Турсунов Баходурхон

**1. Как выражение "is expression" поддерживает "Pattern Matching" в C#, и какова его роль?**

Is – выражение используется для проверки типа объекта во время выполнения:

If (variable **is** **int** intValue) {  
// выполняется, если переменная имеет тип Int, и значение будет доступно как intValue  
}

Switch выражение с pattern matching:

Switch стал более мощным с использованием pattern matching.

**object** obj = 42;  
  
**switch** (obj)  
{  
 **case** **int** intValue:  
 Console.WriteLine("Это целое число: " + intValue);  
 **break**;  
 **case** **string** stringValue:  
 Console.WriteLine("Это строка: " + stringValue);  
 **break**;  
 **default**:  
 Console.WriteLine("Неизвестный тип");  
 **break**;  
}

Типы-паттерны:

Позволяют более удобно фильтровать объекты в switch по их типу

**switch** (obj)  
{  
 **case** **int** \_:  
 Console.WriteLine("Это целое число");  
 **break**;  
 **case** **string** \_:  
 Console.WriteLine("Это строка");  
 **break**;  
 **default**:  
 Console.WriteLine("Неизвестный тип");  
 **break**;  
}

Деконструкция:

Позволяет разбивать объекты на составляющие части (например, свойства) прямо внутри pattern matching

**if** (obj **is** Point { X: **var** x, Y: **var** y })  
{  
 Console.WriteLine($"Точка на плоскости: X={x}, Y={y}");  
}

Сочетание с when-выражением:

When позволяет добавлять дополнительные условия для выполнения блока кода в case:

**switch** (obj)  
{  
 **case** **int** intValue when intValue > 0:  
 Console.WriteLine("Положительное целое число");  
 **break**;  
 **case** **int** intValue when intValue < 0:  
 Console.WriteLine("Отрицательное целое число");  
 **break**;  
 **case** **int** \_:  
 Console.WriteLine("Нулевое целое число");  
 **break**;  
 **default**:  
 Console.WriteLine("Не целое число");  
 **break**;  
}

Эти концепции позволяют более гибко и элегантно обрабатывать различные случаи в коде, основываясь на типах и структурах данных.

**2. Как мы можем использовать "Pattern Matching", чтобы убедиться, что значения**

**не равны null в C#? Приведите пример.**

Pattern matching также может использоваться для проверки на null. С помощью ключевого слова is и шаблона null мы можем удостовериться, что объект не является Null:

**object** someObject = "Hello world!";  
  
**if** (someObject **is** not null)  
{  
 // Блок кода, который выполняется, если someObject не равно null.  
 Console.WriteLine($"Объект не равен null: {someObject}");  
}  
**else**  
{  
 // Блок кода, который выполняется, если someObject равно null.  
 Console.WriteLine("Объект равен null");  
}

**3. Как можно использовать "Pattern Matching" для проверки переменной на соответствие определенным значениям, например значениям в перечислении?**

**enum** DayOfWeek  
{  
 Monday,  
 Tuesday,  
 Wednesday,  
 Thursday,  
 Friday,  
 Saturday,  
 Sunday  
}  
  
DayOfWeek today = DayOfWeek.Wednesday;  
  
**switch** (today)  
{  
 **case** DayOfWeek.Monday:  
 Console.WriteLine("Понедельник");  
 **break**;  
 **case** DayOfWeek.Tuesday:  
 Console.WriteLine("Вторник");  
 **break**;  
 **case** DayOfWeek.Wednesday:  
 Console.WriteLine("Среда");  
 **break**;  
 **case** DayOfWeek.Thursday:  
 Console.WriteLine("Четверг");  
 **break**;  
 **case** DayOfWeek.Friday:  
 Console.WriteLine("Пятница");  
 **break**;  
 **case** DayOfWeek.Saturday:  
 Console.WriteLine("Суббота");  
 **break**;  
 **case** DayOfWeek.Sunday:  
 Console.WriteLine("Воскресенье");  
 **break**;  
 **default**:  
 Console.WriteLine("Неизвестный день недели");  
 **break**;  
}

В данном примере pattern matching используется в switch для сопоставления значения today с каждым элементом перечисления DayOfWeek. Каждый case представляет одно из значений перечисления, и блок кода выполняется при соответствии.

**4. Как в C# можно написать шаблоны, анализирующие несколько свойств объекта?**

Мы можем использовать шаблоны для анализа нескольких свойств объекта с использованием так называемых Property Patterns. Эти шаблоны позволяют нам проверять значения нескольких свойств объекта внутри выражения:

**public** **class** **Point**  
{  
 **public** **int** X { **get**; **set**; }  
 **public** **int** Y { **get**; **set**; }  
}  
  
Point point = **new** Point { X = 5, Y = 10 };  
  
**if** (point **is** { X: 5, Y: **var** yValue })  
{  
 Console.WriteLine($"Объект имеет X=5 и Y={yValue}");  
}  
**else** **if** (point **is** { X: **var** xValue, Y: **var** yValue })  
{  
 Console.WriteLine($"Объект имеет X={xValue} и Y={yValue}");  
}  
**else**  
{  
 Console.WriteLine("Объект не соответствует шаблону");  
}

**5. Почему "Сопоставление шаблонов" считается ценным инструментом при работе с данными, которые не имеют регулярной структуры в C#?**

Сопоставление шаблонов в C# является мощным инструментом для обработки данных, не имеющих регулярной структуры. Во-первых, он предоставляет гибкость в обработке разнообразных типов данных, учитывая не только их тип, но и структуру, свойства, значения и другие характеристики.

Practice:

1. Напишите функцию на C#, которая принимает на вход объект и с помощью "Pattern Matching" определяет, является ли он строкой. Если это строка, выведите ее длину, если нет, выведите сообщение "Это не строка".

**namespace** **Practice\_HW21**  
{  
  
 **class** **Program**  
 {  
 **static** **void** **Main**(**string**[] args)  
 {  
  
 **object** userInput1 = "Hello";  
 **object** userInput2 = 2;   
 PatternMatching(userInput1);  
 PatternMatching(userInput2);  
 }  
  
 **static** **void** **PatternMatching**(**object** obj)  
 {  
 **if** (obj **is** **string** str)  
 Console.WriteLine($"{str.LengthВ}");  
 **else**  
 Console.WriteLine("Это не строка");  
 }  
 }  
}

2. Разработайте программу, которая принимает на вход температуру в градусах по Фаренгейту и, используя "Подбор шаблонов", определяет состояние воды (пар, жидкость, лед) при этой температуре. Для определения состояния используйте следующие значения:

Температура менее 32°F - вода находится в состоянии льда.

Температура от 32°F до 212°F - вода в жидком состоянии.

Температура выше 212°F - вода находится в парообразном состоянии.