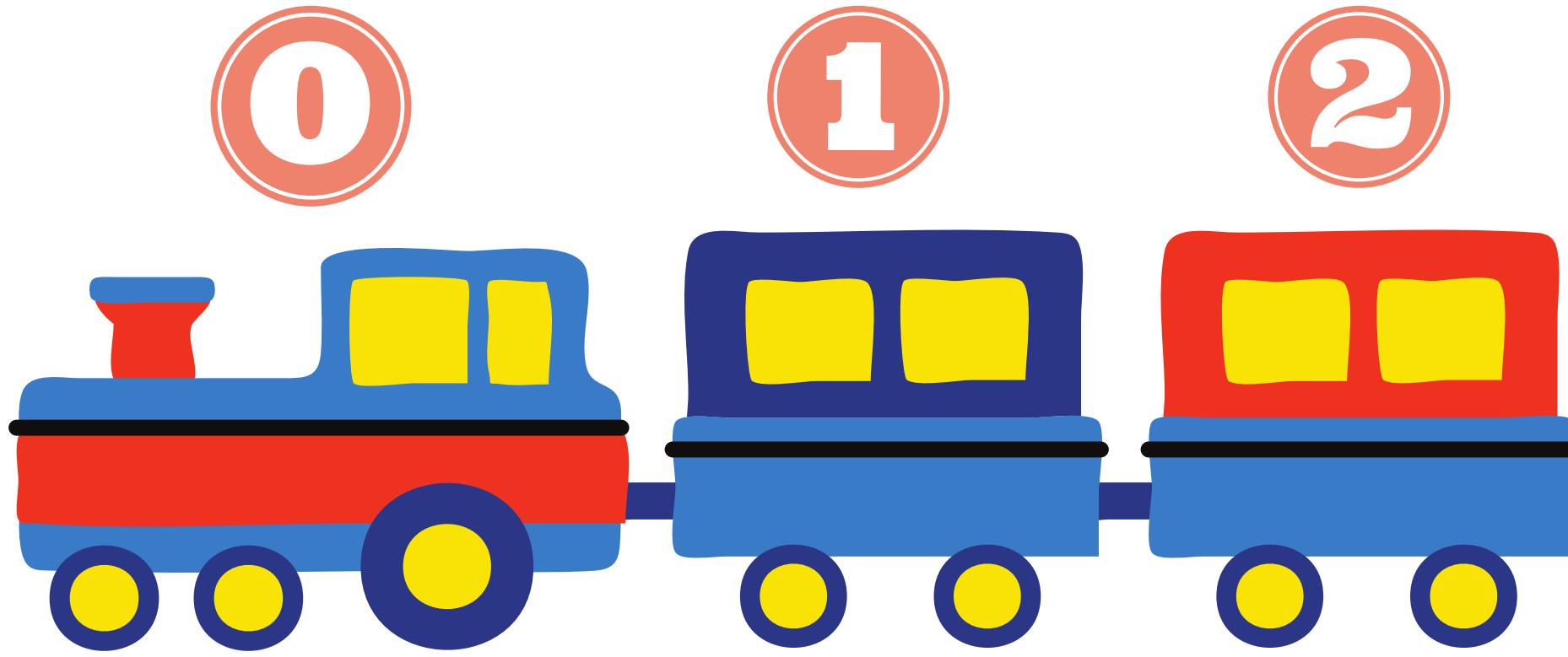




ASICODE

# Курс Основи - C#

ЗАНЯТТЯ № 6 - МАСИВИ



# Що таке масив

Масив – це структура даних зберігання елементом певного типу, має фіксований розмір. Доступ до елементів масиву здійснюється за числовим індексом.



# Оголошення масивів

Для оголошення масиву після вказівки типу його елементів ставляться квадратні дужки:

**int [] a1;**

Перед використанням масив обов'язково потрібно проініціалізувати, це можна зробити відразу, при його оголошенні:

**int [] na2 = new int [ 5 ] ; // масив із п'яти елементів типу int**

Або після оголошення:

**int [] na3;**

**na3 = new int [ 5 ] ; // масив із п'яти елементів типу int**



ASICODE

# доступ до елементів

Для доступу до елементів масиву використовуються цифрові індекси. Значення елементів масиву дорівнюють стандартному значенню для типу, масив якого був створений.

