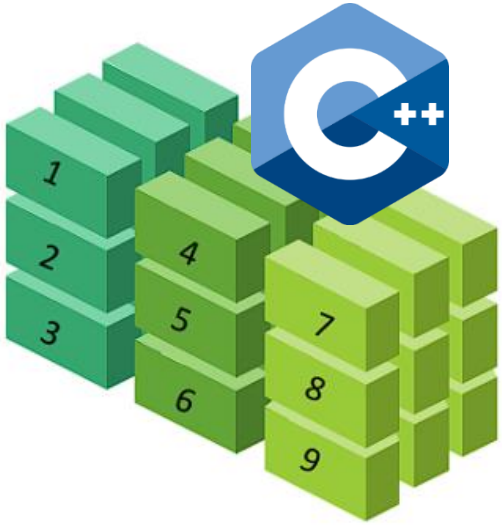
مبانی کامپیوتر و برنامه سازی

اسلاید یازدهم

«برنامه نویسی مقدماتی به زبان C++»

بخش چهارم : آرایه ها



درس

مبانی کامپیوتر

۱۱

محمد سعید صفایی صادق

(استفاده از اسلایدها صرفا برای دانشجویان مجاز می باشد!)

۱۴۰۲

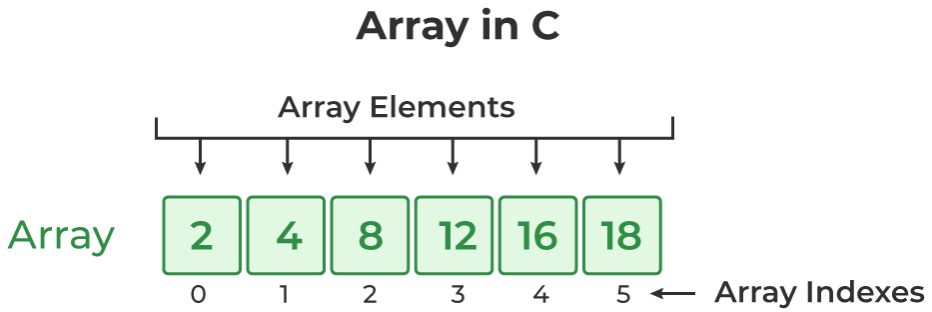
www.SaeidSafaei.ir

منطق آرایه ها

آرایه ها

در برنامه‌هایی که **داده‌های فراوانی** را پردازش می‌کنند استفاده از متغیرهای معمولی کار عاقلانه‌ای نیست زیرا در بسیاری از این برنامه‌ها **پردازش دسته‌ای** صورت می‌گیرد.

به این معنی که مجموعه‌ای از داده‌های مرتبط با هم در حافظه قرار داده می‌شود و پس از پردازش، کل این مجموعه از حافظه خارج می‌شود و مجموعه بعدی در حافظه بارگذاری می‌شود.

