

لزوم وجود ساختار آرایه‌ها  
آشنایی با احلان و مقدار دخی آرایه‌ها و استفاده‌ی درست از اندیس  
آرایه‌های چند بعدی  
تابع‌هایی که آرگومان‌شان آرایه است

---

## آرایه‌ها



# آرایه

- آرایه مجموعه‌ای به هم پیوسته و متوالی از خانه‌های حافظه است که همگی آنها دارای یک نام و همه از یک نوع داده‌ای هستند. به عبارت دیگر آرایه یک ساختار داده‌ای است که از تعدادی عضو (مولفه‌ی) مرتبط و هم‌نوع تشکیل شده است.
- به عنوان مثال آرایه‌ای با نام `c` می‌تواند ۱۲ عضو همگی از نوع `int` داشته باشد.

Name of the array is c

Position number of the element within the array	c[ 0 ]	-45
	c[ 1 ]	6
	c[ 2 ]	0
	c[ 3 ]	72
Name of an individual array element	c[ 4 ]	1543
	c[ 5 ]	-89
	c[ 6 ]	0
	c[ 7 ]	62
	c[ 8 ]	-3
	c[ 9 ]	1
	c[ 10 ]	6453
	c[ 11 ]	78

Value



# اعلان یک آرایه

- برای اعلان یک آرایه باید نوع داده‌های عضو، نام و تعداد عضوها (اندازه‌ی) آرایه مشخص شود:

[طول یا اندازه‌ی آرایه] نام آرایه نوع داده‌ای

- نام آرایه باید یک شناسه‌ی مجاز C++ باشد.
- اندازه (طول) آرایه باید یک عدد صحیح بزرگتر از صفر باشد.
- اندازه‌ی آرایه باید درون قلاب [] نوشته شود.

• مثال:

```
int c[12];
```

- اعلان یک آرایه با نام C با ۱۲ عضو از نوع صحیح: یعنی کامپایلر برای ۱۲ عضو آرایه به مقدار مورد نیاز هر عدد صحیح واحد حافظه در نظر می‌گیرد.

• مثال:

```
double x[27], b[100];
```

```
char name[20];
```



## دسترسی به عضوهای یک آرایه

- پس از اعلان یک آرایه می‌توان به عضوهای آن مقداردهی کرد یا از آن‌ها در اجرای محاسبات استفاده کرد.
- برای دسترسی به یک عضو، ابتدا نام آرایه و سپس اندیس (شماره‌ی مکان) عضو مورد نظر درون قلاب [] مشخص می‌شود.
- توجه: نخستین عضو هر آرایه دارای اندیس **صفر** است و به همین دلیل عضو صفرم هم نامیده می‌شود.
- بنابراین اگر C آرایه‌ای با ۱۲ عضو باشد، با  
 $c[0] \quad c[1] \quad c[2] \quad c[3] \quad \dots \quad c[9] \quad c[10] \quad c[11]$   
می‌توان به عضوهای آرایه‌ی C دسترسی پیدا کرد.
- دقت کنید که اندیس آخرین عضو آرایه همیشه یک واحد از اندازه‌ی آرایه کمتر است.
- پس اندیس آرایه باید یک عدد صحیح نامنفی کوچکتر از اندازه‌ی آرایه باشد.



## دسترسی به عضوهای یک آرایه

- مثال: دستور زیر مجموع سه عضو اول آرایه را در خروجی چاپ می کند:

```
cout << c[0] + c[1] + c[2] << endl;
```

- و دستور زیر عضو هفتم آرایه را بر ۲ تقسیم می کند و نتیجه را در متغیر x جایگزین می کند:

```
x = c[6] / 2;
```

- اندیس آرایه می تواند یک عبارت صحیح (از نوع int) هم باشد: در این صورت ابتدا عبارت ارزیابی شده و سپس مقدار حاصل اندیس را مشخص می کند.

