Advanced C# Notes

Session 01

هنتكلم عن ال Genarics :

بدأت تكون موجودة فى 2005 ولما ظهرت بقينا نستخدمها ومبقناش نستخدم الاوبجكت .

هنطبق عليها بمثال ونعرف الفرق بين ال GenericType , non GenericType

non GenericType

```
class Helper // هنعمل كلاس جديد نضيف في ميزود معينة وليكن بتعمل سواب
    {
        public static void SWAP(ref int num1 , ref int num2)
        {
            int temp = num1;
            num1 = num2;
            num2= temp;
        }
    } // هنطبق دا ف البروجرم

        static void Main(string[] args)
        {
            int x = 2;
            int y = 4;
            Console.WriteLine($"before swap : x = {x} , y = {y}");
            Helper.SWAP(ref x, ref y);
            Console.WriteLine($"After swap : x = {x} , y = {y}");
        } // مثال  طيب لو انا عايز اخليها تعملي سواب ل دابل بدلا من انتجر هضطر اني هعمل واحدة جديدة اعمل اوفرايت عليها واعلمها نيو بيهفور

        #region Swap 2 Double
        public static void SWAP(ref Double num1, ref Double num2)
        {
            Double temp = num1;
            num1 = num2;
            num2 = temp;
        }
        #endregion     // هنطبق دا ف البروجرم

            #region Swap 2 Double
            Double x = 2.2;
            Double y = 4.7;
            Console.WriteLine($"before swap : x = {x} , y = {y}");
            Helper.SWAP(ref x, ref y);
            Console.WriteLine($"After swap : x = {x} , y = {y}");
            #endregionمثال  طيب لو انا عايز اخليها تعملي سواب ل بوينت هضطر اني هعمل واحدة جديدة اعمل اوفرايت عليها واعلمها نيو بيهفور

        #region Swap class (point)
        public static void SWAP(ref point num1, ref point num2)
        {
            point temp = num1;
            num1 = num2;
            num2 = temp;
        }
        #endregion // هنطبق دا ف البروجرم
```

```
#region swap class (point)
point p01 = new point(3, 4);
point p02 = new point(5, 6);

Helper.SWAP(ref p01, ref p02);

Console.WriteLine($"{p01} ,|| , {p02}");
#endregion
```

طيب لحد امتا هنفضل نعمل اوفرلودنج وكلهم بينفذو نفس الشئ ونفس الاكشن بس اللى كنت بغيره هو الداتا تايب ف هنا بقا احنا مش هنعمل كل دا ومن هنا تيجي فكرة الجنراك

GenericType

```
#region GenaricType Example 1
#region SWAP 2 Integer
int x = 2;
int y = 4;
Console.WriteLine($"before swap : x = {x} , y = {y}");
Helper.SWAP<int>(ref x, ref y);
Console.WriteLine($"After swap : x = {x} , y = {y}");
#endregion

#region Swap 2 Double
Double k = 2.2;
Double m = 4.7;
Console.WriteLine($"before swap : x = {k} , y = {m}");
Helper.SWAP<Double>(ref x, ref y);
Console.WriteLine($"After swap : x = {k} , y = {m}");
#endregion

#region swap class (point)
point p01 = new point(3, 4);
point p02 = new point(5, 6);

Helper.SWAP<point>(ref p01, ref p02);

Console.WriteLine($"{p01} ,|| , {p02}");
#endregion
// In Case GenericType "T" is Declared on method Level , Not Class Not interface Not Struct
Compiler can Detect the Type of "T" Based on the Type of input Parameters مع الميزود لو انا مضفتش
نوع الجنراك الكومبيلر هو بيضيفه من نفسه وفقا للبراميتر اللى انا مدخلها غير كدا مبيضفش مثال
 int x = 2;
int y = 4;
Console.WriteLine($"before swap : x = {x} , y = {y}");
Helper.SWAPref x, ref y);
Console.WriteLine($"After swap : x = {x} , y = {y}");

#endregion
```

ال == مش موجودة داخل بعض الداتا تيب زي ال STRUCT فبالتالى بتضطر انى اروح اعملها ب نفسي واعمل اوفر رايت على ال == واعلمها حاجة جديدة وهي ان هي تقدر تستخدمها داخل الاستراكت فقبالتالى لازم تكون برايفت وتكون ولازم تكون بترجعلي بولينboolen وعشان تشتغل معايا لازم اعمل واحدة للنوت اكول مثال

```
// example override on operator
struct employee
{
    //public int id { get; set; }
    public string name { get; set; }
    public int salary { get; set; }

    public override string ToString()
    {
```

```
            return $"{id} , {name} , {salary}";
        }

        // example override on operator
        public struct bool operator ==(employee left, employee right)
        {
            return left.Equals(right) // short code
            //return (left.id == right.id) && (left.name == right.name) && (left.salary ==
right.salary);
        }
        public struct bool operator !=(employee left, employee right)
        {
            return !left.Equals(right) // short code
            //return (left.id != right.id) || (left.name != right.name) || (left.salary !=
right.salary);
        }
    }     // وهكتب كدا فى البروجرم بتاعى

            employee emp1 = new employee() {id = 1,name = "ahmed",salary= 6000 };

            employee emp2 = new employee() { id = 2, name = "ali", salary = 4000 };

            if (emp1 == emp2) // must be override new operator
            {
                Console.WriteLine("equal");

            }

.// Search Method

        #region Generics Search example
        public static int SearchArr(T[] arr, T value)
        {
            if (arr is not null)
            {
                for (int i = 0; i < arr.Length; i++)

                    if (value.Equals(arr[i]))
                        return i;
            }
            return -1;
        }
        #endregion  //هنعمل كدا فى البروجرم بتاعى

            #region Generics Search example
            employee emp1 = new employee() { id = 1, name = "ahmed", salary = 4000 };
            employee emp2 = new employee() { id = 2, name = "ali", salary = 3000 };
            employee emp3 = new employee() { id = 3, name = "omar", salary = 2000 };

            employee[] employees =
            {
                emp1,
                emp2
            };

            int resulte = Helper<employee>.SearchArr(employees, emp2);
            Console.WriteLine(resulte);
            #endregion
```

