Subject:.....

/ /

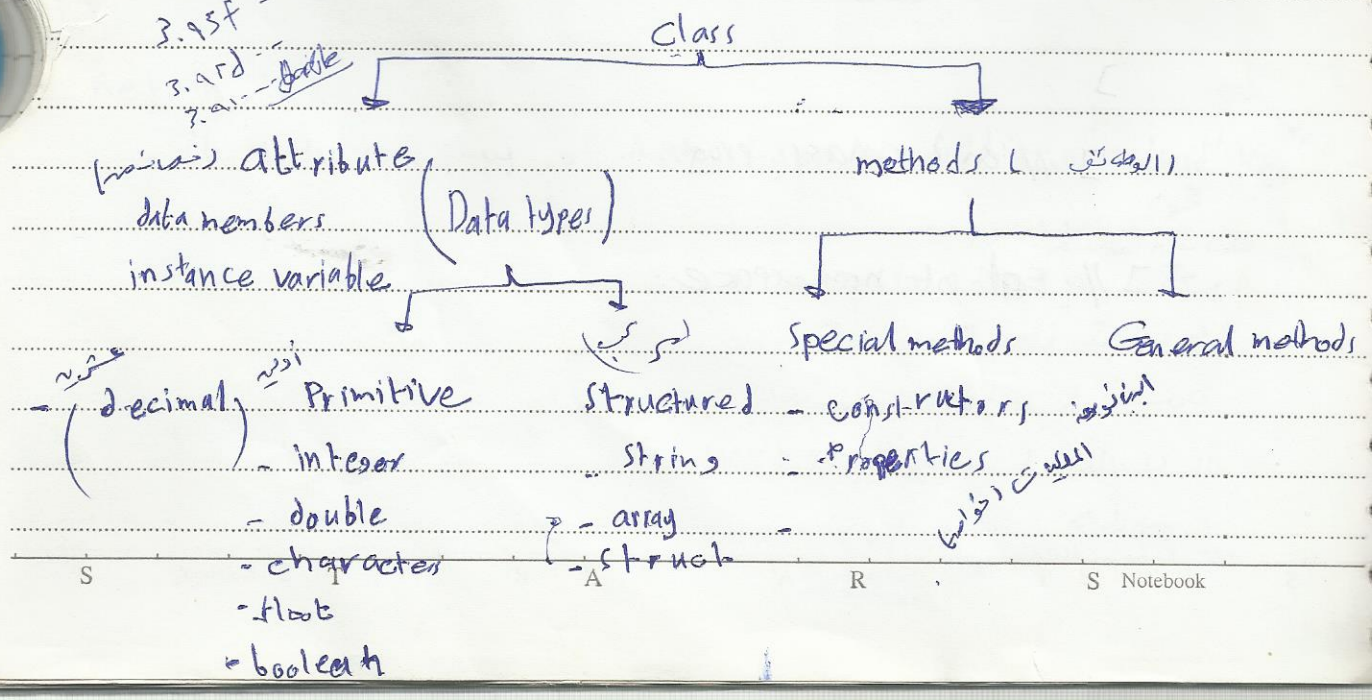
Abstraction: it is an oop concept where we concentrate (emphasize) on the importance and ignore the least important (details)

(Attributes & methods ⇒ class)

Encapsulation: is another oop concept, where the features (attributes) and operations on these features (methods) are put togethers in one thing (unit)

Data hiding: it means prevent the user from accessing the features (attributes) except through the operations (methods)

Class: is the templete that we create objects from.
is the blue print that we create objects on → based.
it is the basic unit of oop that enable us to create objects.



Subject: / /

```
using System;

namespace oop
{
    class student
    {
        int ID;
        string name;
        double avg;
        public student(int a, string n, double c)
        {
            ID = a;
            name = n;
            avg = c;
        }
    } // End of student class
    class prog
    {
        public static void main()
        {
        }
    } // end of class prog
} // End of namespace.
```

(naming conventions)

int studentMarks;
int studentmarks;

constructor

Attributes

Entry point of the program.

Subject:..... / /

* General methods:- (class) (class) (method)

Access_modifier Retuin_Type Mothed_Name (list of parameters optional)

{ // statments

}

Access modifier: (method)

List of keywords that determine who can access and use the method.

