

# My Journey To Learn Flutter

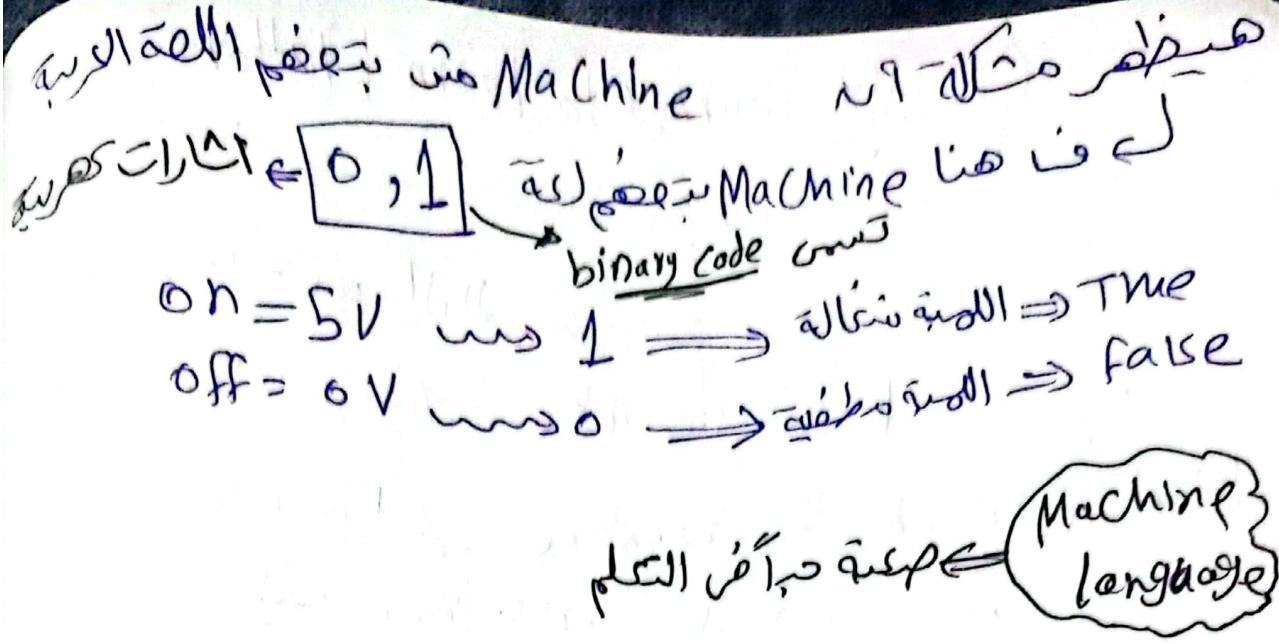
مقدمة في مجال البرمجة  
يعني (ا) برمجة ← العرض اسأد برمج .  
يعني (ا) برماج ← مجموعة من الخطوات يتم تنفيذها  
خطوة بـ خطوة لتنفيذ Task .  
هذا برماج معين

(ا) فم تبسين الماء في زاد النار  
(ب) ضع كيس النار في الكوب  
(ج) انتظر حتى سخن الماء  
(د) ضم الماء الساخن على القيس  
(ه) ازلا كيس النار بعد بضمها  
(ز) ضع ملعة من السكر  
(ي) قلب جيداً

لوعني ؟! تحفنا لبعض الخطوات دى [ برماج معين ] صغير  
نسمة [ Task ] يعني [ دى ] انى دى كل كوب اسما  
شرط دا الشخص ده يلقوه يفهم اللغة البرمجية دا  
لقدر دفهم البرنامج دا .

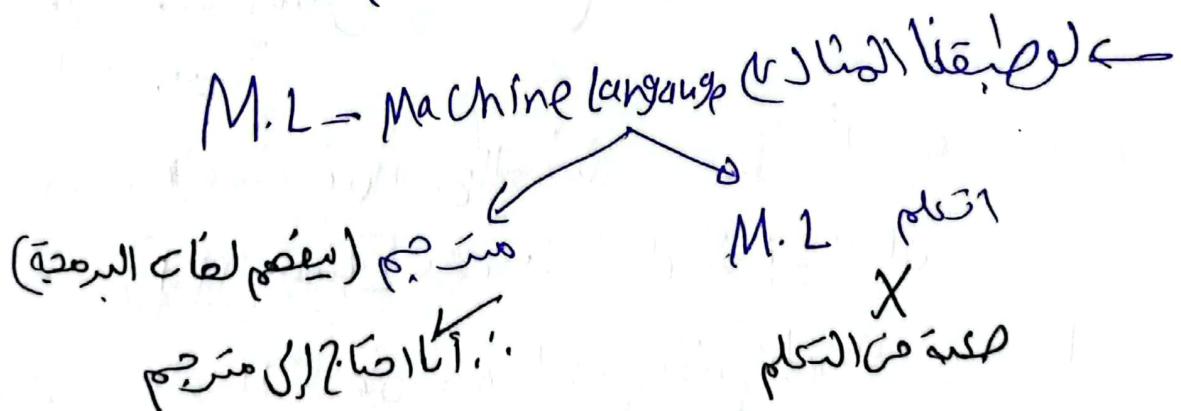
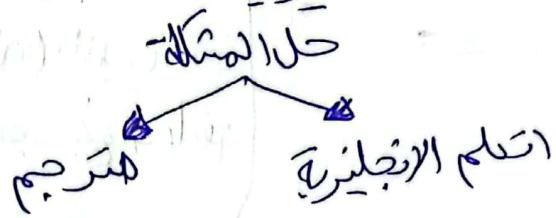
طلب طالب ادا ماير اكتب برماج علما الفتن  
Machine [ Computer ] ... ل مسند هستيم للخطوات دى