Equality [Struct , Class]

ال == غير موجودة في الاستراكت ف احنا لما بنيجي نقارن الكوالتى عندى 2 اوبشن اول اوبشن هو انى استخدم == اما 2 اوبشن انى استخدم Equal method اللى بورثها من الاوبجكت فبالنسبة للاستراكت معندوش باي ديفولت == فلما باجي اقارن الكوالتى فى الاستراكت بقارنها بال Equal method بقوله id.Equals(id) مثلا ... قمثلا لما باجي اقارن اوبجكت بالاكول ميزود عندنا ف الاستراكت بيروح يحفظ الاوبجكت دا ف الاستاك على عكس الكلاس بيروح يحفظ الريفرنس بتاعه ف الاستاك وبيحفظ الفاليو في الهيب

اما ال == موجودة فى الكلاس عادي فهي بتقارن ادريس او ريفرنس او ايدنتتي

مثال اخر بالتطبيق على BubbleSort Algorthem

```
        #region bubbleSort Method Example (in class helper)
    class Helper<T> where T : IComparable
        // T in class and struct munst be include interface >> Icomparable
// T must be class or struct and implmenting the built-In Generic Interface -> Icomparable<T>
class Helper<T> where T : Icomparable<T>
    {
        public static void BubbleSort(T[] array)
        {
            if (array is not null)
                for (int i = 0; i < array.Length; i++)
                    for (int j = 0; j < array.Length - i - 1; j++)
                        if (array[j].CompareTo(array[j + 1]) == 1)
                            SWAP(ref array[j], ref array[j + 1]);
        }
        #endregion

        #region bubbleSort Exmple in (employee class)
        public int CompareTo(object? obj)
        {
             // الكومبير تو جايه ب اوبجكت بيقا انا لازم احول الاوبجكت دا الى امبلوي عن طريق الاكسبليست كاستنج
  employee passedEmp = (employee) obj; // Explicit casting, unsafe casting // may throw Exception
            return this.salary.CompareTo(passedEmp.salary);

            /// اخري طريقة
            ///if (this.salary > passedEmp.salary)
            ///    return 1;
            ///else if (true)
            ///    return -1;
            ///else
            ///    return 0;
        }
        #endregion

            #region bubbleSort in Program file
            employee[] employees =
            {
                new employee(2,"Ahmed",20_000),
                new employee(1,"Ahmed",10_000),
                new employee(3,"Ahmed",30_000)
            };

            Helper<employee>.BubbleSort(employees);

            foreach (employee emp in employees)
                Console.WriteLine(emp);
            #endregion
```

// employee passedEmp = (employee) obj;3 هنا فى مشكلة وعشان نحلها عندى 3 طرق وافضل طريقة هي رقم

1. Is condational operator >> بترجعلي ترو فى 3 حالات وهما
    1. Obj is point

```
      2. Obj is an object from Another class inheriting from point [3D point]
      3. Obj is null
   // بس هتقابلني مشكلة وهي انى مش هعرف انفذ الحالة رقم 2 ففى الحالة دي هستخدم as operator
```

```
2. as operator
              employee passedEmp = obj as employee;
              if (passedEmp is null) return 1;
   // casting will succeded if obj is point
   // if casting failed, will return NULL. Zero Exception will be thrown
```

3. Generic Interface >> Icomparable<T> >> هنطبق بمثال على الاوبجكت وهيكون علي ملف البوينت اللى عملناه فوق

```
    class point : IComparable<point>
    {
        public int x { get; set; }
        public int y { get; set; }
        public point(int _x , int _y)
        {
            x = _x;
            y = _y;
        }
        public override string ToString()
        {
            return $" x = {x} , y = {y}";
        }
      #region another example to solve Explicite casting
        public int CompareTo(point? other)
        {
                if (other is null) return 1;

                if (this.x == other.x)
                    return this.y.CompareTo(this.y);
                else
                    return this.x.CompareTo(this.x);
        }
    }

      #endregion

            #region another example on class point (in program file
            point[] points =
            {
                new point(3,4),
                new point(2,6),
                new point(2,3),
                new point(1,1),
            };
            Helper<point>.BubbleSort(points);

            foreach (point point in points)
            {
                Console.WriteLine(point);
            }
            #endregion
```

الجنارك جميلة وحلوة وكل حاجة بس لازم تكون حذر وانت بتتعامل معاها ومن مشاكلها انك متقدرش تستخدم معاها الoperators زي ال + - == وهكذا `public static T sum(T x, T y) {return x+y;} // invalid`

