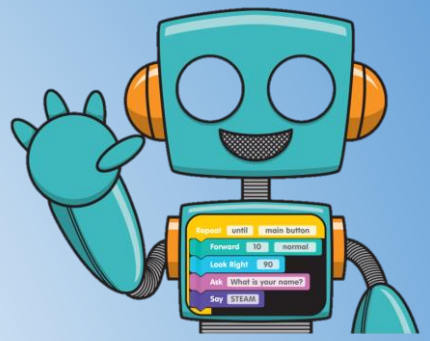


درس برنامه نویسی پیشرفته ۲

برنامه نویسی به زبان سی شارپ (C#)

مدرس: مسعود بایمانی

فصل چهارم

آرایه ها

تعریف آرایه

- به خانه های متوالی از حافظه که دارای یک نوع بوده و می توانند چندین مقدار را در خود نگهداری کنند آرایه گفته می شود.
- هر آرایه یک نام دارد و برای دسترسی به عناصر آن از متغیری به نام اندیس استفاده می گردد. از این رو به آرایه ها متغیر اندیس دار، متغیر فهرستی، لیست یا بردار نیز گفته می شود.

❖ **شیوه تعریف (اعلان) آرایه در سی شارپ:**

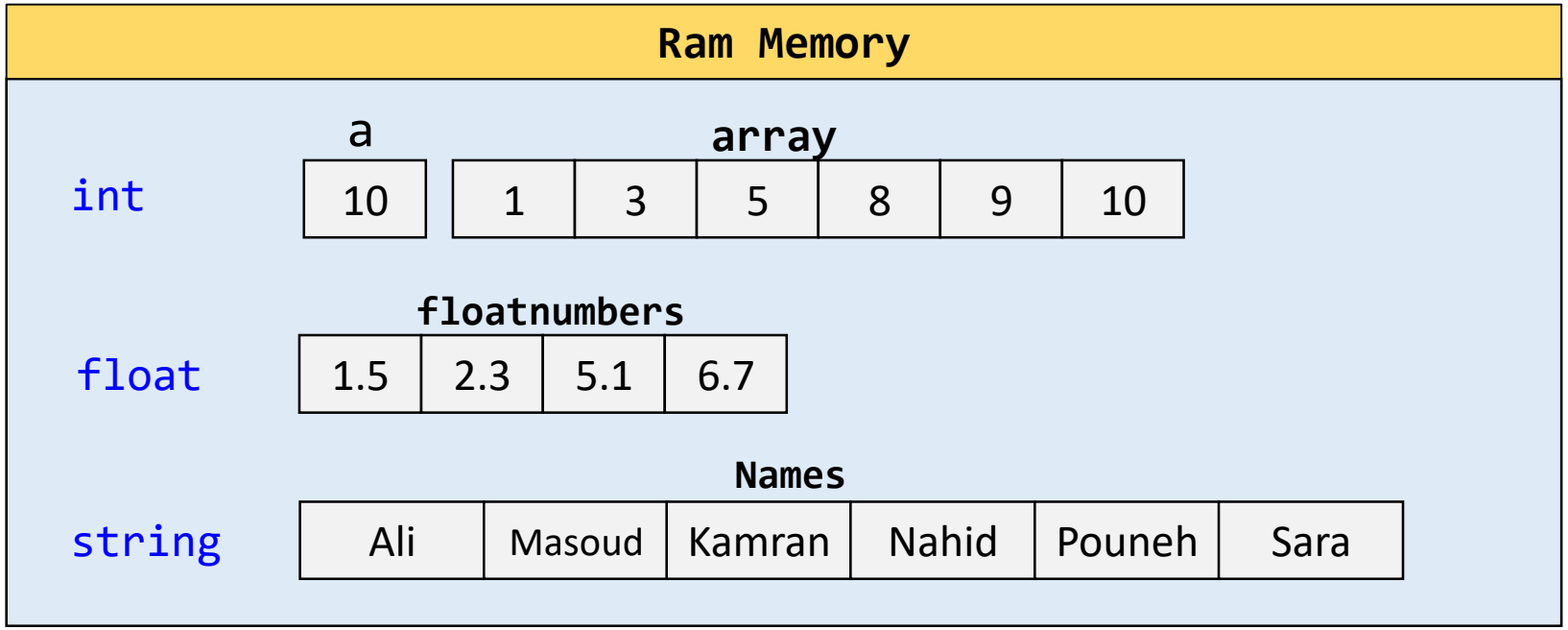
```
DataType[] array_name = {val_1,val_2,...};

DataType[] array_name= new DataType[array_length] {val_1,val_2,...};

DataType[] array_name= new DataType[array_length];
array_name[0]= val_1;
array_name[1]= val_2;
....
```