- مثال: فرض کنید a و b دو متغیر صحیح باشند که به ترتیب درای مقدارهای 5 و 6 باشند. در این صورت دستور زیر به عضو c[11] آرایه مقدار 2 را اضافه می کند و نتیجه را در همین عضو جایگزین می کند:

```
c[a + b] += 2;
```



## مثال ۲۴

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    int n[10];
    for (int i = 0; i < 10; i++)
        n[i] = 0;

    cout << "Element" << setw(13)
         << "Value" << endl;
    for (int j = 0; j < 10; j++)
        cout << setw(7) << j << setw(13)
             << n[j] << endl;
    return 0;
}
```



## شرح مثال ۲۴

- در این برنامه ابتدا آرایه‌ای از نوع `int` با نام `n` و به اندازه‌ی ۱۰ اعلان شده است.
- سپس از یک حلقه برای مقدار دهی اولیه به عضوهای آرایه استفاده شده است. در اینجا همه‌ی عضوهای آرایه برابر با صفر قرار داده شده‌اند.
- در پایان از یک حلقه‌ی دیگر برای چاپ عضوهای آرایه استفاده شده است.
- خروجی برنامه به شکل زیر است:

Element	Value
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0



## مثال ۲۵

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    int n[10] = {32, 27, 68, 18, 95, 14, 90, 70,
                60, 37};

    cout << "Element" << setw(13)
         << "Value" << endl;
    for (int j = 0; j < 10; j++)
        cout << setw(7) << j << setw(13)
             << n[j] << endl;
    return 0;
}
```





## شرح مثال ۲۶

- در این برنامه، هم‌زمان با اعلان آرایه مقدار دهی اولیه نیز صورت گرفته است.
- برای این منظور پس از اعلان، یک عملگر جایگزینی قرار دارد و سپس داخل آکولاد باز و بسته مقادارهای اولیه، که با کاما از هم جدا شده‌اند، نوشته شده است.
- اگر تعداد مقادارهای اولیه کمتر از تعداد عضوها (اندازه‌ی) آرایه باشد، عضوهای باقیمانده به صورت پیش‌فرض مقدار اولیه‌ی صفر می‌گیرند.
- پس اعلان

```
int n[10] = {0};
```

- به طور صریح مقدار اولیه‌ی عضو اول را صفر و به طور ضمنی مقدار اولیه‌ی سایر عضوها را صفر قرار می‌دهد.
- اگر اندازه‌ی آرایه و لیست مقداردهی اولیه در اعلان مشخص شده باشند، آن‌گاه تعداد مقادارهای اولیه باید کوچکتر یا برابر تعداد عضوهای آرایه باشد.
- مثال: اعلان زیر منجر به خطای زمان کامپایل می‌شود زیرا تعداد مقادارهای اولیه از اندازه‌ی آرایه بزرگتر است:

```
int n[5] = {32, 27, 64, 18, 95, 14};
```



## شرح مثال ۲۶

- اگر اندازه‌ی آرایه در اعلان آرایه که دارای لیست مقداردهی اولیه است حذف شود، کامپایلر تعداد عضوهای آرایه را برابر با تعداد عضوهای مقادیر لیست مقداردهی اولیه قرار می‌دهد.
- پس اعلان

```
int n[] = {32, 27, 64, 18, 95, 14};
```

- یک آرایه‌ای پنج عضوی را با نام n ایجاد می‌کند.
- خروجی برنامه به شکل زیر است:

Element	Value
0	32
1	27
2	68
3	18
4	95
5	14
6	90
7	70
8	60
9	37



## مثال ۲۷

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    const int arraySize = 10;
    int s[arraySize];
    for (int i = 0; i < arraySize; i++)
        s[i] = 2 + 2 * i;
    cout << "Element" << setw(13)
         << "Value" << endl;
    for (int j = 0; j < arraySize; j++)
        cout << setw(7) << j << setw(13)
             << s[j] << endl;
    return 0;
}
```



## شرح مثال ۲۷

- توصیف کننده‌ی `const` برای اعلان یک متغیر ثابت به کار می‌رود.
- متغیرهای ثابت باید هنگام اعلان با یک عبارت ثابت مقدار اولیه بگیرند.
- مقدار یک متغیر ثابت پس از اعلان نمی‌تواند در برنامه تغییر کند.
- دستور