Generics Constraint

ودي بتكون عبارة عن شرط او شروط انا بحطها على الجنارك وعندى منها 3 انواع وهما

1_ primary Constraint [0:1] `Helper<T> where T :class, Icomparable<T>`

. 1. General Primary Constraint
T must be >> Class /or struct /or Enum
لو قولتله حاجه من دول يبقا انا شرط عليه ان لازم تستخدمه على

2. special primary Constraint (user-defined class (except sealed))
T must be point or Another class inherits from point

2_ secondary Constraint (interface Constraint [0:M] `Helper<T> where T : Icomparable<T>`
Icomparable<T> >> T must be Class or Struct Implementing IComparable
دا انما لو مقولتلوش حاجة من دول كدا هيفهم انك تقصد ان الشرط لازم يتطبق على الاستراكت او الكلاس زي المثال اللى فوق

3_ Constructor Constraint[0:1] `Helper<T> where T :class, Icomparable<T> , new()`
T must be DataType Having Accessable[Non-private] parameterless constractor
Till C# 10.0 only one constructor constraint
can't use new() [constructor constraint] with struct [special primary constraint]
--------- ---------- نقدر اننا نعمل اكتر من جنارك ونحط علي كل واحدة فيهم شرط كمان بالشكل دا
`Helper<T , T2> where T :class, Icomparable<T> , new() where T2 : struct`
وكمان ممكن احط الجنارك على مستوي الكلاس واحطلها شرط كمان بالشكل دا
`public static T BubbleSort<T>(T[] array) where T : class`

Collactions

عندنا كل الـ Data structur تندرج تحت مسمي الـ Collactions فى ال C# وعندنا شكلين وهما

1. Non Generic collaction واحنا مبنستخدمهوش بس لازم نعرفه عشان نتجنبه
2. Generic collaction

الكولكشن يعنى اى او هما اي qub, stack , linked list , list

Non Generic collaction

و بينقسم الى 2 categery وهما hashTable , listing واشهر مثال على ال non Genecic collaction هو
ال arrayList وموجودة في النيم اسبيس اللى اسمه system.collactions;
arrayList عبارة عن اراي وهي dynamic size وهي عبارة عن array of object >> مثال

```
#region Non collactions ArrayList
     ArrayList ArrayList = new ArrayList();
     Console.WriteLine($"{ArrayList.Count} , {ArrayList.Capacity}"); // 0,0
     ArrayList.Add(1);
     ArrayList.Add(2);
     ArrayList.Add(3);
     ArrayList.AddRange(new int[] {3,4});
     foreach (var item in ArrayList)
         Console.WriteLine(item); // 1 2 3 3 4

     Console.WriteLine($"{ArrayList.Count} , {ArrayList.Capacity}"); // 5,8
     ArrayList.TrimToSize();
     Console.WriteLine($"{ArrayList.Count} , {ArrayList.Capacity}"); // 5,5
#endregion
```

الاراي ليست مع انشاء اول اليمنت بيروح عامل ليست مكونة من 4 اليمنت وهي ال capacity ولو انا ضفت 5 اليمنت مثلا بيروح يضفلي 4 كمان يعنى هتبقا ب 8 طب والاماكن الزيادة الفاضية دي لو انا عايز امسحها همسحها عن طريق ميزود اسمها trimToSize() ومن مشاكل الكوليكشن بقا هي انه وانا بعمل اد ل رقم مثلا ودا int ونوعه valueType هو بيروح محوله الى object يعنى من النوع referance type وده كدا هيكون boxing فبيخزنه في الهيب ودا كدا بيأثر على ال بيرفورمنس واحنا اصلا بنتجنب ال boxing ,unboxing

```
ArrayList ArrayList1 = new ArrayList();
ArrayList.Add(1); // casting from 1[valueType] to object[referenceType]=> Boxing
ArrayList.Add("Ahmed"); // compiler can't Enforce Type safety at compilation Time بس ال clrهيزعل
    public static int sumArrayList(ArrayList ArrayList1)
```

```
        {
            int sum = 0;
            if (ArrayList1 is not null)
                for (int i = 0; i < ArrayList1.Count; i++)
                    sum += (int) ArrayList1[i];
// casting from object[referanceType] to int[ValueType] => unboxing , unsafe casting
            return sum;
        } // in program file
            int sum = sumArrayList(ArrayList1);
            Console.WriteLine(sum);
```

الحالة الوحيدة اللى هستخدم فيها ال **arrayList** لان هى **Heterogeneous list** يعنى **List** بتشيل فاليو من اكتر من داتا تايب بس وانا بستخدمها لازم اكون حذر واخد بالى من كل حاجة واكون عارف انا هضيف اي ف الليست عشان ميحصلش مشكلة عندى

Generic collaction

بينقسم الى **categery 2** وهما **listing , hashTable** وموجود داخل `using System.Collections.Generic;`