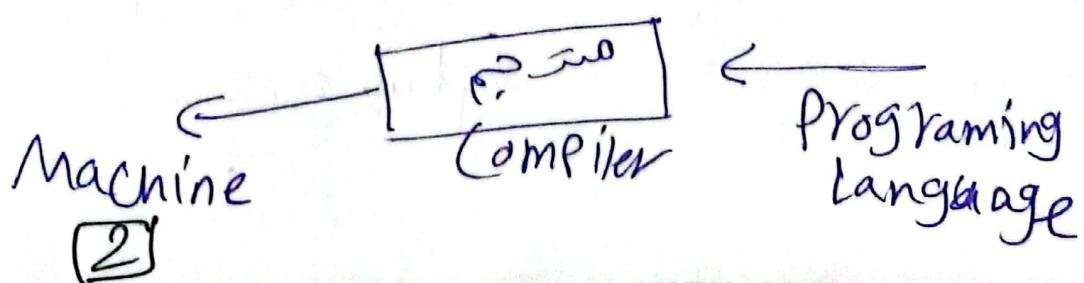


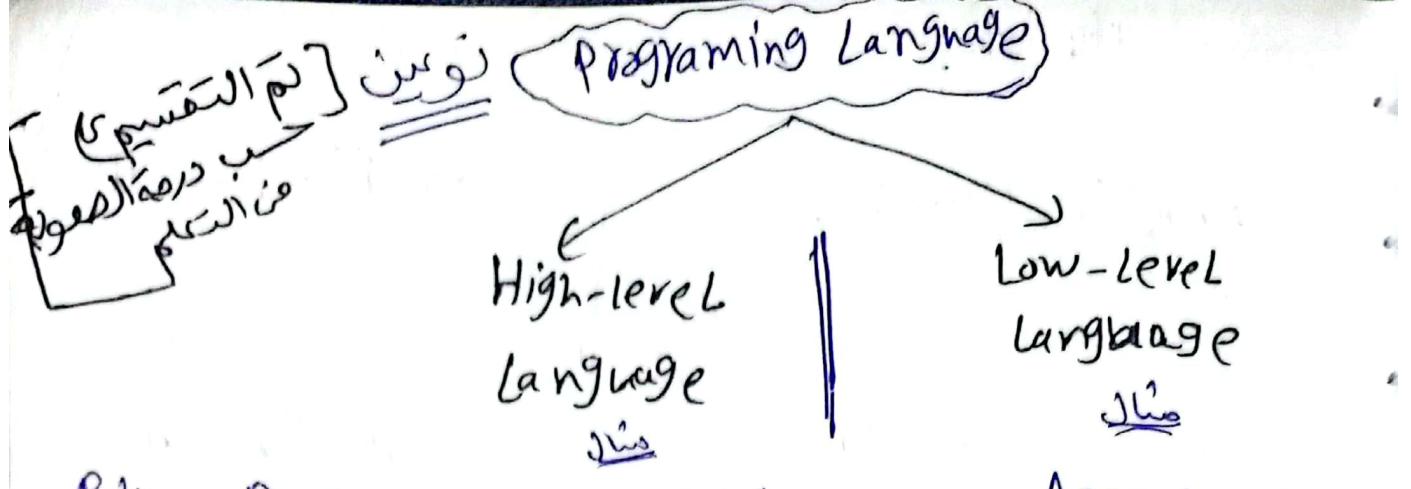
طبعاً أنا أعلم الكلمة دى

اعرض ٢٩ هسأفتر دوّله ما ولنكن امرأنا  
عهل البلد يتصدرون باللغة الإنجليزية عايم لا جيد الترجم  
باللغة الانجليزية



تعلم  $\leftarrow$  سهل لارقا عباره نه لله عقاب و مفاسد  
من المفاصي العادي الإنجليزى





Python - Dart - C++ - Java - Kotlin  
C# - Objective-C .....

غير منسورة مكتوبة باللغة  
الإنجليزية

```
#include <iostream>
using namespace std;
int main()
{
    return 0;
}
```

Assembly Lang.  
Machine

101101101110  
1010101110111.

main function

```
Void Main()
```

Dart (لغة برمجة معاصرة، سهلة ومحببة)  
ستعمل على Android و iOS و Flutter.  
Flutter ← Frame work (هيكل)  
أصدرها Google في 2011.  
لها دعم جوجل.  
هيكل تصميم الواجهات (UI).

→ Print Statement → خطوة أو أمر أو فعل

```
Void main()
```

```
{
```

```
    Print ('Hello World')
```

```
}
```

3] Hello world

run دليل

Terminal (دخل)

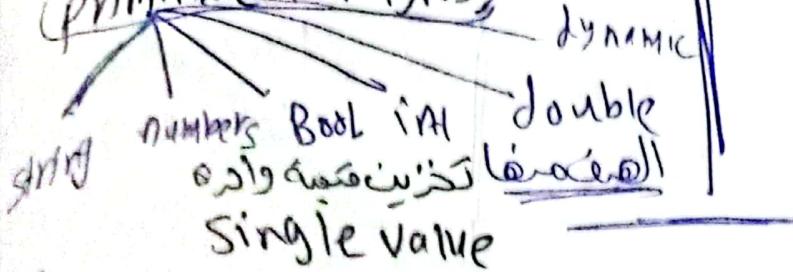
أكسي

لهم إله

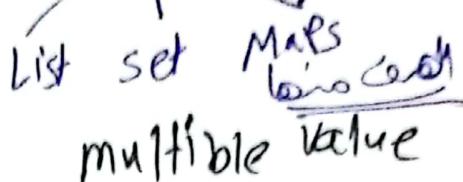
Dart main.dart

# Data types

## Primitive Data types



## Collection Data types



①

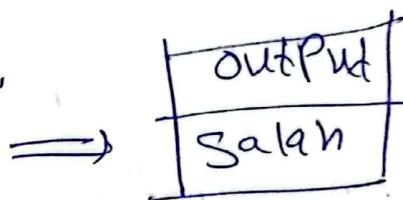


طبع من نوع البيانات تتعامل مع النصوص

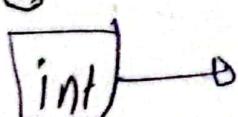
single rotation " " ← كائنات تكون بين  
double rotation " " ← اى نص تم كتابة تكون بين