List of kws:
- public
- private
- protected
- internal

Access: public → private

Restriction: private → public

* The default values of access maditiers for mehtols ont attribtes are private.

Retuin Type:
- it includes any of data types, in addition to "void" kw

S T A S Notebook

Subject:

Method name:

- it should express for show the fanctionality of the method

list of parameters (optional)

- Data types
- seported by comma

- we should pay attention when using a method by:
  - number, types and order of parameters

```
public int Avg (int x, double y, char z)
{   int sum = 0;
    int avg = 0;
    sum = x + y + 2;
    return avg = sum / 3;
}
```

z = 0

sum	avg	x	y	avg
0	0	0	0	0

Subject:........................ / /

*spical methods :

- Constructors: هو دالة ينفذ تلقائيا لل (objects) "جديدة"

من خلاله (inshlaztion) قيم مبدئية التي النهاية ممكن بعد الاستدعاء.

- كل (class) يجب ان يكون به على (constructor

* Goal (هدف): is to initializes the attributes of the class.

- every class must has at least (على الأقل) one constructor.

- A class can have more than one constructor. [1, 8)

-

في حالة عدم وجود (const) يتم انشاء تلقائيا (deform con)

في حالة إنشاء واحد يحذف ونستطيع انشاء (default con) بعدها بكتابته

* الإفتراضي يعطي (deform) معطيات غير ابتدائية Data type ل

- "Default constructor" created for a class that has No contructor
↳ by the compiler

- once you create a constructor the "default constructor" will be deleted.

- the "default" constructor can be created.

- We can create many constructors as we wish.

- The default constructer initializes the attributes (data members) with their default data Type values.

Subject:...................... / /

* why constructors are special? — ما سرّ المميز ؟؟
1. must exist in every class — يجب أن يكون داخل الـ (class)
2. it has the same name as the class. — اسمه يجب أن يكون نفس اسم الـ (class) أو حتى (default)
3. has No Return types. — لا يرجع أي قيمة، ليس له "(retain type)" ولا حتى (void)

```
class Car
{   private
    String color;
    int year;
    double enginesize;
    public Car (        )      // default constructor — the same name class
    {

    }
    public car (  int y       )
    {
        year = y;
    }
    public car ( int x, double b)
    {
        year = x;
        enginesize = b;      // (color crossed out)
    }
```

لا نستطيع الوصول إليه لأنه private
الوصول إليه بالكود
~~يجب أن~~ تكتبه
لا يرجع إذا كتب ←

S T A R S Notebook

A = vml(5, Any)

Subject:…………………… / /

```
class University
{  string name;
   string location;
   int phone;
public university (string n, string L, int ph)
{
this. name = n;
this. location = L;
this. phone = ph;
}
```

no error

(string name, int phone)
{ this.name = name;
this.phone = phone;
}

name, location, phone, constructor

X location = this.location X error

* الاوله والاخير نفس class
الثاني يختلف كلهم
حيث نستطيع التغيير
فقط من حيث الترتيب
البارومتير.

"this" نستخدم نميز بين اسماء المتغيرات المتشابهة
this نعوض عن القيمة (value) بيعود لـ (obj)

dot operator

→ refers to the instance // operator
of the current object of the class
تعود عن القيمة (obj) الذي انشئ فيه.

حيث تم الاستدعاء
بحسب ترتيبها.
في الاخر يختلف اسماء
البارميتر

```
public university (int phone, string name)
{
 this. name = name;
 this. phone = phone;
}
```

وهنا لا نستطيع
التغيير لانه عندما
نستخدمها نفضلها تتم
لما حتى لو حذفت

```
public university (string X, int y)
{ name = X;
  phone = y;
}
```

احد الاكواد بيأخذها
وتتم هذه العملية
(Method overloading)
↓
(constructor ...) لو نستخدم اكثر من constructor

S T A R S Notebook

Subject: A

Method overloading :- الدوال (سن) بتختلف عن بعض بعدد و ترتيب و انواع المتغيرات .

Constructor overloading : have more than one constructor in the class that differ in number, type or order of parameters.