```
const int arraySize = 10;
```

- متغیر ثابت `arraySize` را با مقدار ۱۰ اعلان می‌کند.
- عدم مقداردهی اولیه و جایگزین کردن مقدار در متغیرهای ثابت پس از اعلان خطای زمان کامپایل است:

```
const int x; → خطا
```

```
x = 7; → خطا
```

- برای اعلان اندازه‌ی یک آرایه با یک متغیر، تنها می‌توان از متغیرهای ثابت `(const)` استفاده کرد. عدم استفاده از `const` برای این منظور خطای زمان کامپایل است.



## شرح مثال ۲۷

- استفاده از متغیرهای ثابت برای مشخص کردن طول آرایه‌ها قابلیت تغییر برنامه را بالا می‌برد و خوانایی آن را افزایش می‌دهد.
- در برنامه‌ی قبل اگر بخواهیم طول آرایه را از 10 به 1000 تغییر دهیم کافیست دستور

```
const int arraySize = 10;
```

را با دستور زیر جایگزین کنیم:

```
const int arraySize = 1000;
```

- خروجی برنامه به این شکل است:

Element	Value
0	2
1	4
2	6
3	8
4	10
5	12
6	14
7	16
8	18
9	20



## مثال ۲۸ (جمع مقادیرهای عضوهای یک آرایه)

```
#include <iostream>
using namespace std;

int main()
{
    const int arraySize = 10;
    int x[arraySize];
    for (int i = 0; i < arraySize; i++)
    {
        cout << "\nEnter a grade: ";
        cin >> x[i];
    }

    int total = 0;
    for (int j = 0; j < arraySize; j++)
        total += x[j];
}
```



## مثال ۲۸

```
cout << "Total of array elements: "  
    << total << endl;  
  
cin.get();  
return 0;  
}
```

• یک خروجی نوعی برنامه

```
Enter a grade: 87  
Enter a grade: 68  
Enter a grade: 94  
Enter a grade: 100  
Enter a grade: 83  
Enter a grade: 78  
Enter a grade: 89  
Enter a grade: 91  
Enter a grade: 76  
Enter a grade: 87  
Total of array elements: 853
```



## مثال ۲۹ (نمودار توزیع فراوانی)

- مثال: تعداد فرزندان ۲۰۰ خانواده در یکی از شهرهای ایران به قرار جدول زیر است (بهبودیان ۱۳۸۰).

$x_i$	$f_i$
0	20
1	30
2	50
3	40
4	30
5	20
6	10
مجموع	200





## مثال ۲۹

```
#include <iostream>
using namespace std;

int main()
{
    const int k = 7;
    int x[k] = {0, 1, 2, 3, 4, 5, 6};
    int n[k] = {20, 30, 50, 40, 30, 20, 10};
    cout << "Number of children: " << endl;
    for (int i = 0; i < k; i++)
    {
        cout << x[i] << ": ";
        for (int stars = 0; stars < n[i]; stars++)
            cout << '*';

        cout << endl;
    }
}
```



## مثال ۲۹

```
return 0;  
}
```

• خروجی برنامه

Number of children:

```
0: *****  
1: *****  
2: *****  
3: *****  
4: *****  
5: *****  
6: *****
```



## مثال ۳۰ (استفاده از آرایه به عنوان شمارنده)

```
#include <iomanip>
#include <cstdlib>
#include <ctime>
using namespace std;

int main()
{
    const int m = 7;
    int frequency[m] = {0};

    srand(time(0));

    for (int roll = 1; roll <= 6000000; roll++)
        frequency[ 1 + rand() % 6 ]++;
```



## مثال ۳۰ (استفاده از آرایه به عنوان شمارنده)