void main()

```
{ String name = "Salan";  
print(name); }
```



②

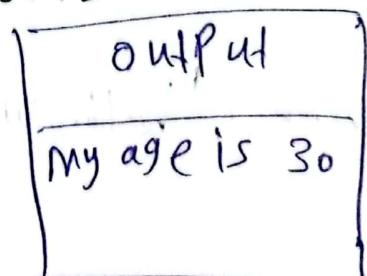


طبع من نوع البيانات تتعامل مع الأرقام الصحيحة

void main()

```
{ int age = 30;  
print("my age is $age"); }
```

}



4

٤) double → نوع من نوع البيانات التي تتعامل مع الأعداد الحقيقة والأرقام الكسرية  
6.1 / 8.7 / 9.3 / 2.5 ← الأرقام الكسرية

Void math()

{

double gpa = 2.3;

Print ('my gpa is \$gpa');

}

Output  
my gpa is 2.3

٥) numbers → نوع من نوع البيانات التي تتعامل مع الأرقام  
الصحيحة والأرقام الكسرية

↓  
أصناف

num

أصناف للأرقام

Void main()

{

num rating = 4.3;

Print (' rating is \$rating');

}

Output  
rating is 4.3

٦) boolean → إما صحيحة أو خطأ

always

bool

Yes/No ≡ True/False ≡ 0,1 ≡ On, off ≡

Void main()

{  
    bool(isBusy)=True;

    Print(isBusy);

}

Output  
True

5

٦) **dynamic** → نوع من أنواع البيانات التي يمكنها أن تكون متغيرة ونوع البيانات الذي جاء معه من المبرمج

مثلاً أفرضاً متغير `x` يختلف في نوعه من الستراتجيات

dynamic `x` يمكنها أن تأخذ قيمة int أو float

dynamic `x` يمكنها أن تأخذ قيمة string

`void main()`

{

`dynamic x = 13;`

`dynamic name = 'salah';`

`dynamic y = True;`

}

**dynamic**

↓

weak data type

متغير

`int - num - bool - double - string`

↓

Strong data type

**Var key word**

→

data type متغير

Keyword لـ var

`Var`  $\xrightarrow{\text{بيان نوع المتغير}}$   $\xrightarrow{\text{بيان نوع المتغير}}$  dynamic

Var text;

لـ dynamic

`void main()`

→ text = 10;

→ text = 10.5;

→ text = 'Islam';

Var name = 'salah'; → name  $\Rightarrow$  string

Var age = 10; → age  $\Rightarrow$  int

Var gpa = 2.3; → gpa  $\Rightarrow$  double

{} ↴

## Arithmetic operators

العلاءات المعاصرة

World Maine)

```
int num1=15;  
int num2=10;  
int result=num1+num2;  
print(result);
```

output  
25

```
void main ()  
{
```

```
int x = 10;  
int y = 20;  
int result;
```

```
result = num1+num2;  
print(result);
```

```
result = num1 * num2  
print(result), output
```

result = num1 - num2  
print(result), Output

result = num1 / num2; output

result = num1 % num  $\Rightarrow$  zero

فِي اسْكَانِ الْمُتَعَلِّمِ  
وَلِنَجْعَلَهُ مَسْجِدًا لِلْأَعْلَامِ

لے گئے اصل تبدیل و میلوں سے میں چھوٹا ہو گیا۔

٦) حلین  $\leftarrow$  دلیل  $\leftarrow$  خبر  $\leftarrow$  دلیل  $\leftarrow$  دلیل  $\leftarrow$  دلیل  $\leftarrow$  دلیل

1-36

1-1218 ←

result = num1 / num2

$$J_{\text{sub}} = \frac{J_{\text{sub}}}{n_{\text{sub}}} \approx \frac{J_{\text{sub}}}{n_{\text{sub}}} \rightarrow J_{\text{sub}} \approx \frac{J_{\text{sub}}}{n_{\text{sub}}}$$

شیخ + ی = شیخی



## Dot Operator

الخطوة  
لـ دى جى دى فى الظاهر أو دى فى الواقع فى الصيغة المحررية

```
void main()
{
```

```
    String name = 'Salah';
```

```
    print(name.length)
```

Output  $\Rightarrow$  5

```
}
```

```
Print(name.toUpperCase());
```

Output
SALAH

## Collection data type

List  
Map  
Set

خزن متغير البيانات  
أو تكرار البيانات

① Lists

List < data type >  
string, int, ...

$\Rightarrow = \{ value, value, \dots \}$

```
void main()
{
    List <String> names = [Salah, Hassan, Wael]
```

```
    print(names);
```

Output [Salah, Hassan, Wael]

```
    print(names[0]);
```

Output Salah

```
    print(names[1]);
```

Output Hassan

```
}
```

8

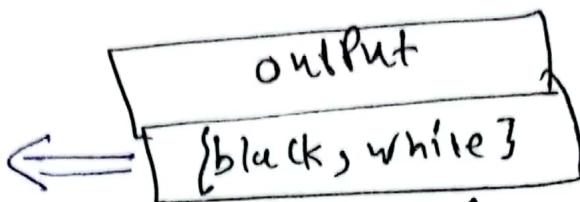
② Set

Set < datatype >  $\text{set} = \{ \text{value}, \text{value}, \text{value}, \text{value} \}$

void main()