مثال

```
int a=10;
int[] array={ 1, 3, 5, 8, 9, 10};
float[] floatnumbers= { 1.5f, 2.3f, 5.1f, 6.7f };
string[] Names = { "Ali", "Reza", "Sara"  };
```



نکات کار با آرایه ها

۱- نوع آرایه یک نوع از انوع قابل تعریف در زبان C# می باشد.

۲- طول آرایه همواره یک عدد صحیح مثبت است.

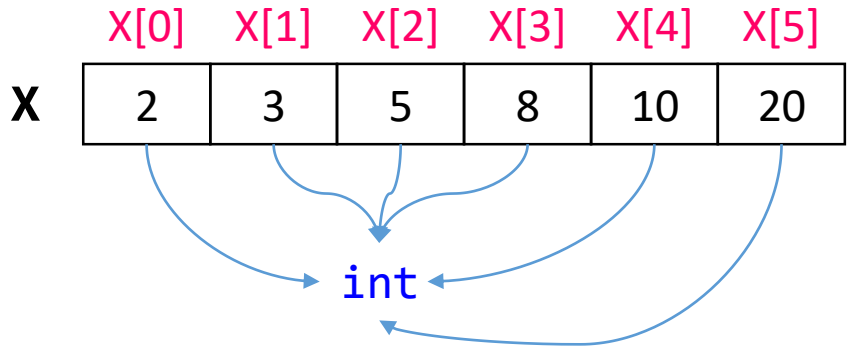
۳- برای دسترسی به عناصر آرایه نام آن به همراه شماره اندیس خانه مورد نظر استفاده می گردد.

```
int[] a = {1,2, 3, 4};

int b= a[2];
```

۴- اندیس آرایه ها در زبان C# از صفر شروع می شود و با توجه به متوالی بودن خانه های آرایه تا طول آرایه منهای ۱ واحد ادامه دارد.

```
int[] X= {2,3,5,8,10,20};
```



نکات کار با آرایه ها

۵- میزان حافظه مصرفی یک آرایه برابر است با میزان حافظه مصرفی نوع آرایه (DataType) ضربدر طول آرایه (array_length).

طول آرایه × تعداد بایت مصرفی نوع آرایه = میزان حافظه اشتغال شده توسط آرایه

```
int[] x = new int[10];

Memory consumption (X)= 4 Byte * 10 =40 Byte
```

۶- کار کردن با آرایه ها مستلزم استفاده از حلقه های تکرار الخصوص حلقه تکرار for می باشد.

۷- در سی شارپ مقدار پیش فرض برای آرایه ها عدد مقدار صفر و برای آرایه های رشته ای مقدار null می باشد.

۸- اگر تعداد مقادیری که در آرایه قرار می دهیم کمتر یا بیشتر از طول آرایه باشد با خطا مواجه می شویم.

تخصیص مقدار به خانه های آرایه

۱- تعیین مقدار در هنگام تعریف:

```
int[] X = { 1, 3, 5, 8, 9, 10 };
float[] Y = { 1.5f, 2.3f, 5.1f, 6.7f };
char[] Z = { 'a', 'b', 'c', 'd', 'e', 'f' };
string[] Names = { "Ali", "Reza", "Sara" };
```

۲- پس از اعلان آرایه و با استفاده از عملگر انتساب:

```
int[] A=new int[4];
A[0] = 124;
A[1] = 21;
A[2] = 20;
A[3] = 25;
```

```
string[] Names= new string[3];
Names[0] ="Masoud";
Names[1] = "Ali";
Names[2] = "Sara";
```