Generic collaction : listing

```
#region Generic Collaction - Lists
        List<int> lists = new List<int>();
        lists.Add(1); // index[0]
        lists.Add(2); // index[1]
        lists.AddRange(new int[] { 3, 4 }); // index[2,3]
        lists.Add(5);
        lists.TrimExcess();
        Console.WriteLine($"{lists.Count} , {lists.Capacity}"); // 5,5
        lists[0] = 100; // use indexer as setter
        Console.WriteLine(lists[0]); // use indexer as Getter
        lists[4] = 5; // index[4]   // u can't use indexer as for Adding
        #endregion
    #region list Example2
        public static int list(List<int> list)
    {
        int sum = 0;
        if (list is not null)
            for (int i = 0; i < list.Count; i++)
                sum += list[i];
        return sum;
    }
    #endregion
        int sum = sumList(lists);
        Console.WriteLine($"sum = {sum}");
```

listing - list Methods

```
        #region list Methods
        List<int> lists = new List<int>();
        lists.Insert(0,1); // (index,value);
        lists.InsertRange(1,new int[] { 2, 3}); // (index,range);
        lists.Clear(); // to clear all elements
        lists.Contains(1); // check if 1 is includ in list (if includ = true)
        lists.EnsureCapacity(20);// update Capacity
        lists.Reverse(); // to reverse list
        lists.Sort(); // to sort list
        lists.ToArray(); // to convert list to array

        foreach (var number in lists)
            Console.WriteLine(number);

        int[] Numbers = new int[] { 1, 2, 3 };
        lists.CopyTo(Numbers); // (array) copy to in index 0
        lists.CopyTo(Numbers, 1); // (array,arrayIndex)
        lists.CopyTo(1, Numbers, 1, 5); // (indexToGetIndex,array,arrayIndexToAdd,count)
```

```
foreach (var number in Numbers)
    Console.WriteLine(number);

lists.Equals(Numbers); // to compare between 2 array
```

listing - other lists

عبارة عن مجموعة من nodes بتكون linked ببعض وكل node شايلة عنوان الـnode اللى بعدها لان كل node منهم بتكون في مكان مختلف عن التانيه فى الـ heap والرابط ما بينهم هو الـ linkedList ودا شكلها في الـ heap .

سؤال مهم انترفيو طب لو عندى مجموعة من الـ values عايز اخزنهم اخزنهم فين بقا فى list, ArrayList, LinkedList , Array في الحالة دي هسأل نفسي سؤال لو الـ values دي هيتروجينين ليست (من كذا تايب مختلف) وش هستخدم الـ arrayList انما لو هما هيموجينس (يعنى متجانسين كلهم من تايب واحد) في الحالة دى لو هما fixed length هستخدم الـ Array انما لو هما dynamic length هستخدم الـ List or LinkedList طب امتا اختار مابين الاتنين دول .. لو عمليات الـ retrieve اكتر من عمليات الـ add هستخدم list انما لو عمليات الـ add اكتر هستخدم linkedList . الخلاصة امتا استخدم كل واحدة فيهم

ArrayList : لو الـ values دي هيتروجينين ليست (من كذا تايب مختلف)
Array : لو الـ values دي هيموجينس ليست (يعنى متجانسين كلهم من تايب واحد) و fixed length
List : لو الـ values دي هيموجينس ليست و dynamic length و عمليات الـ retrieve اكتر من عمليات الـ add
LinkedList : لو الـ values دي هيموجينس ليست و dynamic length و عمليات الـ add اكتر من عمليات الـ retrieve

مثال على الـ LinkedList .

```
LinkedList<int> linkedlist = new LinkedList<int>();
```

listing - Stack (first in last out)

هو شبه صندوق الكتب فهو اول كتاب حطيته فى الصندوق هو اخر كتاب هيخرج والعكس اخر كتاب هحطه ف الصندوق هو اول كتاب هيخرج

```
Stack<int> stack = new Stack<int>();
```

طيب اي ال bussines case لو بعمل لعبة بابجي وبعمل كلاس للخزنة بتاعت السلاح لان اول طلقة هتتحط فيها هي اخر طلقة هتتضرب

```
Stack<int> stack = new Stack<int>();
stack.Push(1);
stack.Push(2);
Console.WriteLine(stack.Pop()); // 2
```

listing - Queue (first in first out)

```
Queue<int> queue = new Queue<int>();
```

هو عبارة عن طابور اول حد جيه في الليست هو اول حد هيخرج

طيب اي ال bussines case لو انا بعمل سيستم لكول سنتر ف اول واحد يكلمني هو اول واحد ارد عليه .

```
Queue<int> queue = new Queue<int>();
queue.Enqueue(1);
queue.Enqueue(2);
Console.WriteLine(queue.Dequeue()); // 1
```

Generic collaction : hashTable (Dictionary)

hashTable : هو عبارة عن table بيتكون من column 2 .. column key and column value (key:value)

Dictionary : عبارة عن مجموعة من key value pair .. pair يعنى زوج ... والـ key must be unique مثال

```
Dictionary<string, long> phoneBook = new Dictionary<string, long>();
phoneBook.Add("ahmed", 01155023); // unsafe code >> because ahmed maybe entered
phoneBook.Add("ali", 0120278792); // unsafe code >> because ali maybe entered
// phoneBook.Add("ahmed", 01155023); // invalid >> key must be UNIQUE
foreach (KeyValuePair<string,long> person in phoneBook)
    Console.WriteLine($" {person.Key} , {person.Value} "); // ahmed , 01155023    ali , 0120278792
```

indexer in Dictionary : بستخدم الـ key كـindex بس اخد بالى من نوع ال key لو استرينج اكتبه استرينج بالشكل دا

```
Dictionary<string, long> phoneBook = new Dictionary<string, long>();
phoneBook.Add("ahmed", 01155023); // unsafe code >> because ahmed maybe entered
phoneBook.Add("ali", 0120278792); // unsafe code >> because ali maybe entered
```

```
// phoneBook.Add("ahmed", 01155023); // invalid >> key must be UNIQUE

//// indexer dectionary
Console.WriteLine(phoneBook["ahmed"]); // use indexer as getter // unsafe code
phoneBook["ahmed"] = 999; // use indexer as setter // unsafe code
phoneBook["mona"] /*new key*/ = 01555634; // use indexer as add // unsafe code

foreach (KeyValuePair<string, long> person in phoneBook)
     Console.WriteLine($" {person.Key} , {person.Value} "); // ahmed , 01155023    ali , 0120278792
```

