &(*s).id, &(*s).avg);
}
```

- we can but $S_1 = S_2$ in structure
- but we can't say if ($S_1 == S_2$)

s->name, &s->id, &s->avg

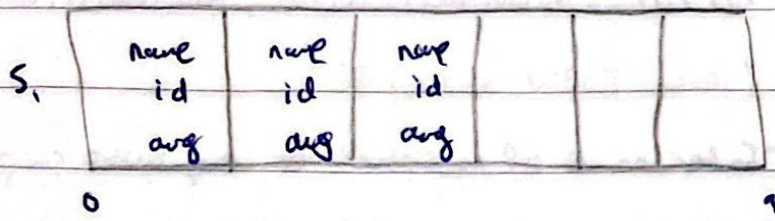



```
=> #include <stdio.h>
#define S 10
type def struct
{ char name[S];
  int id[S];
  dopel avg[S];
} stud-t;
int isEqual (stud-t , stud-t);
int main ( )
{ student-t  S1, S2 = {"almacl", 5, Fo.3 };
  if (isEqual (S1, S2) )
     printf ("same \n");
     return 0; }
int isEqual (stud-t S1, stud-t S2)
   { return (strcmp (S1.name , S2.name) == 0 &&
             S1.id == S2.id &&
             S1.avg == S2.avg )
   }
```

⇒

```
#include <stdio.h>
#define S 10
typedef struct
{   char   name[S];
    int    id[S];
    double avg[S];
}stud_t;
int main ( )
{   stud_t   students[S];
    int   id , flag=0;
    for (i=0; i<S; i++)
      {  printf("Enter name, id and avg of student\n");
         scanf("%s%d%lf", student[i].name,
&student[i].id, &student[i].avg );
      }
    printf("Enter id of student\n");
    scanf("%d", &id);
    for (i=0; i<S; i++)
      if (student[i].id == id)
        {  flag=1;
           printf("name = %s\n", student[i].name);
           printf("avg = %.2f\n", student[i].avg);
           break;
        }
```

```
if (flag == 0)
    printf(" no such id \n");
```



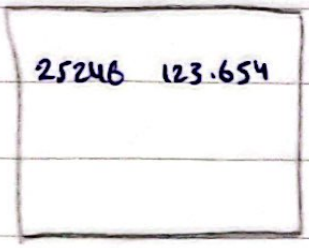
⇒
```
#include <stdio.h>
# define S 10
typedef struct
{ char name[S];
  int wid;
  double salary;
} worker_t;
int findworker (worker_t[ ], int, int);
void printworker (worker_t);
int main ( )
{
  worker_t worker[S];
  int i=0, size=0, id;
  printf("Enter name of worker or enough to stop \n");
  scanf("%s", worker[i].name);
```

```
while (strcmp(worker[i].name, "enough") != 0)
    {   printf("Enter id and salary\n");
        scanf("%d %lf", &worker[i].wid, &worker[i].salary);
        i++;
        printf("Enter name of worker or enough to stop\n");
        scanf("%s", worker[i].name);
    }
    size = i;
printf("Enter id of worker\n");
scanf("%d", &id);
pos= find worker(worker, size, id);
if (pos == -1)
    printf("No such worker\n");
else
    print Worker(worker[pos]);
return 0;
}
```

```
int findworker (worker_t w[], int s, int id)
{  int i;
   for (i=0; i<s; i++)
      if (id == w[i].wid)
         return i;
   return -1;
}
void PrintWorker (worker_t wk)
{
   printf("name = %s\n", wk.name);
   printf("id = %d\n", wk.wid);
   printf("Salary = %.1f\n", wk.Salary);
}
for (i=0; i<size; i++)
   PrintWorker(worker[i]);
```

Binary Files.

text files



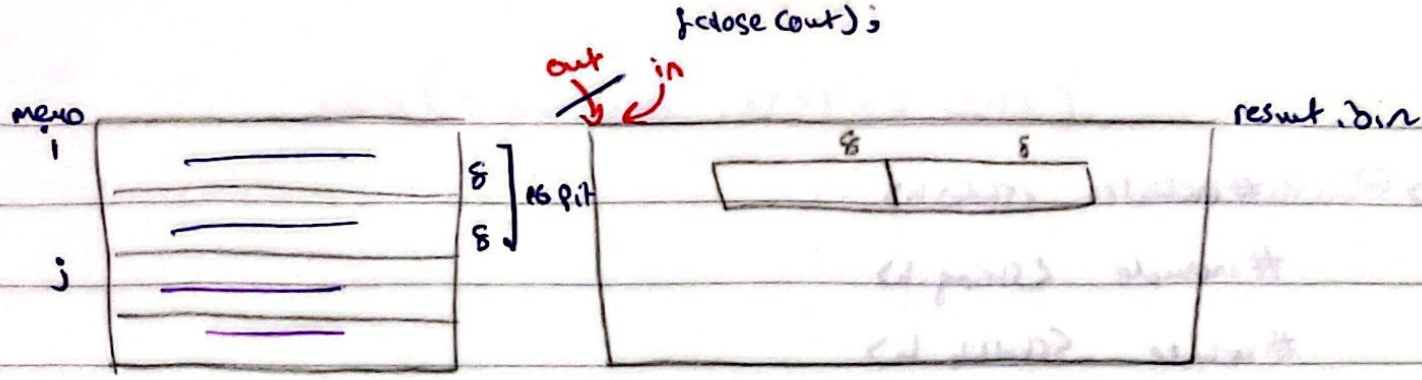
int = 16 bit.

```
FILE *in, *out;
in= fopen("data.bin", "rb");   not important in the code.
out = fopen("data.bin", "wb");
int i= 5;                      عدد عناصر الكتابة
fwrite (&i , sizeof(int) ,1, out);
```

عنوان المراد كتابته — pointer — حجم المراد كتابته

صاحة المراد كتابته بالذاكرة.

```
fclose (out);
in= fopen("result.bin" , "rb");
fread ( &i , size of(int) ,1, in);
fclose (in);
printf ("j= %d ", j);   ⇒ j=5
```



⇒ type def struct

{ char name[10];

int id } stud_t;

int main ( )

{ FILE * in ;

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

in = fopen ("data.bin" , "wb");

fwrite (&5, sizeof (stud_t), 1, in);

fwrite (arr, sizeof (int) , 5, in);

int arr2[10], size ;

fclose(in);

in = fopen("data.bin" , "rb");

size = fread (arr2, sizeof(int), 10, x);

5

=>

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
int main ( )
{ FILE *in;
  char filename[10];
  printf("Enter input file name \n");
  scanf("%s", filename);
  in = fopen(filename, "r");
  if (in == NULL)
  { printf("%s no such file\n", filename);
    exit(0);
  }
  printf("keep going \n");
```

or

```
  while (in == NULL)
  { printf("%s no such file", filename);
    printf("Enter input file name \n");
    scanf("%s", filename);
    in = fopen(filename, "r");
  }
  printf("good \n");
```

while (cin >> roper, "r") == NULL)

تكون بد ✓ ✓