Set < string > favColors = {'black', 'white', 'black'}

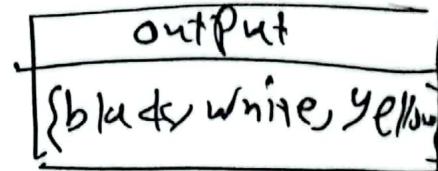
Set  
اسئل



print(favColors);

favColors.add('yellow');

print(favColors);



③ Map

void main()

unique identifier (key, value) لـ key لـ value

Map < string, num > ages = {'ahmed': 15,

'kareem': 20, 'wael': 16, 'saber': 12}

print(ages['kareem']);  $\rightarrow$  kareem

print(ages['wael']);  $\rightarrow$  wael

list  $\rightarrow$  []  $\rightarrow$  مكتبة العناصر

maps  $\rightarrow$  {}  $\rightarrow$  key, value

sets  $\rightarrow$  {}  $\rightarrow$  منفرد العناصر

arabic operators  
arabic translation

> < و = و >= و <= و !=  
أكتر من أقل من أو مساوي أو أقل من أو مساوي  
كثير من قليل من أو مساوي

if

## conditional

Void main()  
{

    bool isWeatherGood = true;

    if (isWeatherGood)

    {

        print("go to the sea")

    }

}

## logical operator

or → ||  $\Rightarrow$  سطراً واحداً صحيح كله صحيح

And → &&  $\Rightarrow$  جميع الأسطر صحيح

Void main()

int ~~score~~MathScore = 70;

int ArabicScore = 60;

if (MathScore >= 50 && ArabicScore >= 50)

{

    print("You passed");

}

10

## if - else

```
void main()
{
    bool isWeatherGood = false
    if (isWeatherGood)
    {
        print("go to the sea")
    }
    else
    {
        print("go to the cinema")
    }
}
```

## Nested if

[ested if ] if جوisi

if - else if - elseif ----- else

```
void main()
{
    bool isAdmin = true
    bool isModerator = false

    if (isAdmin)
    {
        print("you are Admin")
    }
    else if (isModerator)
    {
        print("you are moderator")
    }
    else
    {
        print("you are user")
    }
}
```



2 3

## switch

```
Void main () {  
    string grade = 'A'  
    switch (grade) {  
        case 'A':  
            cout << "excellent";  
            break;  
        case 'B':  
            cout << "very good";  
            break;  
        case 'C':  
            cout << "good";  
            break;  
        case 'D':  
            cout << "fair";  
            break;  
        default:  
            cout << "invalid grade";  
    }  
}
```

## Scope

```
Void main () {  
    // main function scope  
    if ( )  
    {  
        // if scope  
        switch ( )  
        {  
            // switch scope  
        }  
    }  
}
```

(2)

3

3 // switch scope

## Ternary operator

If-else-like

Condition ? expr If True : expr If False

```
void main()
```

```
{ int x = 10; int y = 20;
```

```
int max = 0;
```

```
if (x > y)
```

```
    max = x;
```

```
else
```

```
{    max = y;
```

```
}
```

```
printf("The max is %d\n");
```

```
}
```

Output
The max is 20

## Ternary operator

similar

```
void main()
```

```
{
```

```
int x = 10; int y = 20;
```

```
int max = (x > y) ? x : y;
```

```
printf("The max is %d\n");
```

else

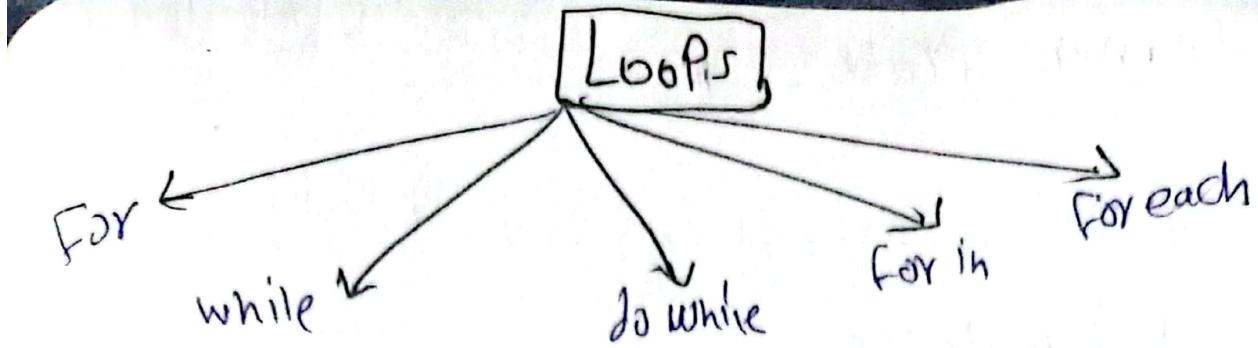
```
void main()
```

else

```
var score = 75;
```

```
var result = (score >= 60) ? "Pass" : "Fail";
```

```
} printf(result);
```



**For**

عندما نريد ان نكرر مجموعه من اعمالي

for (initialization; condition; increment/decrement)  
{

// بيمارس الالغاز

}

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

{

Print(i);

}

Output:
0
1
2
3
4
5
6
7
8

**While**

عندما نريد ان نكرر مجموعه من اعمالي حتي تتحقق شرط معين

while (condition){

// statements

    // increment or decrement

}

Do

void main()

{ int i=1

    while (i<=10){

        Print(i);

        i++;

}

Output
1
2
3
4
5
6
7
8
9
10

## Do while

الدوال مبروحة

do {

// statement 1 ;

// statement 2 ;

⋮

// statement N ;

} while ( condition ) ;

↓  
↙

void main()

{ int i = 1 ;

do {

print(i);

i++;

} while ( i <= 10 );

}

Output
1
2
3
4
5
6
7
8
9
10

## For in

←

index <sup>index last</sup> <sub>first</sub>

void main() {

List<int> ages = [ 19, 14, 23, 15, 17 ]

for( var item in ages)