: Best code for Dictionary

```
Dictionary<string, long> phoneBook = new Dictionary<string, long>();
// طريقة افضل للاضافة
phoneBook.TryAdd("medo", 888);   // safe code

//  or best code to add in dictionary // safe code
if (!phoneBook.ContainsKey("ahmed")) // check if ahmed is entered in the past or not
    phoneBook.Add("ahmed", 346365);
else
    phoneBook["ahmed"] = 1111; // update (if ahmed is contain update number)


// indexer if u need update with tryAdd
if(!phoneBook.TryAdd("medo", 888)) // return false because is not found
    phoneBook["medo"] = 5555; // update

// indexer safe code to getter
Console.WriteLine(phoneBook.TryGetValue("ahmed" , out long number));
// if ahmed not found >> return default value from (long) 0

foreach (KeyValuePair<string, long> person in phoneBook)
     Console.WriteLine($" {person.Key} , {person.Value} "); // ahmed , 01155023    ali , 0120278792
```

Session 02

هنتكلم عن ال Delegate :

هو عبارة عن feature موجودة في الـ c# فقط ومش موجودة فى لغة تانية .. وعنده استخدمين وهما

1. Pointer to function :
2. Event-driving programing :

Pointer to function

الدليجت اصلا عبارة عن كلاس فهو مش حاجة جديدة يعنى (الـcompiler بيروح يحوله ل كلاس)

```
// create class to create function inside it
  class stringFunctions
{
    public static int getCountOfUpperCase(string name)
    {
        int count = 0;
        if (name is not null)
            for (int i = 0; i < name.Length; i++)
                if (char.IsUpper(name[i]))
                    count++;
        return count;
    }
}  // in program file in name space
```

```
    // step 0 Delegate Declaration
    public delegate int stringFunctionDelegate(string str);
    // new class (Delegate), any reference from this Delegate can refer يشاور to function or more
function (pointer to function)
    // these functions may be static function (class member functions) or NonStatic (object
member functions)
    // but thses functions must be with the same signature of the Delegate (int(string))
    // regardless function name regardless access modifier
```

فى السطر دا بيعمل نيو ديليجت بيتحول لكلاس واي ريفرنس من الدليجت دا يقدر يشاور علي فانكشن او اكتر والفانكشن دي ممكن تكون استاتك او نان استاتيك والفانكشن دى لازم تكون نفس السجنيتشر بتاع الدليجيت بغض النظر عن اسمها او الاكسيس موديفاير بتاعها .

```
            // in program file in main class
             // step 1 declare referance from delegate
            stringFunctionDelegate stringFunctionDelegate;

            // step 2 initialize the referance (pointer from function)
            stringFunctionDelegate = stringFunctions.getCountOfUpperCase;

            // step 3 call method (use the delegate referance)
            Console.WriteLine(stringFunctionDelegate.Invoke("Ahmed ALi"));
```

ف انا عندى 4 خطوات بعملها عشان استخدم الدليجت وهما

1.. اعمل Declare للدليجت بتاعى بس لازم يكون نفس السجنيتشر بتاعت الفانكشن زي المثال دا كدا نوع انتجر وبياخد براميتر استرنج بغض النظر عن اسم الفانكشن او الاكسيس موديفاير بتاعها .

2.. بأخد ريفرنس من الدليجت بتاعى عشان استخدمه فى الخطوة التالتة

3.. بعمل intialize للريفرنس دا بقا

4.. بعمل call وبستخدم الدليجت بتاعى بقا

```
namespace Delegate    // نفس المثال بس من غير كومنتات وشرح
{
    // step 0
    public delegate int stringFunctionDelegate(string str);
    class Program
    {
        static void Main(string[] args)
        {
            stringFunctionDelegate stringFunctionDelegate;
            stringFunctionDelegate = stringFunctions.getCountOfUpperCase;
            Console.WriteLine(stringFunctionDelegate.Invoke("Ahmed ALi"));
        }
    }
```

Delegate Example 02

```
    class pobuleStart // create class pobule sort ex
    {
        // sort Ascending >
        public static void PubleSort(int[] array)
        {

            if (array is not null)
                for (int i = 0; i < array.Length; i++)
                    for (int j = 0; j < array.Length - i - 1; j++)
                        if (array[j] > array[j+1])
                            swap(ref array[j + 1], ref array[j]);
        }
        // sort Descending >
        public static void PubleSortDesc(int[] array)
```

```
        {

            if (array is not null)
                for (int i = 0; i < array.Length; i++)
                    for (int j = 0; j < array.Length - i - 1; j++)
                        if (array[j] < array[j + 1])
                            swap(ref array[j + 1], ref array[j]);
        }
        // SWAP meghod
        public static void swap(ref int x ,ref int y)
        {
            int temp = x;
            x = y;
            y = temp;
        }
    }   // in program file
            int[] numbers = { 3, 2, 4, 1, 6, 4, 7, 3 }; // ctrate aray

            pobuleStart.funcPubleSort(numbers); // to call funcPubleSort

            foreach (var number in numbers)
                Console.WriteLine(number); // 1,2,3,3,4,6,7
    }
```