```
for (int face = 1; face < arraySize; face++)  
    cout << setw(4) << face << setw(13)  
        << frequency[face] << endl;  
  
cin.get();  
return 0;  
}
```

- یک خروجی نوعی برنامه

1	999605
2	1001321
3	1000622
4	999699
5	999696
6	999057



## شرح مثال ۳۰

- تابع `srand` یک آرگومان از نوع `unsigned int` می‌گیرد تا با هر بار اجرای برنامه، دنباله‌ای متفاوت از اعداد شبه تصادفی تولید کند.
- تابع `srand` در سرفایل `<cstdlib>` است و به فضای نام `std` تعلق دارد.
- تابع `time` زمان محلی را بر حسب ثانیه‌های طی شده از نیمه‌شب اول ژانویه ۱۹۷۰ به وقت گرینویچ را بر می‌گرداند.
- تابع `time` در سرفایل `<ctime>` است و به فضای نام `std` تعلق دارد.
- دستور

```
srand(time(0));
```

- باعث می‌شود کامپیوتر برای بدست آوردن مقدار اولیه برای تولید اعداد شبه تصادفی، به طور اتوماتیک ساعت سیستم را بخواند.
- اگر از دستور بالا استفاده نشود در هر اجرای برنامه اعداد شبه تصادفی یکسانی تولید خواهد شد.



## مثال ۳۱

- داده‌های مربوط به نظر ۴۰ دانشجو در مورد کیفیت غذای سالن غذاخوری دانشگاه بر اساس مقیاس ۱ تا ۱۰ (۱ خیلی بد، ۱۰ خیلی خوب) در اختیار است. برنامه‌ی زیر داده‌ها را در یک آرایه‌ی صحیح قرار می‌دهد و فراوانی هر جواب را محاسبه و در خروجی چاپ می‌کند.



## مثال ۳۱

```
#include<iostream>
#include<iomanip>
using namespace std;

int main()
{
    const int N = 40;
    const int k = 11;

    const int responses[N] = {1, 2, 6,
        4, 8, 5, 9, 7, 8, 10, 1, 6,
        3, 8, 6, 10, 3, 8, 2, 7, 6,
        5, 7, 6, 8, 6, 7, 5, 6, 6,
        5, 6, 7, 5, 6, 4, 8, 6, 8, 10 };

    int frequency[k] = {0};
```



## مثال ۳۱

```
for (int answer=0; answer < N; answer++)  
    frequency[ responses[answer] ]++;  
  
cout << "Rating" << setw(17)  
    << "Frequency" << endl;  
  
for (int rating=1; rating < k; rating++)  
    cout << setw(6) << rating << setw(17)  
        << frequency[rating] << endl;  
  
return 0;  
}
```

Rating	Frequency
1	2
2	2
3	2
4	2
5	5
6	11
7	5
8	7
9	1
10	3





## شرح مثال ۳۱

- آرایه‌ی `responses` به صورت `const` اعلان شده است پس مقدار عضوهای آن در برنامه نمی‌توانند (و نباید هم) تغییر کند.
- اگر داده‌هایی که در آرایه‌ی `responses` قرار دارند حاوی یک مقدار غیرمجاز مانند ۱۳ باشند، برنامه تلاش می‌کند یک واحد به `frequency[13]` اضافه کند که چون آرایه‌ی `frequency` دارای ۱۱ عضو است، این اندیس خارج از کران آرایه است.
- `C++` کران‌های آرایه‌ها را کنترل نمی‌کند و بنابراین نمی‌تواند از دسترسی به عضوی که در آرایه وجود ندارد جلوگیری کند.
- دسترسی به عضوی که در خارج از کران‌های آرایه قرار دارد خطای منطقی زمان اجراست. این خطا، خطای دستوری نیست.
- هنگام پردازش آرایه‌ها به وسیله‌ی حلقه‌ها، برنامه‌نویس باید مطمئن شود اندیس آرایه هرگز از صفر کمتر نمی‌شود و همیشه یک واحد از طول آرایه کمتر است.