Наприклад, для масиву a3 – це будуть нулі, оскільки типу int значення за замовчуванням: 0;

```
Console.WriteLine ( na3 [ 0 ] ) ; // значення: 0  
Console.WriteLine ( na3 [ 1 ] ) ; // значення: 0
```

Якщо спробувати вивести елементи масиву na1 :

```
Console.WriteLine ( na1 [ 0 ] ) ; // помилка компіляції  
т.к. масив попередньо потрібно проініціалізувати.
```





# Ініціалізація масивів

Розглянемо різні варіанти ініціалізації масиву. Як вже було сказано, можна просто вказати кількість елементів у масиві, при цьому його елементам будуть надано значення за замовчуванням:

```
bool [] ba1 = new bool [ 3 ];  
Console.WriteLine ( "ba1[0]: " + ba1 [ 0 ].ToString () );
```

Після оголошення масиву значення елементам надаються через індекс:

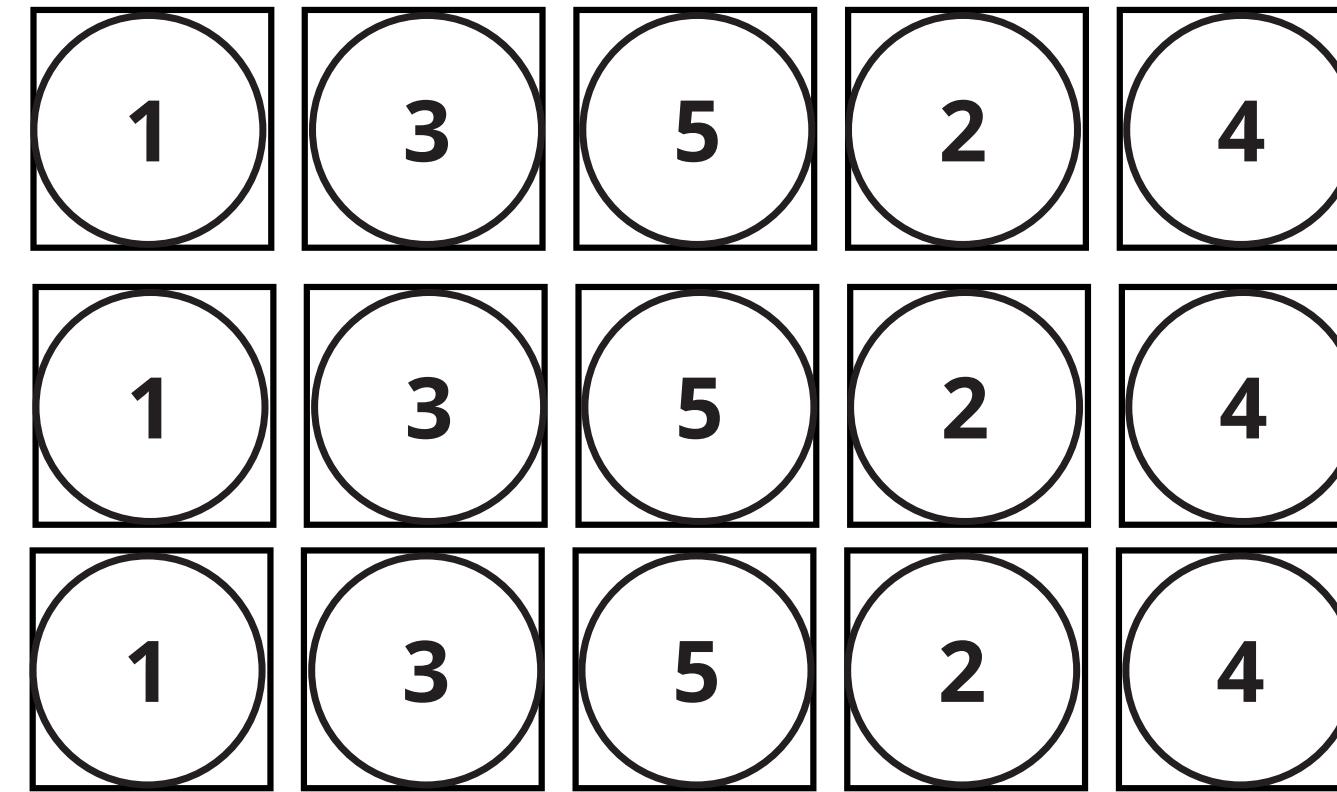
```
string [] sa1 = new string [ 3 ];  
sa1 [ 0 ] = "abc" ;  
sa1 [ 1 ] = "def" ;  
sa1 [ 2 ] = "ghi" ;  
Console.WriteLine ( $ "sa1: {sa1[0]} , {sa1[1]} , {sa1[2]} " );
```

Є можливість задати конкретні значення в момент оголошення з використанням ключового слова new та зазначенням типу:

```
double [] da1 = new double [ 3 ] { 0.1 , 0.2 , 0.3 } ;  
Console.WriteLine ( $ "da1: {da1[0]} , {da1[1]} , {da1[2]} " );
```

Або без ключового слова new :

```
double [] da2 = { 0.4 , 0.5 , 0.6 } ;  
Console.WriteLine ( $ "da2: {da2[0]} , {da2[1]} , {da2[2]} " );
```



# Багатовимірні масиви

Масиви, що має більше одного виміру, називаються багатовимірними. До цього ми працювали з одновимірними масивами. С # пропонується до використання два види багатовимірних масивів: прямокутні і зубчасті , які іноді називаються масиви масивів.