{ print(item); // 19, 14, 23, 15, 17

(لولیتہ کریمی)

for each گوچیف

IS

for each

void main()

{  
List <string> names = {"Salah", "Hassan"}  
}

names.forEach((var item) {  
print(item);  
})

Null Safety → [الحاجة إلى]  
[null safety]

لتحصل على مثبات non-nullable و المثلث صيغة  
أي مثبات إلى المثلث

لتحل محل nullable عبارة عن مثبات و المثلث  
مثبات صيغة المثلث صيغة المثلث أي حول  
منها المثلث  
والرقم مثل  
@ check int result;

void main() {  
int num1;  
int num2 = 10;  
int result = num1 \* num2; // error  
Print result → error

}

(16)

```

void main() {
    int? number1;
    int number2 = 16;
    if (number1 == null) {
        int result = number1 ?? number2;
        print(result);
    }
}

```

→ [?] → null operator  
 → number1 => nullable  
 (or null and then what)

3

[?]   
 nullable  
 non-nullable  
 into  
 nullable

3 Dart lemosi  
 non-nullable

1   
 nullable  
 non-nullable  
 into  
 null ? -

2?   
 if else  
 int x; int y = 16;  
 result = x ?? y;  
 print(result); //16

## Function

6 [5] [4] [3] [2] [1]  $\leftarrow$  Blue box ← loops  
[2] [1] [3] [4] [5]  $\leftarrow$  Green box ← functions

[1] = [2] = [3] = [4] = [5]  $\leftarrow$  Blue box ← functions  
[1] = [2] = [3] = [4] = [5]  $\leftarrow$  Green box ← functions

## Basic Function

```
void main() {  
    printMyStory(); // function
```

}

```
void printMyStory()
```

{

```
    print("my name is salah");  
    print(" my age is 22");  
    print(" I love coding");
```

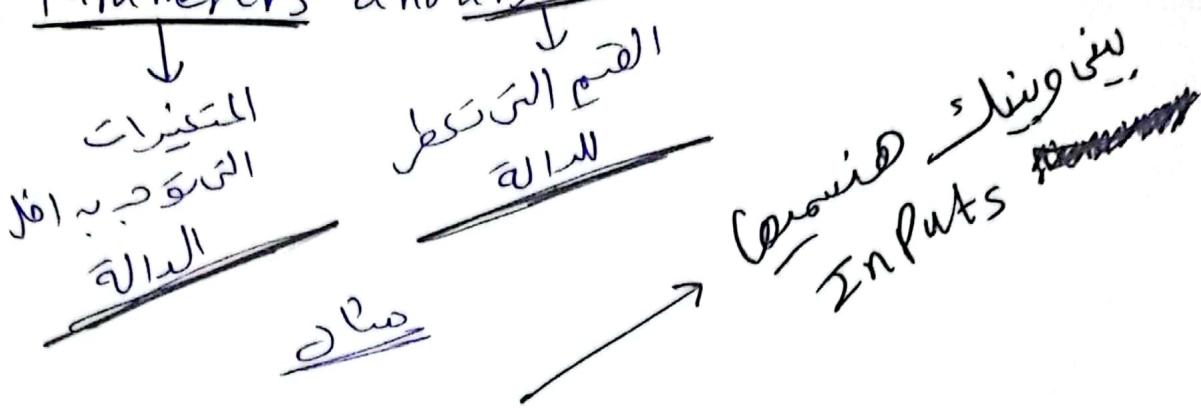
}

## Return Statement

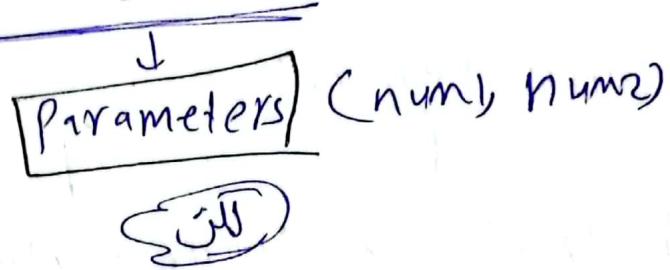
```
void main() {  
    int result = sum();  
    int num3 = result + 5;  
    print(num3);  
}  
int sum() {  
    int num1 = 15;  
    int num2 = 40;  
    int result = num1 + num2;  
    return result;
```

18]

## Parameters and arguments



```
int sum ( int num1 , int num2 ) ;
```



```
fat sum ( 50, 20 ) ;
```



### Optional Parameters

كذلك

لارجومات أخرى (أيضاً آخر)

فعلاً الآخر  
error

```
void main ( ) ;  
print ( calc ( 800, 10 ) );  
print ( calc ( 100, 40 ) );  
print ( calc ( 100 ) );  
print ( calc ( 70 ) );
```

```
}  
double calc ( double price, { double discount = 0 }  
{ double finalPrice = price - price * discount / 100;  
return finalPrice;
```

}

19

Named Parameters → optional [ ] مُفعّل من

double sum ( double num1, double num2 )

Named Parameters

void required double price ;  
هذا هو price

### ① Positional Parameter

الترتيب [ ]

مُحْفَظ

void greet ( string name, int age )  
print ("Hello \$name, I am \$age years old");

void main()

{ greet ('Israh', 20); → أوائل الترتيب  
}

الترتيب  
أوائل الترتيب  
error

① Named Parameter	Optional Parameter [ ]
من الترتيب	الترتيب [ ]
void greet( string name, int age ) { print ( "Hello ", name ); print ( ", I am ", age ); }	Positional → الترتيب [ ]
{ greet ( age:25, name:'Salah' ); } error	Named → الترتيب من المسمى
② Required	Required → لزم سك قيمة
جبار، تحط المقدمة error	Optional → من المسمى تحط قيمة
required string name لازم تحط الاسم	

(2)

ODP

البرمجة الموجهة لل Đối / البرمجة الموجهة للبيانات

## → object oriented programming

➊ superclass (class)

➋ Class (object) + functions  
attributes

Salah → ➊ Human object  
➋ Height, skin color, ...