مثال ۱

- برنامه نویسید که ۱۰ عدد را از ورودی دریافت کرده و به ترتیب عکس دریافت در خروجی چاپ نماید.

```
int[] numbers = new int[10];

for (int i = 0; i < 10; i++)
{
        Console.Write("Enter Number {0}: ",(i+1));
        numbers[i] = Convert.ToInt32(Console.ReadLine());
}

Console.WriteLine("Revers is: ");
for (int i = 9; i>= 0; i--)
{
        Console.Write("{0}\t",numbers[i]);
}
Console.ReadKey();
```

مثال ۲

- برنامه نویسید که n عدد را از ورودی دریافت کرده و به ترتیب عکس دریافت در خروجی چاپ نماید.

```
int arraysize;
Console.Write("Enter the number of cells: ");
arraysize = Convert.ToInt32(Console.ReadLine());

int[] numbers_array = new int[arraysize];

for (int i = 0; i < numbers_array.Length; i++)
{
    Console.Write("Enter Number {0}: ",(i+1));
    numbers_array[i] = Convert.ToInt32(Console.ReadLine());
}

Console.WriteLine("Revers is: ");
for (int i = numbers_array.Length-1 ; i>= 0; i--)
{
    Console.Write("\t{0}",numbers_array[i]);
}
Console.ReadKey();
```

مثال ۳

- برنامه ای بنویسید که نمره ۱۰ دانشجو را از ورودی دریافت کرده و سپس اختلاف نمره هر دانشجو با مقدار بزرگترین نمره دریافتی را چاپ نماید.

```
float[] grades = new float[10];
float maxScore = 0;
for (int i = 0; i < grades.Length; i++)
{
    Console.Write("Enter Score of Student {0}: ", (i+1));
    grades[i] = float.Parse(Console.ReadLine());
    if (grades[i]> maxScore ){maxScore = grades[i];}
}

for (int i = 0; i < grades.Length; i++)
{
    float diffrence = grades[i] - maxScore;
    Console.WriteLine("Student[{0}]'s Score difference" +
                      " with Max Score : {1}", (i + 1),diffrence);
}
Console.ReadKey();
```

جستجو در آرایه ها

- به عملیات یافتن مقدار یا مقادیری مشخص در یک آرایه عملیات جستجو گفته می شود.
- مثال: برنامه ای بنویسید که لیستی از اسامی را دریافت کرد. سپس اسم دیگری را از ورودی گرفته معین کند این اسم در اسامی لیست موجود است یا خیر؟

```
String[] NamesList = { "Niloufar", "Ali", "Reza", "Parivash", "Sara" };
string find_what = string.Empty;
bool flag = false;

Console.Write("Find What: ");
find_what = Console.ReadLine();

for (int i = 0; i < NamesList.Length; i++)
     if (NamesList[i] == find_what)
     {
          Console.WriteLine("{0} is in the {1}th row of the list.", find_what, i);
          flag = true;
          break;
     }

if (!flag)
     Console.WriteLine("Not find!!!");
Console.ReadKey();
```

مرتب سازی با آرایه ها

- یکی دیگر از عملیات های پرکاربرد در علم کامپیوتر است که با استفاده از آرایه ها انجام می شود.
- الگوریتم مرتب سازی حبابی (Bubble Sort)

```
int[] A = { 3, 5, 7, 2, 1 };
int temp;

for (int i = 1; i < A.Length; i++)
{
    for (int j = 0; j < A.Length - i; j++)
    {
        if (A[j] > A[j + 1])
        {
            temp = A[j];
            A[j] = A[j + 1];
            A[j + 1] = temp;
        }
    }
}
```

```
Array is:
    3    5    7    2    1
Sorted Array is:
    1    2    3    5    7
```

حلقه foreach

حلقه foreach یکی دیگر از ساختارهای تکرار در سی شارپ می باشد که به طور خاص برای آرایه ها، لیست ها و مجموعه ها طراحی شده است. حلقه foreach با هر بار گردش در بین اجزاء، مقادیر هر یک از آنها را در داخل یک متغیر موقتی قرار میدهد و شما میتوانید بواسطه این متغیر به مقادیر دسترسی پیدا کنید. در زیر نحوه استفاده از حلقه foreach بیان شده است.

```
foreach (datatype temporaryVar in array)
{
    code to execute;
}
```

```
int i = 0;
String[] NamesList = { "Niloufar", "Ali", "Reza", "Parivash", "Sara" };

foreach (String name in NamesList)
{
    Console.WriteLine("Name[{0}]= {1}",i++,name);
}
Console.ReadKey();
```

