## 1)Struct: value type

في حالة المقارنة بين 2 structs بيقارن value بتاعة الاتنين نفس الكلام لو عملت copy من واحد للتاني وغيرت قيمة التاني الأول مش هيتغير وكمان في الاغلب immutable بس عيوبه انو مينفعش تستخدم فيه inheritance او polymorphism وافضل استخدام ليه مع small data الي محتاجة الخصائص الي فاتوا

## 2)Class: reference type

في حالة المقارنة بين 2 classes بيقارن reference بتاع الاتنين نفس الكلام لو عملت copy من واحد للتاني هيبقي الاتنين بيشاوروا علي نفس object يعني أي تغير في التاني او الأول بيغير نفس object في الاغلب mutable وبيستخدم مع حاجات اكتر تعقيد ومحتاجة الخصائص الي فاتت

## 3)Record: reference type

بالرغم من انه reference type الا انه بيقارن value بيتاعه الاتنين نفس الكلام لو عملت copy من واحد للتاني وغيرت قيمة التاني الأول مش هيتغير وكمان هو immutable وبيقدر بيستخدم خصائص OOP زي inheritance او polymorphism في حالة record class وبيستخدم مع حاجات معقدة محتاجة الخصائص الي فاتت

```
namespace Struct;

5 references
struct Point(int x, int y) //immutable struct
{
    3 references
    public int X { get; } = x;
    3 references
    public int Y { get; } = y;

    1 reference
    public Point Move(int dx, int dy) => new Point(X + dx, Y + dy);
}
0 references
internal class Program
{
    0 references
    static void Main(string[] args)
    {
        Point p1 = new Point(10, 20);
        Point p2 = p1;
        p2 = p2.Move(5, 5);

        Console.WriteLine($"{p1.X}  {p1.Y}");
        Console.WriteLine($"{p2.X}  {p2.Y}");
    }
}
```

```
Microsoft Visual Studio Debug C
10  20
15  25

D:\courses\.net backend e
Press any key to close th
```

```
namespace Class
{
    3 references
    class BankAccount(string accountNumber, decimal balance)
    {
        0 references
        public string AccountNumber { get; } = accountNumber;
        3 references
        public decimal Balance { get; private set; } = balance;
        1 reference
        public void Deposit(decimal amount) => Balance += amount;
    }
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            BankAccount account1 = new BankAccount("123", 100m);
            BankAccount account2 = account1;
            account2.Deposit(20m);

            Console.WriteLine(account1.Balance);
            Console.WriteLine(account2.Balance);
        }
    }
}
```

```
Microsoft Visual Studio De
120
120

D:\courses\.net backe
Press any key to clos
```

```
namespace Record
{
    5 references
    public record User(string Name, int Age); //immutable
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            User u1 = new User("mohamed", 25);
            User u2 = new User("mohamed", 25);
            User u3 =u1 with { Age = 30 };
            Console.WriteLine(u1 == u2);
            Console.WriteLine(u1 == u3);
        }
    }
}
```

```
Microsoft Visual Studio Debug C
True
False

D:\courses\.net backend e
Press any key to close th
```