Car → ➊ Toyota  
➋ color, speed

Class

```
void main() {
    Human salah = Human();
    salah.name = "hassan";
    salah.age = 22;
    print(salah.age);
}
```

Class Human {

```
int? age;
String? name;
double? height;
double? weight;
```

variables  
functions  
+ surfaces  
Class Capital

11/20

21

3

## Method

function must  
class belongs

Void main() {

Human x = Human();

$$x \cdot 0.9e = 22;$$

X' name = 'salam'

X. `read()`; // reading is useful

3

## Class Human (J)

8

Int? age j

Survey? name:

void read()

```
print(if reading is useful )
```

3

3

الله

## Class

## Properties

~~class~~ → attributes / fields

color, age, name

مسار

↳ , class  $\Rightarrow$  methods

walk, ran, sleep

JLW

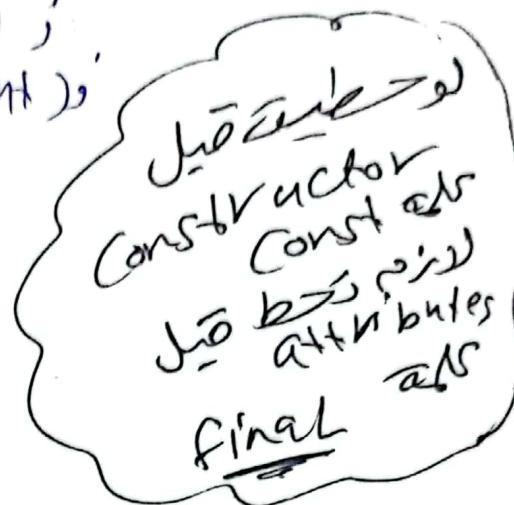
22

Constructor → جملة by default مكتوبة  
جملة new هي المثلثة التي

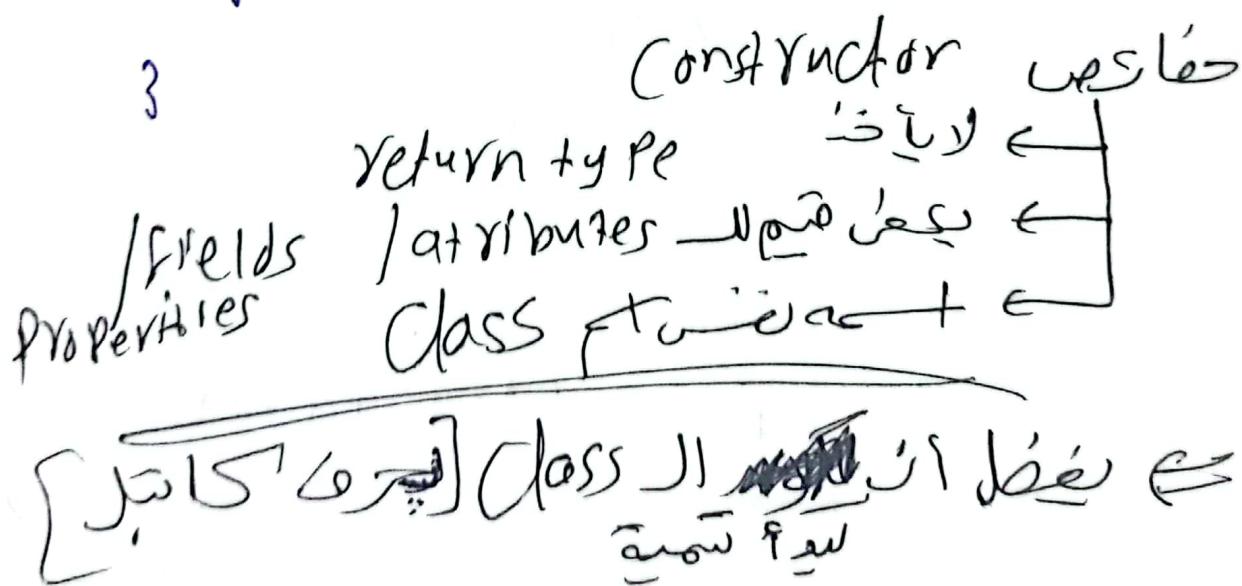
Void main() {  
Human x = Human(Salman, 22, 173.0);  
print(Human.name);  
print(Human.age);  
print(Human.height);  
}

Class Human {

final String name;  
final int age;  
final double height;



const Human (String name, int age, double height)  
{  
this.name = name;  
this.age = age;  
this.height = height;  
}



Encapsulation

private attribute via a file  
private access also in class  
private member attribute  
ENCapsulation (is not used)

void main() {  
 import 'Human.java';

Human salam=Human(173, 80)

salam.setNumberofArms(20);

Print(salam.getNumberofArms()); // 2

}

Class Human {

{ int numberofArms=2;  
double height;  
double weight;

Human(double height, double weight)

{ this.height=height;  
this.weight=weight; }

} void SetNumberofArms(int numberofArms)

{ if (numberofArms <= 2 || numberofArms > 10)  
{ }

{ this.numberofArms=numberofArms; }

} int GetNumberofArms()  
{ int return this.numberofArms; }

(24) ?