(Setter, getter)

Properties :- (city) ← اوبراتور عند طريق (pratice) تعرف (method)

- ~~is to be~~ (data members) تتعامل مع الدوال التي

- enable to access the private data members (attributes)

access → modify ← تغيير قيمة (تعديل) Data members ; Read

- Each private attributes one property :-

كل (pratice data) تستخدم (property) one

General Form :- (one property → one attribute)

```
Access Modifier   return_Type   Method_Name          لا يحتوي (param)
  {
     get { return attribute_Identifier ; }            * لا توجد دائما الاقواس
     set { attribute_Indeitifier = value ; }
  }
```

ex.

```
class Rectangle
{  double height ;           instance variable
   double weidth ;
   public Rectangle ( double height , double weidth )
   {  this. height = height ;       int z = 0 ; → Local variable
      this. weidth = weidth ;
   }
   public double Myheight
   {  get { return = height ; }
      set { height = value ;
   }
```

Subject:..... / /

```
public double theWidth
{ set { width = value; }
  get { return width; }

}
}
```

Subject:..... / /

object :- is an Instance of the class.

Genral Form :-

Class_Name identifier = new constructor ( ) ;

```
namespace Exam
{
class University
{ string name;
  int phone;
  string location;
public university (string name, string location, int phone)
{ this.name = name;
  this.phone = phone;
  this.location = location;
}
public string Name
{ get {return name;}
  set {name = value;}
}
}
```

public university ( ) { } default

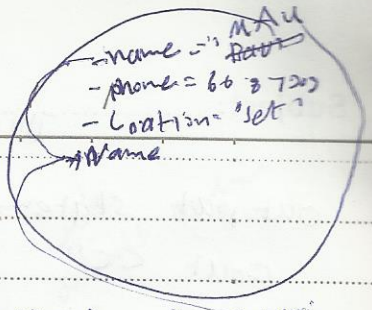
"public int phone;"

set ; set ; } default value.

user program

ObjectName.MethodName( ) ;

Subject:.....



Class program

{ public static void main ( )

{ university Myuni = new University ("BAU", "Salt", 668778) ;

Myuni . Name = "MAU" ;

dot operator

Unjresity Uni1 = new ("Jordan", "Amman", 991889") ;

University Uni2 = new university ( ) ;

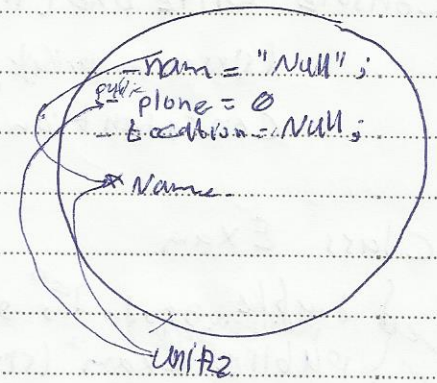
Uni2 .Name = "JusT" ;

Uni2 . phone = 123456 ;

string X = Uni2 . Name ;

}

}



* data meber Public → property

if the attribute (Data member) is defined as public member then No need for Property method to change or return it's value;

S T A R S Notebook

adress to the object.

Subject: This: refers to the instance of the class

out put statement:- object . object method

cout << " "; ≡ Consloe. Write ("welcome ");

cout << " " endl; ≡ Consloe. WriteLine (" Hello");

method → overloading . 18 times.

int x = 10;

Consloe. write (x); 10welcome

Console. write ("welocome";

Console. write line ("Hello" + x + " " + x); عم

(string + variable) دمج

Concatention.

واحدة (method في

Hello10 10

نسيطم المحافظ دفع الكلام نفس

Class Exam

{ double grade1 = 0, grade2;
public Exam (string name)
{ this. name = name;

public void Change Value (double x, double grade2)
{ grade1 = x;
this. grade2 = grade2;
}

public double MyValue ( )
{
return grade2;
}

public void Display ( )
{ Console. write line ("My grade2 value is" + My value ( ));
}
}

grade1 grade2 here

method header signature object

method heading

my grade value is ...

