Typ wyliczeniowy (zwane również enumeration lub enum) jest to typ danych składający się z zestawu nazwanych wartości. Do zmiennej, która została zadeklarowana jako typ wyliczeniowy może być przypisane wyliczenie jako wartość. Wyliczenia uczynią kod bardziej czytelnym.

```
using System;

class CSharpApp
{
    enum Days
    {
        Monday,
        Tuesday,
        Wednesday,
        Thursday,
        Friday,
        Saturday,
        Sunday
    }

    static void Main()
    {
        Days dni = Days.Monday;

        if (dni == Days.Monday)
        {
            Console.WriteLine("To jest poniedziałek");
        }
        Console.WriteLine(dni);
    }
}
```

W powyższym kodzie utworzono wyliczenie dni tygodnia.

```
enum Days
{
    Monday,
    Tuesday,
    Wednesday,
    Thursday,
    Friday,
    Saturday,
    Sunday
}
```

Typ wyliczeniowy jest tworzony ze słowem kluczowym **enum**. Poniedziałek, wtorek ... Dni tygodnia są numerowane 0 .. 6.

```
Days dni = Days.Monday;
```

Zmienna **dni** jest zadeklarowana jako typ **Days**. Jest ona inicjowana na poniedziałek.

```
if (dni == Days.Monday)
{
    Console.WriteLine("To jest poniedziałek");
}
```

Kod jest bardziej czytelny niż w przypadku porównywania zmiennej dni do jakiegoś numeru.

```
Console.WriteLine(dni);
```

Powyższa linia wyświetla słowo poniedziałek na konsoli.

Domyślnym typem podstawowym każdego elementu w elemencie enum jest int. Za pomocą dwukropka, można określić innego typu liczbowego integralną.

```
using System;

class CSharpApp
{
    public enum Sezon : byte
    {
        Wiosna = 1,
        Lato = 2,
        Jesien = 3,
        Zima = 4
    }

    static void Main()
    {
        Sezon s1 = Sezon.Wiosna;
        Sezon s2 = Sezon.Jesien;

        Console.WriteLine(s1);
        Console.WriteLine(s2);
    }
}
```

Pory roku mogą być łatwo wykorzystane jako teksty stałe. Możemy określić podstawowy typ dla wyliczenia oraz możemy podać dokładne wartości dla nich.

```
public enum Sezon : byte
    {
        Wiosna = 1,
        Lato = 2,
        Jesien = 3,
        Zima = 4
    }
```

Za pomocą dwukropka i typu danych określamy typ podstawowy dla **enum**. Nadajemy również każdemu elementowi określoną liczbę.

```
Console.WriteLine(s1);
Console.WriteLine(s2);
```

Te dwie linie wydrukują wartości **enum** na konsoli.

Podstawowe wartości numerycznych można sprawdzać przy rzutowanie typu podstawowego.

```
Sezon tenSezon = Sezon.Wiosna;
byte SezonLiczba = (byte) tenSezon;
```

Tablice

```
int[] array = new int[5];
int[] array2 = { 1, 3, 5, 7, 9 };
```

Tablice wielowymiarowe

```
int[,] array = new int[4, 2];
int[,] array2D = new int[,] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
```

Zadanie 1

1. Utwórz nowy projekt dla aplikacji konsolowej.
2. Dodaj typ wyliczeniowy **ruch**, z trzema możliwymi wartościami:
    - kolko,
    - krzyzyk,
    - brak.

    Ostatnia z wartości oznacza, że ruch nie został jeszcze wykonany. Typ wyliczeniowy może być dodany w obrębie klasy z metodą **main**, w innej klasie lub poza klasą. Od tego, w którym miejscu zostanie on dodany zależy reszta kodu programu.

3. Dodaj typ wyliczeniowy **gracz**, określający, kto wykonuje ruch (kółko, czy krzyżyk).
4. Dodaj typ wyliczeniowy **rezultatGry** o następujących możliwych wartościach: **remis**, **nierozstrzygnieta**, **wygrywaKolko**, **wygrywaKrzyzyk**.

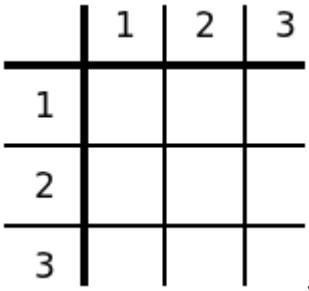
Zadanie 2

Utwórz klasę o nazwie **plansza**. Klasa ta zawiera 9cio elementową jednowymiarową tablicę obiektów **ruch**. Dodatkowo klasa ta zawiera następujące metody:

- **bool zaznaczRuch(int x, int y, ruch znak)**
- **rezultatGry sprawdzRezultat()**
- **string ToString()**
- **static gracz zmienGracza(gracz aktualnyGracz)**

Zadanie 3

Napisz metodę **bool zaznaczRuch(int x, int y, ruch znak).** Metoda zaznacz ruch, zaznacza na podstawie współrzędnych **x** wiersza oraz **y** kolumny. Przyjmijmy, że numeracja wierszy i kolumn rozpoczyna się od 1.