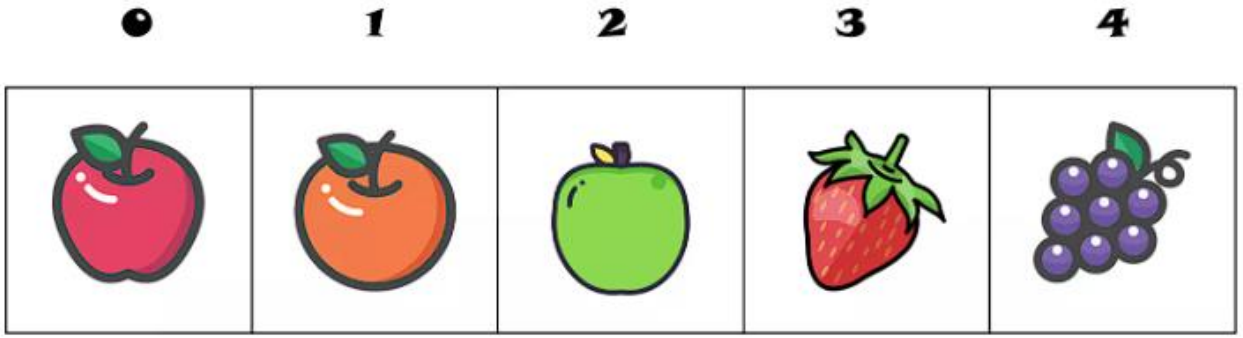


آرایه ها

اگر قرار باشد برای این کار از متغیرهای معمولی استفاده شود بیشتر وقت برنامه‌نویس صرف پر و خالی کردن انبوهی از متغیرها می‌شود.

به همین دلیل در بیشتر زبان‌های برنامه‌نویسی آرایه‌ها تدارک دیده شده‌اند.

آرایه را می‌توان متغیری تصور کرد که یک نام دارد ولی چندین مقدار را به طور هم‌زمان نگهداری می‌نماید.

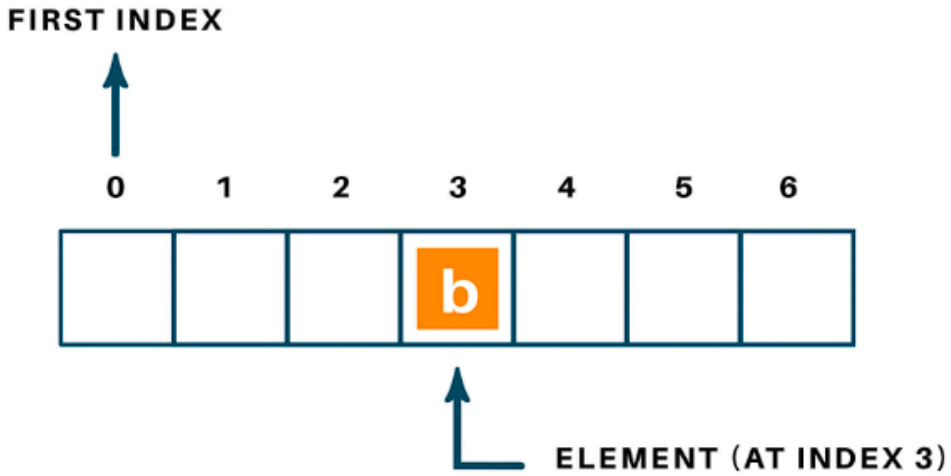


آرایه ها

یک آرایه، یک زنجیره از متغیرهایی است که همه از یک نوع هستند.

به این متغیرها «اعضای آرایه» می‌گویند.

هر عضو آرایه با یک شماره مشخص می‌شود که به این شماره «ایندکس» یا «زیرنویس» می‌گویند



آرایه ها

عناصر یک آرایه در خانه‌های پشت سر هم در حافظه ذخیره می‌شوند.

به این ترتیب آرایه را می‌توان بخشی از حافظه تصور کرد که این بخش خود به قسمت‌های مساوی تقسیم شده و هر قسمت به یک عنصر تعلق دارد.

0	17.50
1	19.00
2	16.75
3	15.00
4	18.00

شکل مقابل آرایهٔ a که پنج عنصر دارد را نشان می‌دهد.

عنصر a[0] حاوی مقدار ۱۷.۵ و عنصر a[1] حاوی ۱۹.۰ و عنصر a[4] حاوی مقدار ۱۸.۰ است.

انواع آرایه:

۱- آرایه با اندازه ثابت: اندازه آرایه قبل از اجرا در زمان کامپایل مشخص می شود و در حین اجرا قابل تغییر نیست.