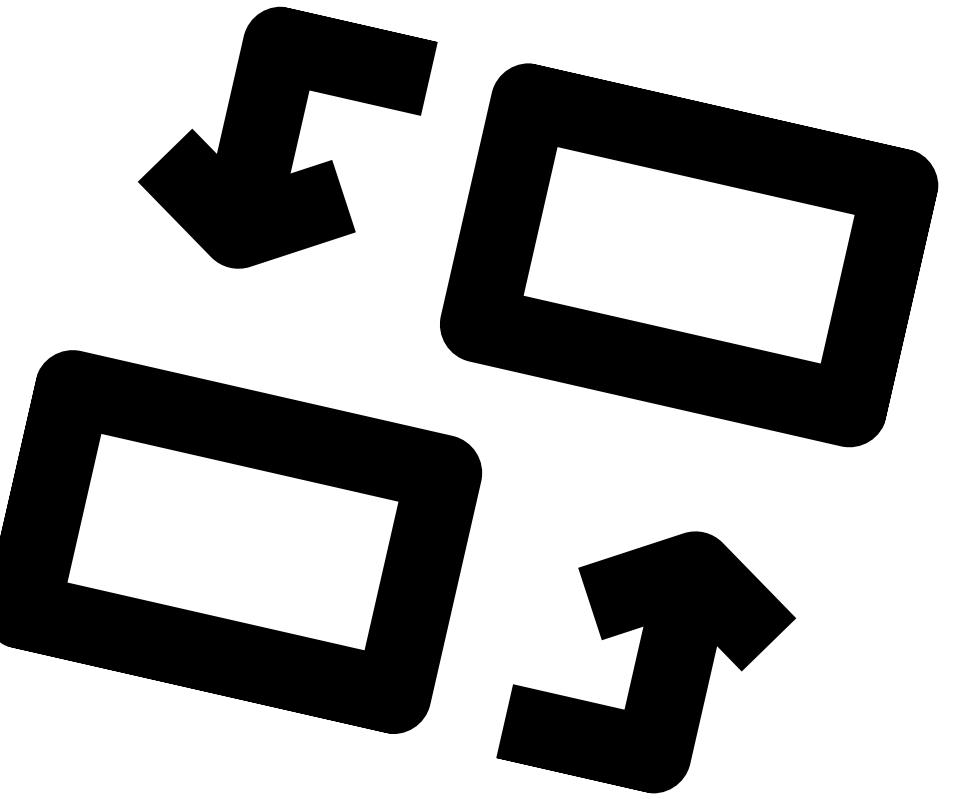
Прямокутні масиви можуть містити кілька вимірів (два і більше), при цьому кількість елементів у кожному вимірі (у кожному рядку) однаакова.

У зубчастих масивах елементами верхнього рівня є інші масиви, це дозволяє створювати багатовимірні структури, у яких рядки мають різну довжину.



# Деякі методи для масивів

- Length - Число елементів у масиві. Враховуються всі виміри.
- Rank - Ранг масиву – кількість вимірів.
- BinarySearch(Array, Object) - Виконує пошук елемента у масиві.
- Clear(Array, Int32, Int32) - Надає значення за умовчанням певному кількості елементів масиву починаючи з заданого індексу.
- Clone() - Створює копію масиву (неповну).
- Copy(Array, Array, Int32) - Копіює дані з одного масиву до іншого в заданій кількості.
- CopyTo (Array, Int32) - Копіює елементи з поточного масиву заданий, починаючи з зазначеного індексу.
- Exists<T>(T[], Predicate<T>) - Визначає наявність елемента, що задовольняє предикату.
- GetValue(Int32) - Повертає значення за вказаним індексом.
- IndexOf(Array, Object) - Повертає індекс першого входження елемента масиві.
- Reverse(Array) - Задає зворотний порядок для елементів масиві.
- Sort(Array) - Сортує елементи масиву.





ASICODE

Дякую за увагу