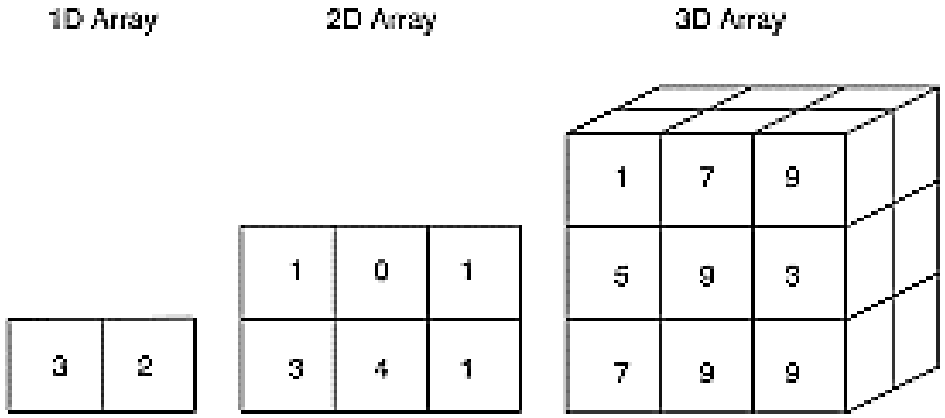
آرایه های چند بعدی

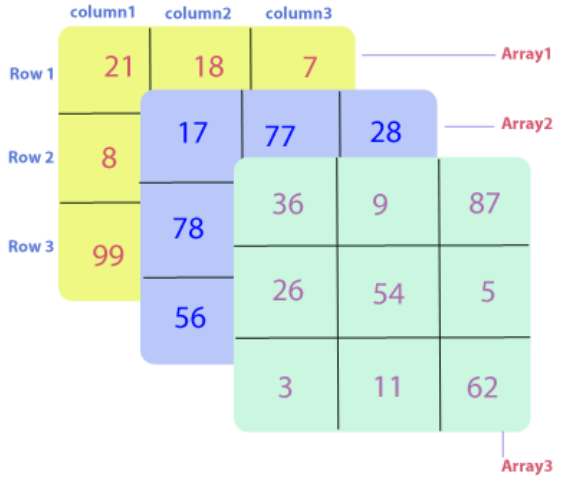
آرایه‌های چند بعدی آرایه‌هایی هستند که برای دسترسی به هر یک از عناصر آنها باید از چندین اندیس استفاده کنیم. یک آرایه چند بعدی را می‌توان مانند یک جدول با تعدادی ستون و ردیف تصور کنید. با افزایش اندیس‌ها اندازه ابعاد آرایه نیز افزایش می‌یابد و آرایه‌های چند بعدی با بیش از دو اندیس به وجود می‌آیند. نحوه ایجاد یک آرایه با دو بعدی به صورت زیر است:

```
datatype[,] arrayName = new datatype[lengthX, lengthY];
```

و یک آرایه سه بعدی به صورت زیر ایجاد می‌شود:

```
datatype[, ,] arrayName = new datatype[lengthX, lengthY, lengthZ];
```





