Self-Study Report:
what is casting opertator,
null propagation operator as readability, Jagged array
jump table ,
conditioning on bool , numeric comparison

Casting Operator بيستخدم لتحويل نوع بيانات معين إلى نوع بيانات تاني .مثلاً، لو عندك رقم صحيح (Integer)وعايز تحوله إلى رقم عشري (Double)، بتستخدم ال casting operator

فيه نوعين من ال casting

Implicit Casting ده بيحصل تلقائي لما بتحول من نوع بيانات أصغر لنوع بيانات أكبر زي من int ل double

Explicit Casting
ده بيحتاج تكتب ال casting operator بأيدك، خصوصًا لما بتحول من نوع بيانات أكبر لنوع بيانات أصغر زي من double إلى int

لما بنيجي نشتغل على كود في السي شارب، فممكن نحتاج نتعامل مع قيم ممكن تكون null عشان نحسن قراءة الكود بتاعنا ونتجنب الأخطاء، بنستخدم الـ Null اللي هو Propagation Operator

مثال:

تخيل إن عندك كائن (object) لو عايز تطبع قيمة معينة جواه، لكن ممكن الكائن ده يكون null، لو حاولت تتعامل معاه بشكل عادي وطلع null، البرنامج هيرمي خطأ (Exception) لكن باستخدام الـ Null Propagation Operator، الكود هيشتغل بأمان حتى لو الكائن null

```
class Person
{
  public string Name { get; set; }
}

class Program
{
  static void Main()
  {
    Person person = null;

    // Without Null Propagation Operator
    if (person != null)
    {
      Console.WriteLine(person.Name);
    }

    // With Null Propagation Operator
    Console.WriteLine(person?.Name);
  }
}
```

```
using System;

class Program
{
  static void Main()
  {
    // Define a jagged array with 3 elements
    int[][] jaggedArray = new int[3][];

    // Initialize the sub-arrays with different lengths
    jaggedArray[0] = new int[2];
    jaggedArray[1] = new int[3];
    jaggedArray[2] = new int[4];

    // Fill the sub-arrays with values
    for (int i = 0; i < jaggedArray.Length; i++)
    {
      for (int j = 0; j < jaggedArray[i].Length; j++)
      {
        jaggedArray[i][j] = i + j;
      }
    }

    // Display the values
    for (int i = 0; i < jaggedArray.Length; i++)
    {
      Console.Write("Row " + i + ": ");
      for (int j = 0; j < jaggedArray[i].Length; j++)
      {
        Console.Write(jaggedArray[i][j] + " ");
      }
      Console.WriteLine();
    }
  }
}
```

Jagged Array

ال Jagged Arrayهو مصفوفة (Array)من المصفوفات، يعني كل عنصر فيها بيحتوي على مصفوفة فرعية ممكن تكون بأطوال مختلفة .بمعنى آخر، مش زي الـ 2D Arrayاللي بيبقى فيها كل الصفوف بنفس الطول، الـ Jagged Arrayبيديك حرية إن كل صف يكون بطول مختلف.

Note:

Define: We create a jagged array with 3 elements. Each element is an array itself.

Initialize: We set different lengths for each sub-array.

Fill: We use loops to fill each sub-array with values.

Display: We print out the values of each sub-array.

ممكن نستخدم المصفوفات من الدوال أو تفويضات Delegates لعمل Jump Table دي طريقة لتحسين سرعة التنفيذ في جمل switch-case، خاصة لما يكون عندنا عدد كبير من الحالات

```
using System;

class Program
{
  // تعريف بعض الدوال للحالات المختلفة
  static void Case0() { Console.WriteLine("Case 0"); }
  static void Case1() { Console.WriteLine("Case 1"); }
  static void Case2() { Console.WriteLine("Case 2"); }

  static void Main()
  {
    // تعريف مصفوفة من التفويضات (Delegates)
    Action[] jumpTable = { Case0, Case1, Case2 };

    Console.Write("Enter a number (0-2): ");
    if (int.TryParse(Console.ReadLine(), out int value) && value >= 0 &&
value < jumpTable.Length)
    {
      // استخدام الـ Jump Table لاستدعاء الدالة المناسبة
      jumpTable[value]();
    }
    else
    {
      Console.WriteLine("Invalid input");
    }
  }
}
```

كدة هنا انا عرفت ثلاث دوال، كل دالة بتمثل حالة معينة.

عملت مصفوفة من نوع Action، وكل عنصر فيها بيشاور لدالة معينة.

استخدمت القيمة المدخلة من المستخدم لتحديد واستدعاء الدالة المناسبة باستخدام الـ Jump Table

الطريقة دي بتخلي الكود cleanوأسهل في القراءة، وكمان بتحسن الأداء لما يكون عندنا حالات كتير

التعامل مع الشرط في Boolean و Numeric:

Numeric لو عندك متغير رقمي وعايز تعمل مقارنة بين قيم

```
int number = 10;

if (number > 5)
{
   Console.WriteLine("The number is greater than 5!");
}
else if (number == 5)
{
   Console.WriteLine("The number is 5!");
}
else
{
   Console.WriteLine("The number is less than 5!");
}
```

التعامل مع الشرط في Boolean و Numeric:

Booleanلو عندك متغير booleanو عايز تتأكد من قيمته، بتعمل كده

```
bool isTrue = true;
if (isTrue)
{
   Console.WriteLine("The value is true!");
}
else
{
   Console.WriteLine("The value is false!");
}
```

مقارنة بين القيم:
لو القيمة أكبر من حاجة معينة> :

لو القيمة أقل من حاجة معينة< :

لو القيمة مساوية لحاجة معينة== :

لو القيمة أكبر أو تساوي حاجة معينة=< :

لو القيمة أقل أو تساوي حاجة معينة=> :