Jeśli pod podanym polem został już zaznaczony ruch, to metoda zwraca wartość **false**, w przeciwnym przypadku zwraca wartość **true**.

Zadanie 4

Napisz metodę **string ToString().** Metoda **ToString** zwraca w postaci tekstowej stan szachownicy, np w taki sposób:

X . O
. X O
O . X

Metoda będzie służyła do wyświetlania stanu tablicy. Za każdym razem kiedy chcemy wyświetlić planszę po prostu wykonujemy metodę **ToString()** na obiekcie klasy **plansza**.
W dalszej części instrukcji zostaną dokładnie opisane metody, jakie należy zaimplementować.

Zadanie 5

Napisz metodę **rezultatGry sprawdzRezultat()**, która określa rezultat gry. Przyjmijmy, że poszczególne elementy tablicy typów wyliczeniowych **ruch** oznaczają następujące pola szachownicy:

0	1	2
3	4	5
6	7	8

Sprawdzanie rezultatu gry polega na:

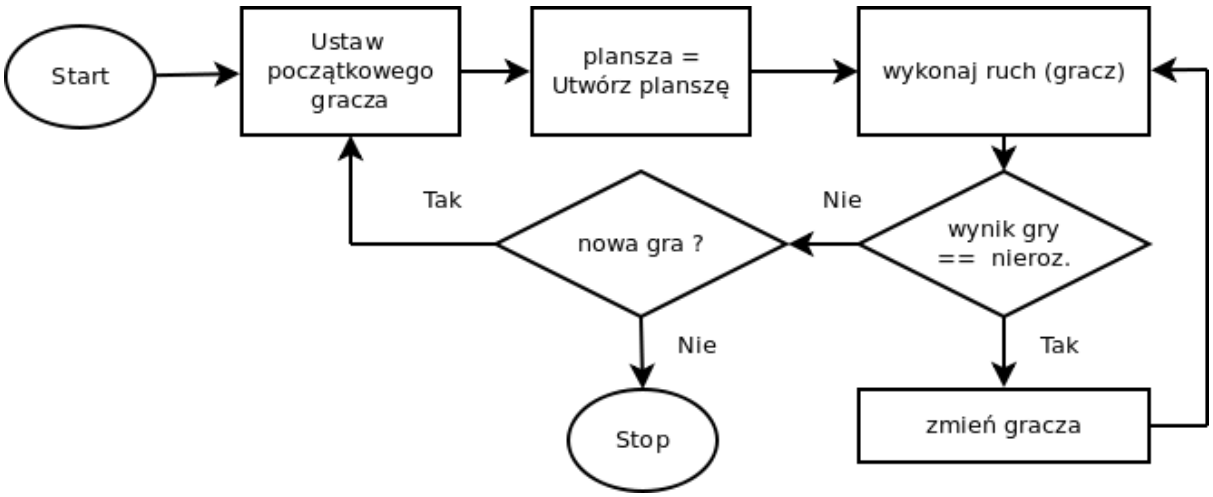
1. Sprawdzeniu, czy istnieje wiersz, kolumna lub przekątna, dla której zaznaczone są same kółka lub krzyżyki. Jeśli tak to rezultat gry jest odpowiednio: **wygraloKolko**, **wygralKrzyzyk**
2. Sprawdzenie, czy wszystkie pola są zaznaczone. Jeśli tak, to oznacza to remis, w przeciwnym przypadku gra nie została jeszcze rozstrzygnięta i można wykonywać kolejne ruchy.

Zadanie 6

Napisz metodę **static gracz zmienGracza(gracz aktualnyGracz).** Metoda jako argument przyjmuje aktualnego gracza i zwraca wynik przeciwny. Przykładowo jeśli jako argument podamy gracz **Kolko** jako wynik metody otrzymamy gracz **Krzyzyk** i odwrotnie.

Zadanie 7

Zaimplementuj algorytm działania programu, przedstawiony na poniższym rysunku.


Start
Ustaw początkowego gracza
plansza = Utwórz planszę
wykonaj ruch (gracz)
wynik gry == nieroz.
Nie
Tak
nowa gra ?
Tak
Nie
Stop
zmień gracza