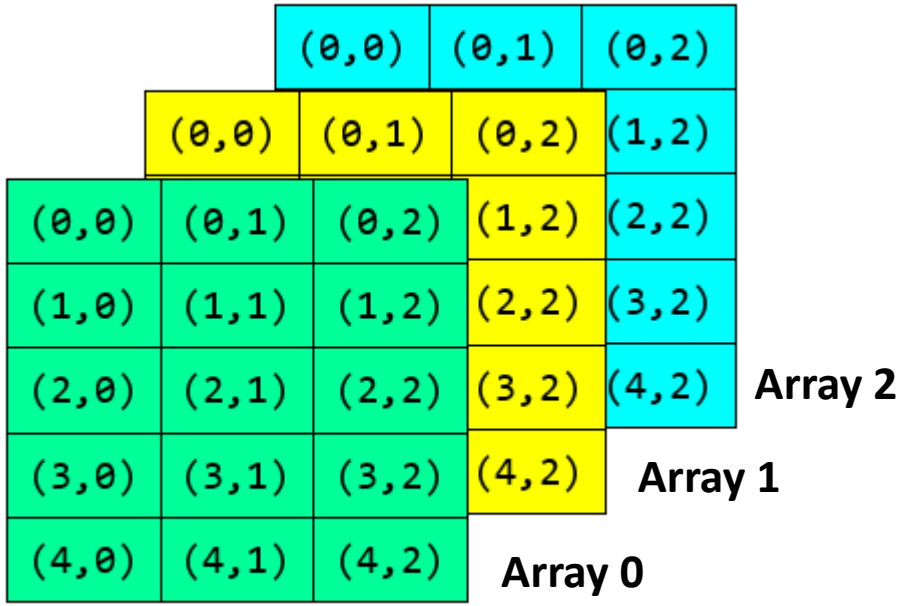
آرایه های چند بعدی

آرایه های چند بعدی در حقیقت مجموعه ای از آرایه های یک بعدی هم اندازه هستند که در کنار یکدیگر قرار گرفته اند.

```
int[,] numbers = new int[3, 5];
```

```
int[, ,] numbers = new int[3, 5, 3];
```

	Col 0	Col 1	Col 2
Row 0	(0,0)	(0,1)	(0,2)
Row 1	(1,0)	(1,1)	(1,2)
Row 2	(2,0)	(2,1)	(2,2)
Row 3	(3,0)	(3,1)	(3,2)
Row 4	(4,0)	(4,1)	(4,2)



نکات آرایه های چند بعدی

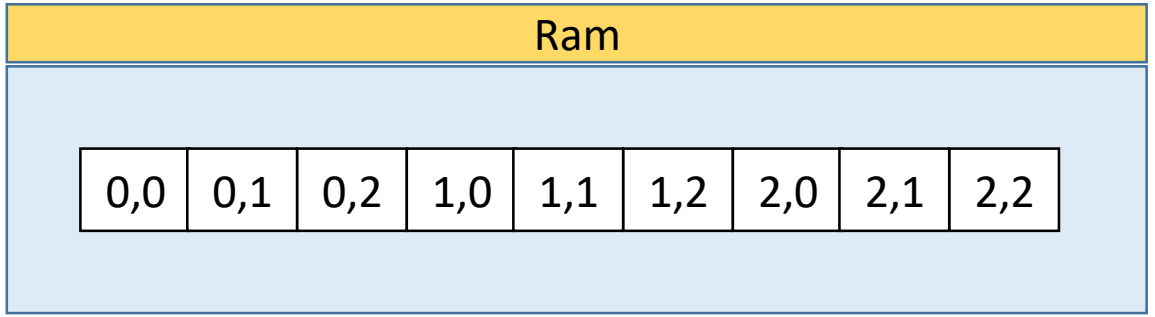
۱- می‌توان یک آرایه با تعداد زیادی بعد ایجاد کرد به شرطی که هر بعد دارای طول مشخصی باشد. با این حال اغلب برنامه نویسان حداکثر از آرایه های ۲ بعدی یا ۳ بعدی استفاده می نماید.

۲- به آرایه های یک بعدی بردار و به آرایه های دو بعدی ماتریس نیز گفته می شود.

۳- برای دستیابی به عناصر آرایه های چند بعدی باید از چند متغیر اندیس (به تعداد ابعاد آرایه) استفاده نمود.

۴- در زبان سی شارپ آرایه ها به صورت سطری در حافظه ذخیره می شوند. یعنی ابتدا سطر اول، سپس سطر دوم، سطر سوم و .... ذخیره می گردند.