## Encapsulation

private  
attribute مُخْصِّصٌ لـ "this"  
attribute ترجمة مُخْصِّصٌ لـ "this"

method مُفْتوحٌ (1)  
method مُفْتوحٌ (2)  
method مُفْتوحٌ (3)

## Setter

labeled  
Set inf ()  
{

}

↓  
this

Set NumberofAVMs (int numberofAVM)  
{

if (NumberofAVMs <= 2 && NumberofAVMs >= 0)

{  
    this.numberofAVMs = NumberofAVMs

} }

25

## ARROW FUNCTION

like no, but it's a function

Int getCode()  $\Rightarrow$  this. - code

it is  $\leftarrow$  Getter

int get  $\leftarrow$  ~~get~~ code  $\Rightarrow$  this. - code;

## inheritance

extends  $\leftarrow$  it's part of all the class

class

```
void main() {
    Lion SCar = Lion();
    Dog Jack = Dog();
    Cat Kitty = Cat();
    SCar. eat();
    Jack. eat();
    Kitty. eat();
    SCar. Roar();
    Jack. bark();
    Kitty. meow();
}
```



[26]

Class Animal

{  
int numberoflimbs = 4;

eat () {

Print ('eating');

}

String skinColor;  
double weight;  
//Constructor

Animal ({ required this. skinColor,  
, required this. weight}).

Class Lion extends Animal

{

void roar () {

Print ('roaring');

}

//Super constructor

Lion ({ required super. skinColor,  
Required super. weight});

}

Class Dog extends Animal

{

bark () {

Print ('barking');

}

Lion ({ required String skinColor;  
Required double weight});  
: Super (skinColor: skinColor,  
, weight: weight);

}

Class Cat extends Animal

{

Meow () {

Print ('meowing');

}

3

27

Override فایل وارد کردن از پایه وی میتواند ۲

```
eat()
{
    Print("eating");
}
```

```
@override
void eat()
{
    print("lion is eating");
}
```

### Abstract Class

Abstract Class Animal

```
{  
    eat();
}
```

```
}
```

Class Lion extends Animal

```
{
    @override
    eat()
    {
        " "
    }
}
```

Implements

extends ~~abstract~~ ~~Super~~  
inheritance ~~multiple~~ ~~class~~

Class Animal

{

}

Class Lion extends Animal

{

}

implements (L) ~ (A)

لهموا ز

- derived class must implement methods

@Override

پیویسی

کر

public class extends Animal

Named constructor

void main()

{ Circle circle1 = Circle (radius: 5, x: 3, y: 4);

circle1.draw();

Circle circle2 = Circle (radius: 5);

circle2.draw();

}

class Circle {

double radius;

int? x;  
int? y;

late keyword

late int x;  
late int y;

Circle () { required this.radius, required this.x, required

this.y; }

draw() {

print('draw this circle at x=\$x and y=\$y with radius \$radius');

circle·origin {{required this radius}}  
{  
x=0  
y=0  
}  
}

// named constructor

### Mixins

DAT گوئیں کہ اسی میکسین کا استعمال کرنے کے لئے اسے کسی دو یا کاملاً مختلف کلاس کا میکسین کرنا چاہیے۔ اسے کاملاً مختلف کلاس کا میکسین کرنا چاہیے۔

void main()  
{

Dog jack ~ Dog();  
jack.walk();

}

Class Animal

{ int numberofLimbs = 2; }

}

// mammals یعنی

// reptiles رواحی

~~Class~~ MammalsMixIn {  
mixIn walk();  
}

}

mixIn

~~Class~~ ReptilesMixIn {  
crawl();  
}

58

Class Dog extends Animal With mammals

    walk() {}  
}

Class Cat extends Animal With mammals

    walk()

}

Class Snake extended Animals With Reptiles

{

    crawl()

}

object class

void main() {

    Dog Jack = Dog();

    List <Object> = Jack;

}

    class Animal() {}  
    class Dog extends Animal  
    {  
        int weight;  
        Dog(this.weight);  
    }

    class Cat extends Animal {}

anonymous

    Object object as b.  
    {  
        a  
    }

    List <Animal> animals;  
    [Dog(b0), Cat(b1)];

**Enum** → **enum** gender { male, female }

→ **string gender = male;** this  
 male female

→ **string month = 'Jan';**  
 Jan, Feb, March, April, May, ... like

```

void main()
{
    Gender gender = Gender::male;
    switch (gender) {
        case Gender::male:
            break;
        case Gender::female:
            break;
    }
}
    
```

**enum Gender { male, female }**

**exception** → run time error

Print (code);  
 Syntax error → // syntax error  
 logical error → // logical error