طيب انا كدا بكرر الكود بس الفرق ف انى بغير علامة الاكبر من والاصغر فهعمل كلاس وجوا **2** فانكشن تكومبير من عشان اسورت وهعمل دليجت عشان الموضوع يكون دينامك

```
    public delegate bool compareFuncDelegate(int a, int b);
    class pobuleStart
    {
        // sorting
        public static void funcPubleSort(int[] array , compareFuncDelegate compareFuncDelegate)
        {

            if (array is not null)
                for (int i = 0; i < array.Length; i++)
                    for (int j = 0; j < array.Length - i - 1; j++)
                        //if (array[j] > array[j+1])
                        //if(compaireFunctions.funcGreterThan(array[j], array[j + 1]))
                        if (compareFuncDelegate.Invoke(array[j], array[j + 1]))
                            swap(ref array[j + 1], ref array[j]);
        } // SWAP meghod
        public static void swap(ref int x ,ref int y)   {   int temp = x;  x = y; y = temp;  }
    }

    class compaireFunctions
    {
        public static bool funcGreterThan(int x, int y) { return x > y; } // sort Ascending >
        public static bool funcLessThan(int x, int y) { return x < y; }  // sort Descending >

    }   /// in program file
            int[] numbers = { 3, 2, 4, 1, 6, 4, 7, 3 };

            pobuleStart.funcPubleSort(numbers , compaireFunctions.funcGreterThan);
            Console.WriteLine("----------------------------");
            pobuleStart.funcPubleSort(numbers, compaireFunctions.funcLessThan);

            foreach (var number in numbers)
                Console.WriteLine(number);
```

طب انا عايز اخلي الكود **more dynamic** شوية وادخل في الGenaric وانى اقدر اعمل **sort** على اي تايب سواء **array** او **string** او...

```
    public delegate bool compareFuncDelegate<T1>(T1 a, T1 b);
    class pobuleStart<T>
    {
```

```
        // sorting by Genaric
        public static void funcPubleSort(T[] array, compareFuncDelegate<T> compareFuncDelegate)
        {

            if (array is not null)
                for (int i = 0; i < array.Length; i++)
                    for (int j = 0; j < array.Length - i - 1; j++)
                        if (compareFuncDelegate.Invoke(array[j], array[j + 1]))
                            swap(ref array[j + 1], ref array[j]);
        }
        public static void swap(ref T x, ref T y) { T temp = x; x = y; y = temp; }
    }

    class compaireFunctions
    {
        public static bool sortSTRGreterThan(string x, string y) { return x?.Length > y?.Length;
} // sort Ascending >
        public static bool sortSTRLessThan(string x, string y) { return x?.Length > y?.Length; }
// sort Descending >
        public static bool funcGreterThan(int x, int y) { return x > y; } // sort Ascending >
        public static bool funcLessThan(int x, int y) { return x < y; }  // sort Descending >

    } // in program file

            int[] numbers = { 3, 2, 4, 1, 6, 4, 7, 3 };
            // sort_int
            pobuleStart<int>.funcPubleSort(numbers , compaireFunctions.funcGreterThan);

            foreach (var number in numbers)
                Console.WriteLine(number);

            Console.WriteLine("----------sort_String----------");

            string[] names = { "ahmed", "mai", "omar", "ali", "mahomoud", "yasser" };
            pobuleStart<string>.funcPubleSort(names, compaireFunctions.sortSTRGreterThan);

            foreach (var name in names)
                Console.WriteLine(name);
```

Example 03

```
In program file
List<int> numbers = Enumerable.Range(0, 100).ToList(); // == 0:99 // (1,100)== 1:100
```

لو انا عايز اطبع الارقام الفردية او الزوجية من الليست دي هحتاج انى اعمل فانكشن تعملي دا بشكل ديناميك فهستخدم الدليجت والجنارك اول حاجة هعمل فانكشن

```
        public static List<T> getNumbersBasedOnfunc<T>(List<T> numbers , delegetFuncOddOREven<T>
delegetFuncOddOREven)
        {
            List<T> result = new List<T>();

            if (numbers is not null)
                for (int i = 0; i < numbers.Count; i++)
                    if (delegetFuncOddOREven.Invoke(numbers[i]))
                        result.Add(numbers[i]);
            return result;
        } // create class to insert inside it 2 function to check Numbers
    class CheckoddOREvenNumbers
    {
        public static bool funOddNumbers(int number) { return number % 2 == 1 ; }

        public static bool funEvenNumbers(int number) { return number % 2 == 0 ; }
    }
// create delegate to make function more dinamic (add delegate in namespace for the class)
```

```
public delegate bool delegetFuncOddOREven<in T1>(T1 number);

// in the program file to run code
            List<int> EvenNumbers = getNumbersBasedOnfunc<int>(numbers ,
CheckoddOREvenNumbers.funEvenNumbers);
            foreach (int number in EvenNumbers)
                Console.WriteLine(number);
```