0,0	0,1	0,2
1.0	1.1	1.2
2,0	2,1	2,2



مقداری دهی به آرایه های چند بعدی

```
datatype[,] arrayName = new datatype[x, y] { { r0c0, r0c1, ... r0cY },
                                             { r1c0, r1c1, ... r1cY },
                                                      .
                                                      .
                                                      .
                                             { rXc0, rXc1, ... rXcY } };
```

```
datatype[,] arrayName = { { r0c0, r0c1, ... r0cY },
                          { r1c0, r1c1, ... r1cY },
                                   .
                                   .
                                   .
                          { rXc0, rXc1, ... rXcY } };
```

```
int[,] numbers = { { 1, 2, 3, 4, 5 },
                   { 6, 7, 8, 9, 10 },
                   { 11, 12, 13, 14, 15 } };
```

گردش در میان عناصر آرایه های چند بعدی

- گردش در میان عناصر آرایه های چند بعدی نیاز به کمی دقت دارد. یکی از راه های آسان استفاده از حلقه foreach و یا حلقه for تو در تو است.

```
using System;

public class Program
{
    public static void Main()
    {
        int[,] numbers = { { 1, 2, 3, 4, 5 },
                           { 6, 7, 8, 9, 10 },
                           { 11, 12, 13, 14, 15 }
                         };

        foreach (int number in numbers)
        {
            Console.Write(number + " ");
        }
    }
}
```

گردش در میان عناصر آرایه های چند بعدی

```
int[,] numbers = { { 1, 2, 3, 4, 5 },
                   { 6, 7, 8, 9, 10 },
                   { 11, 12, 13, 14, 15 }
                 };

for (int row = 0; row < numbers.GetLength(0); row++)
{
    for (int col = 0; col < numbers.GetLength(1); col++)
    {
        Console.Write(numbers[row, col] + " ");
    }

    //Go to the next line
    Console.WriteLine();
}
```

مثال ۴

- برنامه ای بنویسید که نمره چهار درس مربوط به سه دانشجو را از ما دریافت کرده و معدل سه دانشجو را حساب کند.

```
public class Program
{
    public static void Main()
    {
        double[,] studentGrades = new double[3, 4];
        double total;

        for (int student = 0; student < studentGrades.GetLength(0); student++)
        {
            total = 0;

            Console.WriteLine("Enter grades for Student {0}", student + 1);

            for (int grade = 0; grade < studentGrades.GetLength(1); grade++)
            {
                Console.Write("Enter Grade #{0}: ", grade + 1);
                studentGrades[student, grade] = Convert.ToDouble(Console.ReadLine());
                total += studentGrades[student, grade];
            }

            Console.WriteLine("Average is {0:F2}", (total / studentGrades.GetLength(1)));
            Console.WriteLine();
        }
    }
}
```

خروجی مثال ۴

```
Enter grades for Student 1
Enter Grade #1: 92
Enter Grade #2: 87
Enter Grade #3: 89
Enter Grade #4: 95
Average is 90.75
Enter grades for Student 2
Enter Grade #1: 85
Enter Grade #2: 85
Enter Grade #3: 86
Enter Grade #4: 87
Average is 85.75
Enter grades for Student 3
Enter Grade #1: 90
Enter Grade #2: 90
Enter Grade #3: 90
Enter Grade #4: 90
Average is 90.00
```