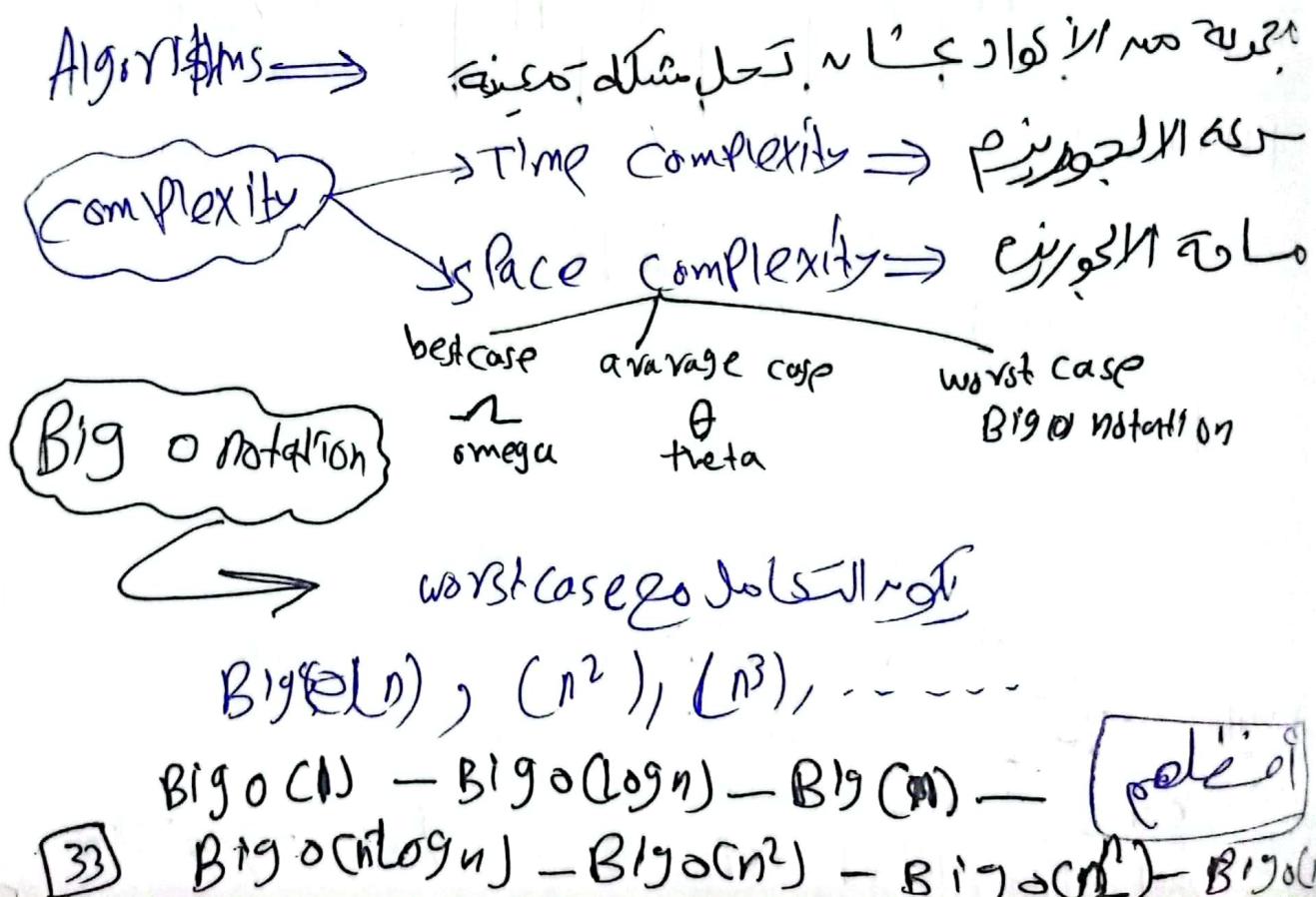
(3) zero →

```

void main()
{
    List<string> names = { 'salan' },
    try {
        names[5],
    }
    catch (e)
    {
        print("Sorry there is an exception")
    }
}

```

## Data Structure and Algorithms



## Linear search Algorithm:-

Index $\rightarrow$	0	1	2	3	4	5
List $\rightarrow$	3	5	1	4	0	17

Condition for loop to continue  
current value is less than target value

④ increments the index

loop ends if current value is equal to target value

return index for 4 value

loop exits

## Binary search Algorithm:-

Index      0    1    2    3    4    5

Left	22	20	17	13	11	9	Right
------	----	----	----	----	----	---	-------

Right = 9      Left = 22      الناتج من المدخل

Right, left  $\Rightarrow$  اول خط لخط

$$Mid = \frac{Right + Left}{2} \Leftrightarrow Mid$$

Mid  $\Rightarrow$  الناتج من المدخل

Initial Mid  $\Rightarrow$  Mid value

Mid no < target  $\Leftrightarrow$  مناسب

Right = Mid + 1       يجعل

loop until mid < target

Mid + 1 = Left       $\Leftrightarrow$  مناسب

and Mid no > target

# Linear $\rightarrow$ Binary search

## Linear

$\Rightarrow$  Data is not sorted and  
[مختلط غير مرتب]

$\Rightarrow$  Not efficient of  
large list

$\Rightarrow$  Time complexity:  $O(n)$

$\Rightarrow$  Slower

## Binary search

$\Rightarrow$  Data is sorted [مكتوب]

$\Rightarrow$  Efficient of large  
list

$\Rightarrow$  Time complexity:  $O(\log n)$

$\Rightarrow$  Faster

## Selection Sort Algorithm

الخوارزمية تختار العنصر الأقل  $\leftarrow$

$\Rightarrow O(n^2) \Rightarrow$  nested loop  $\square$  for  $\sim L^2$   
for  $\Rightarrow$  for

[10, 5, 14, 33, 100, 70]  $\leftarrow$  Input

[5, 10, 14, 33, 70, 100]  $\leftarrow$  Output

[100, 70, 33, 14, 10, 5]  $\leftarrow$  Output

$\Rightarrow$  Very slow

DS

# Recursion

الاستدعاء الذاتي ←

فانكشن تستخدم نفسها ←

لـ مترتبة معرفة الـ

$$5! = 5 \times 4 \times 3 \times 2 \times 1 \rightarrow 5 \times 4!$$

$$4! = 4 \times 3 \times 2 \times 1 \rightarrow 4 \times 3!$$

$$3! = 3 \times 2 \times 1 \rightarrow 3 \times 2!$$

$$2! = 2 \times 1 \rightarrow 2 \times 1!$$

$$1! = 1$$

void main()

{  
    print(fact(5)) ;     // 120

3

int fact (int num)

{  
    if (num == 1)

{  
        return 1 ;

}

    return num \* fact(num - 1) ;

}

36

Stack) دیکھ لے جو List or Array نہیں

(Last In First Out) LIFO

Stack میں اول وادا خروج سے خدا کیا

Stack جو دل کرے کیا

10 ←

Stack جو دل کرے کیا

20 ←

30 ← stack جو دل کرے کیا

40 ← stack جو دل کرے کیا

2 جسے اول وادا خروج سے خدا کیا

10 ← & 30 ← 40 ← 2 جسے

Ctrl + Z ← Redo    stack کے کام پر نہیں  
stack کے function

push, pop, isEmpty, top, size ---