اللى فوق دا شرح الكود انما نضيف الكود بالترتيب بدون شرح عشان يكون واضح

```
    public delegate bool delegetFuncOddOREven<in T1>(T1 number);
    class Program
    {
        public static List<T> getNumbersBasedOnfunc<T>(List<T> numbers , delegetFuncOddOREven<T>
delegetFuncOddOREven)
        {
            List<T> result = new List<T>();

            if (numbers is not null)
                for (int i = 0; i < numbers.Count; i++)
                    if (delegetFuncOddOREven.Invoke(numbers[i]))
                        result.Add(numbers[i]);
            return result;
        }
        static void Main(string[] args)
        {
            List<int> numbers = Enumerable.Range(0, 100).ToList(); // == 0:99 // (1,100)== 1:100

            List<int> EvenNumbers = getNumbersBasedOnfunc<int>(numbers ,
oddOREvenNumbers.funEvenNumbers);
            foreach (int number in EvenNumbers)
                Console.WriteLine(number);
        }
    }
    class oddOREvenNumbers
    {
        public static bool funOddNumbers(int number) { return number % 2 == 1 ; }

        public static bool funEvenNumbers(int number) { return number % 2 == 0 ; }
    }
```

Built-in Delegates

عندنا فى الـ .Net يوجد 3 **builtInDelegate** جاهزين بنستخدمهم ع طول بدل ما نروح نعرف **Delegate** وهما

1.. **Predicate** بيشاور على فانكشن بتاخد مني **Pramater** من اي نوع ولازم الفانكشن دي ترجعلى **boolen** ولو جينا نطبق على المثال اللى فوق دا هنمسح الدلجيت الى انا عامله وهعوض عن اي حاجة ف الكود بكلمة **predicate** زي كدا بالظبط

```
    class Program
    {
        public static List<T> getNumbersBasedOnfunc<T>(List<T> numbers , Predicate<T>
delegetFuncOddOREven)
        {
            List<T> result = new List<T>();

            if (numbers is not null)
                for (int i = 0; i < numbers.Count; i++)
                    if (delegetFuncOddOREven.Invoke(numbers[i]))
                        result.Add(numbers[i]);
            return result;
        }
        static void Main(string[] args)
        {
            List<int> numbers = Enumerable.Range(0, 100).ToList(); // == 0:99 // (1,100)== 1:100
```

```
            List<int> EvenNumbers = getNumbersBasedOnfunc<int>(numbers ,
oddOREvenNumbers.funEvenNumbers);
            foreach (int number in EvenNumbers)
                Console.WriteLine(number);
        }
    }
    class oddOREvenNumbers
    {
        public static bool funOddNumbers(int number) { return number % 2 == 1 ; }

        public static bool funEvenNumbers(int number) { return number % 2 == 0 ; }
    }
```

2.. Func بيشاور على فانكشن بتأخد pramater من 0 ل 16 ولازم يكون عندها return فنا كدا عندى منها 17 نسخة .. نسخة بتشاور على فانكش وبتاخد P واحد وبترتيرن او بتشاور على فانكشن وبتاخد2 p وبترتيرن او بتشاور على فانكشن بتاخد3برااميتر وبترتيرن وهكذا لحد ماتوصل للبراميتر رقم 17 .. نفس الكلام هنمسح الدليجت اللى كنا عاملينه وهنعوض عنه بكلمة func في الكود بتاعى مثال

```
    class pobuleStart<T>
    {
        // sorting by Genaric
        public static void funcPubleSort(T[] array, Func<T> compareFuncDelegate)
        {

            if (array is not null)
                for (int i = 0; i < array.Length; i++)
                    for (int j = 0; j < array.Length - i - 1; j++)
                        if (compareFuncDelegate.Invoke(array[j], array[j + 1]))
                            swap(ref array[j + 1], ref array[j]);
        }
        public static void swap(ref T x, ref T y) { T temp = x; x = y; y = temp; }
    }

    class compaireFunctions
    {
        public static bool sortSTRGreterThan(string x, string y) { return x?.Length > y?.Length;
} // sort Ascending >
        public static bool sortSTRLessThan(string x, string y) { return x?.Length > y?.Length; }
// sort Descending >
        public static bool funcGreterThan(int x, int y) { return x > y; } // sort Ascending >
        public static bool funcLessThan(int x, int y) { return x < y; }  // sort Descending >

    } // in program file

            int[] numbers = { 3, 2, 4, 1, 6, 4, 7, 3 };
            // sort_int
            pobuleStart<int>.funcPubleSort(numbers , compaireFunctions.funcGreterThan);

            foreach (var number in numbers)
                Console.WriteLine(number);

            Console.WriteLine("----------sort_String----------");

            string[] names = { "ahmed", "mai", "omar", "ali", "mahomoud", "yasser" };
            pobuleStart<string>.funcPubleSort(names, compaireFunctions.sortSTRGreterThan);

            foreach (var name in names)
                Console.WriteLine(name);
```

3.. Action بيشاور على فانكشن بتأخد pramater من 0 ل 16 ولازم يكون void مش بيرتيرن حاجة .

■ طب امتا اعمل delegate ب ايدي لو انا عايز اعمل فانكشن بتاخد مني اكتر من 17 برامتر هضطر اني اعمل دليجت بنفسي بس دائما مش بنعمل اكتر من 16 براميتر يعنى ودائما هنستخدم ال builtInDelegate ومش هعمله ب ايدي .