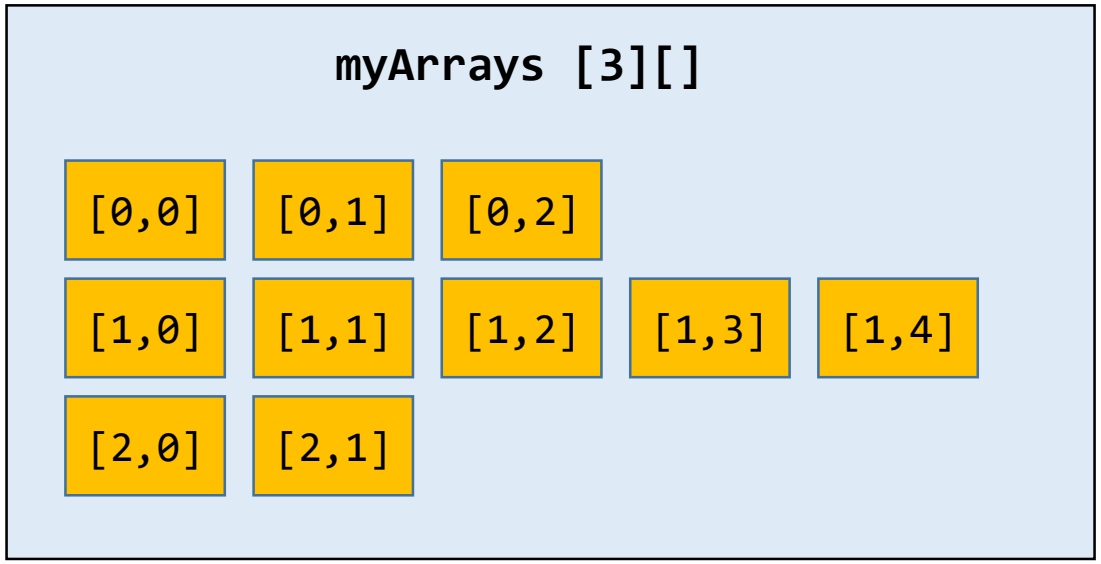
آرایه‌های دندانه‌دار

آرایه‌های دندانه‌دار نوعی از آرایه‌های چند بعدی هستند که شامل ردیف‌هایی با تعداد ستون‌های مختلف‌اند. آرایه چند بعدی ساده، آرایه‌ای به شکل مستطیل است، چون تعداد ستون‌های آن یکسان است ولی آرایه دندانه‌دار دارای سطرهایی با تعداد ستون‌های متفاوت است. بنابراین می‌توان یک آرایه دندانه‌دار را آرایه‌ای از آرایه‌ها فرض کرد.

نحوه تعریف آرایه‌های دندانه‌دار :

```
datatype[][] arrayName;
```

```
int[][] myArrays = new int[3][];
myArrays[0] = new int[3];
myArrays[1] = new int[5];
myArrays[2] = new int[2];
```



نحوه مقداردهی به آرایه‌های دندانه‌دار

```
int[][] myArrays = new int[3][];
myArrays[0] = new int[3] { 1, 2, 3 };
myArrays[1] = new int[5] { 5, 4, 3, 2, 1 };
myArrays[2] = new int[2] { 11, 22 };
```

```
int[][] myArrays = new int[3][] { new int[3] { 1, 2, 3 },
                                  new int[5] { 5, 4, 3, 2, 1 },
                                  new int[2] { 11, 22 }
                                };
```

می‌توان از ذکر طول ردیف‌های آرایه صرف نظر کرد:

```
int[][] myArrays = new int[][] { new int[] { 1, 2, 3 },
                                 new int[] { 5, 4, 3, 2, 1 },
                                 new int[] { 11, 22 }
                                };
```

نحوه مقداردهی به آرایه‌های دندانه‌دار

```
int[][] myArrays = { new int[] { 1, 2, 3 },
                     new int[] { 5, 4, 3, 2, 1 },
                     new int[] { 11, 22 }
                   };
```

به منظور دسترسی به عناصر یک آرایه دندانه‌دار می‌بایست از شماره ردیف و ستون‌های آن استفاده نمود:

```
array[row][column]
```

```
Console.WriteLine(myArrays[1][2]);
```

نکته در مورد دسترسی به عناصر آرایه‌های دندانه‌دار

۱- از یک حلقه foreach ساده نمی‌توان برای دسترسی به اجزای این آرایه‌ها استفاده کرد.

```
foreach (int array in myArrays)
    Console.WriteLine(array);
```

اگر از حلقه foreach استفاده کنیم با خطا مواجه می‌شویم، چون عناصر این نوع آرایه‌ها، خود یک آرایه هستند نه عدد یا رشته

یا ... . برای حل این مشکل باید به صورت زیر عمل کرد:

```
foreach (int[] array in myArrays)
{
    foreach (int number in array)
    {
        Console.WriteLine(number);
    }
}
```

نکته در مورد دسترسی به عناصر آرایه‌های دندانه‌دار

۲- دسترسی به عناصر آرایه‌های دندانه‌دار با استفاده از یک حلقه for تو در تو نیز قابل اجراست:

```
for (int row = 0; row < myArray.Length; row++)
{
    for (int col = 0; col < myArray[row].Length; col++)
    {
        Console.WriteLine(myArray[row][col]);
    }
}
```

در اولین حلقه for با استفاده از خاصیت Length، myArray تعداد ردیف‌های آرایه را به دست می‌آوریم. در حلقه for دوم نیز با استفاده از خاصیت Length عنصر ردیف جاری تعداد ستون‌ها را به دست می‌آوریم. سپس با استفاده از اندیس، عناصر آرایه را چاپ می‌کنیم.