S T A R S Notebook

Subject:..... / /

```
Class Prog
{ public static void main()
  { Exam e1 = new Exam("oop");
    Exam e1 = "oop"   (x, 123.59);

    Exam e1 = new Exam(x);
    e1.grade2 = 99.6;

    e1.Grade2 = 95.6;

    Console.WriteLine(e1.Myvalue());
    double y = e1.Myvalue();
    Console.WriteLine(y);
    y = y + 1.99;
    y++;
    e1.Display();
    e1.ChangeValue(95.6, 3.95);
```

My grade1 value is 95.6

Subject: .... / /

Today :-

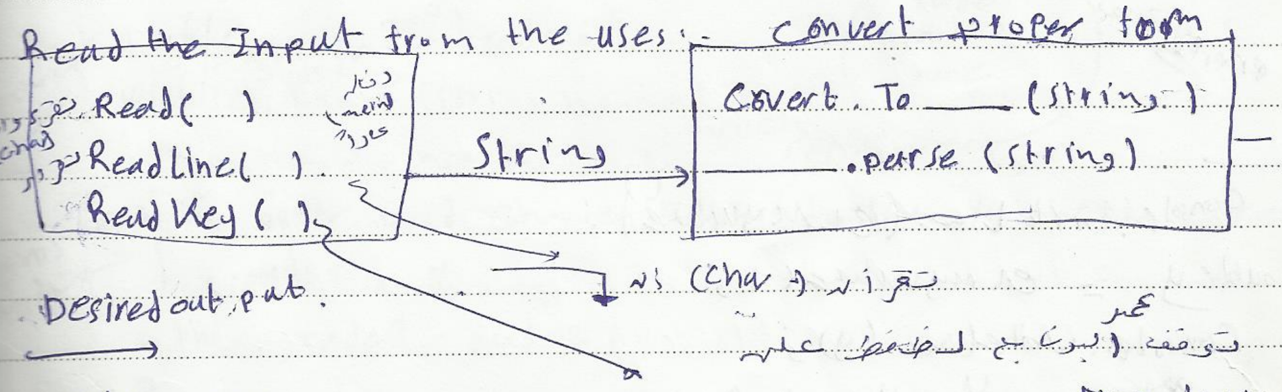
١- إيه الـ Inputs في الـ C# في القراءة كـ (string)

٢- ثم تحويله إلى الشكل اللي تريده.

عن طريق (method)

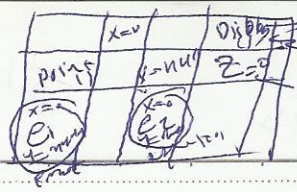
* Taking input
  - Read ( ) → (overloading) عدد ١٩
  - Readline ( )
  - Readkey ( )

* static keyword
  - static members.
  - static class.



Class prog
{ public static void main( )
{
Console. Writeline ("place your grade from 0 to 100");
× String S = Read ( ) ; تقرا أول حرف فقط
string S = console. Readline( ) ;
doubleX = Convert. To Double (S) ;

≡ double X = Convert. To Double (console. Readline ( )) ;



Subject:.....

Console.WriteLine ("Place enter your year of Birth");

string w = Console.ReadLine();

int y = Int.parse (w);

= int y = Int.parse (Console.ReadLine());

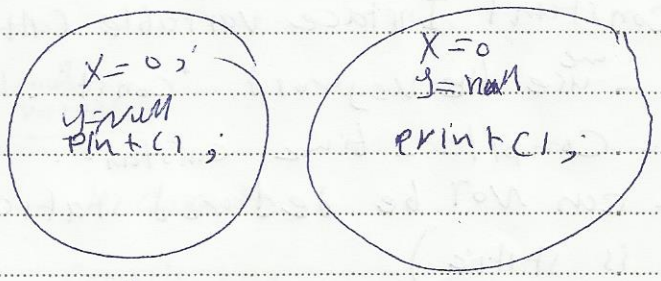
if (y < 1900 || y > 2016)

Static keyword:-

① static class

② static method

③ static Attribute.



① static class:

- can NOT be instantiated (can NOT create object)
- can NOT use "new" keyword.
- static class contains static methods and attributes.

```
class MyClass
{ int x; string y; static int z;
  public static void Display()
  { int sum = (x); }   // X error
  public void print()
  { Console.WriteLine(x,y); }
  public void setZ (int x)
  { z = x; }
  public int getZ()
  { return z; }
```

```
class Prog
{ public static void main()
  { MyClass e1 = new MyClass();
    e1.Display(); // Error
    MyClass.Display();
    MyClass e2 = new MyClass();
    e1.setZ(20);
    int x = e2.getZ();
    e1.number;
    MyClass.number = 1000;
    e2.setZ(35)
    int xy = e1.get(x)
```