--- امثلة سريعة للتأكيد .. هنعمل كلاس في 4 فانكشن هنعمل كول لكل واحدة فيهم عن طريق ال builtInDelegate بالشكل دا

```
class TestBuiltInDelegate
{
    public static bool funPredicate(int number) { return number > 0 ; }
    public static string funFunc(int x) { return x.ToString(); }
    public static void funAction() { Console.WriteLine("hi From ActionFunc"); }
    public static void funAction2(string name) { Console.WriteLine($"Hello {name}"); }
} /// in program file
```

```
Predicate<int> predicate = TestBuiltInDelegate.funPredicate;
predicate.Invoke(10); // to run or u can use syntax sugar predicate(10);
predicate(10); // Syntax sugar to run  // true

Func<int , string> Func  = TestBuiltInDelegate.funFunc; // take int , return string
Func(20);
// Action have 2 version
Action action = TestBuiltInDelegate.funAction; // Action version 1
action(); // hi From ActionFunc
Action<string> action2 = TestBuiltInDelegate.funAction2; // Action version 2
action2("Ahmed"); // Hello Ahmed
```

طيب انا فوق عملت كلاس عشان احط في فانكشنز والفانكشنز دى انا هستخدمها مرة واحدة فى حياتى وانا عندى امكانية ان الـ c# هي اللى تـ implement فكرة الفانكشن بروجرامينج عن طريق حاجة اسمها Anonymous Function/Method بالشكل دا

Anonymous Function/Method c# 2.0

يعنى فانكشن ملهاش اسم هناديها مرة واحدة فقط في حياتى ف مش محتاج انى اديها اسم .

```
Predicate<int> predicate = delegate (int number) { return number > 0; };
predicate(10);

Func<int , string> Func  = delegate (int x) { return x.ToString(); };
Func(20);

Action action = delegate () { Console.WriteLine("hi From ActionFunc"); };
action();
Action<string> action2 = delegate (string name) { Console.WriteLine($"{name}"); };
action2("Ahmed");
```

Lambada Expression c# 3.0

زي ال arrow function في الjavaScript بس هنا اسمها Lambada ...

read as Goes to => و اسمه called as fatArrow

تقدر تمسح كلمة دليجت وتقدر متحددش نوع البراميتر ولو الفانكشن بتاعتك سطر واحد تقدر تمسح الاقواس والريتيرن ولو بياخد براميتر واحد همسح الاقواس برضو

```
Predicate<int> predicate = number => number > 0;
predicate(10);

Func<int, string> Func = x => { return x.ToString(); };
Func(20);

Action action = () => { Console.WriteLine("hi From ActionFunc"); };
action();
Action<string> action2 = name => { Console.WriteLine($"Hello {name}"); };
action2("Ahmed");
```

New Feature New Update for Delegate : C# 10.0 (.Net 6 2021)

بقيت اقدر بدل ما اكتب **built in delegate** بقيت اقدر اكتب كلمة **var** وهي فى الاخر اصلا بتتحول للكود الاساسي بتاعها .. ودي قوة للي سي شارب ف ان هي تقدر تتلون بالالوان الاخري (اللغات الاخري) .

```
var predicate = (int number) => number > 0;
predicate(10);

var Func = (int x) => x.ToString();
Func(20);

var action = () => { Console.WriteLine("hi From ActionFunc"); };
action();
var action2 = (string name) => { Console.WriteLine($"Hello {name}"); };
action2("Ahmed");
```

Some List_Methods That Take Functions As Parameters

فانا ممكن ابعت الفانكشن كبراميتر لفانكشن تانيه بطريقة الـ **anonymose** او **lambada** بالشكل دا المثال رقم **3** اللى فوق هناخد الجزء الموجود فى البروجرم ونعدل عليه .. دي اسمها فى الجافا سكريبت **callback function**

```
 List<int> numbers = Enumerable.Range(0, 100).ToList();
// by anonymose method
List<int> EvenNumbers = getNumbersBasedOnfunc<int>(numbers, delegate (int number) {
return number % 2 == 1; });
// by Lambada method
List<int> EvenNumbers2 = getNumbersBasedOnfunc<int>(numbers, number => number % 2 == 1);

    foreach (int number in EvenNumbers)
        Console.WriteLine(number);
```

وكمان احنا عندنا فانكشن ممكن تعملي دا اسمها **findAll** وهي بتستلم مني اكسبريشن ولو اتحقق بتجبهولى

```
List<int> numbers = Enumerable.Range(0, 100).ToList();
List<int> EvenNumbersFindAll = numbers.FindAll( number => number % 2 == 1 );
foreach (int number in EvenNumbersFindAll)
    Console.WriteLine(number);
```

اما **find** بترجعلي اول رقم فقط اللى حققلي الاكسبريشن اللى انا مدهوله بالشكل دا **// findLast** عكسها

```
int EvenNumbersFind = numbers.Find(number => number % 2 == 1);
Console.WriteLine(EvenNumbersFind);
```

اما **findIndex** بترجعلي اول رقم فقط داخل رينج انا بحددهله من الليسته وفقا للاكسبريشن اللى انا مدهوله بالشكل دا

```
int EvenNumbersFindINdex = numbers.FindIndex(5 , number => number % 2 == 1);
Console.WriteLine(EvenNumbersFind);
```

اما **findLastIndex** عكس اللى فوق بتسيرش من الاخر

اما **forEach** شبه اللى ف الجافااسكريبت وفي المثال دا بيروح يعمل لوب على الليست وبيستلم كل رقم في الـ**num** وبيزود عليه **10**

```
List<int> numbers = Enumerable.Range(0, 100).ToList();
numbers.ForEach((num => { num += 10; }));
foreach (int number in numbers)
    Console.WriteLine(number);
```

اما **RemoveAll** بتحذف وفقا للاكسبريشن اللى انا حاطه