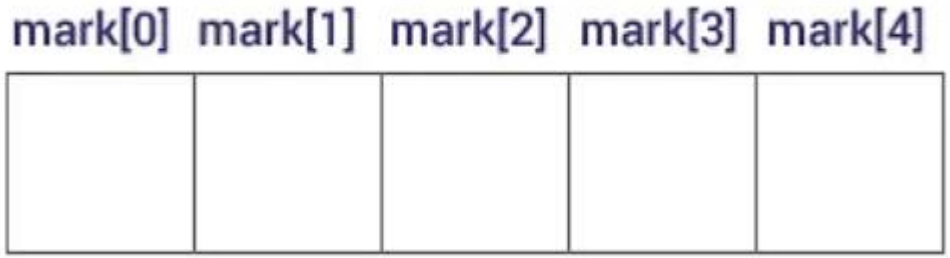
۲- آرایه با اندازه متغییر (آرایه پویا): اندازه آرایه در زمان اجرا/یا در زمان کامپایل مشخص می شود، و در حین اجرا قابل تغییر است.

نوع دوم آرایه ها با استفاده از اشاره گرها pointer تعریف می شوند.

چند نکته کلیدی

۱- توجه کنید که نخستین اندیس آرایه‌های C++ از **صفر** و نه یک آغاز می‌شود.

در این مثال mark[0] نخستین عنصر است.



۲- اگر اندازه آرایه برابر با n باشد، برای دسترسی به آخرین عنصر باید از اندیس n-1 استفاده شود. در مثال فوق mark[4] آخرین عنصر آرایه است.

چند نکته کلیدی

۳- فرض کنید آدرس آغازین a[0] برابر با ۲۱۲۰ dباشد.

در این صورت آدرس بعدی a[1] برابر با d۲۱۲۴، آدرس a[2] برابر با ۲۱۲۸ dو همین طور تا آخر خواهد بود.

دلیل این مسئله آن است که اندازه آرایه ۴ بایت است.

مقداردهی آرایه ها در C++

- نحو کلی برای اعلان آرایه به شکل زیر است:

```
type array_name[array_size];
```

- عبارت type نوع عناصر آرایه را مشخص می‌کند.
- *array_name* نام آرایه است .
- array_size تعداد عناصر آرایه را نشان می‌دهد.
- این مقدار باید یک عدد ثابت صحیح باشد و حتما باید داخل کروشه [] قرار بگیرد.

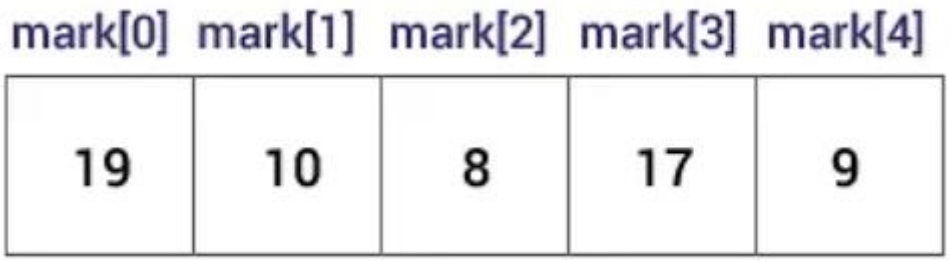
مقداردهی اولیه آرایه در C++

امکان مقداردهی اولیه یک آرایه در زمان اعلان آن وجود دارد. برای نمونه:

```
int mark[5] = {19, 10, 8, 17, 9};
```

روش دیگر برای مقداردهی اولیه آرایه در زمان اعلان به صورت زیر است:

```
int mark[] = {19, 10, 8, 17, 9};
```



مقداردهی اولیه آرایه در C++

- به این ترتیب مقادیر داخل فهرست به همان ترتیبی که چیده شده‌اند درون عناصر آرایه قرار می‌گیرند.
- اندازه آرایه نیز برابر با تعداد عناصر موجود در فهرست خواهد بود.
- پس همین خط مختصر، آرایه‌ای از نوع int و با نام mark و با تعداد 5 عنصر اعلان کرده و هر 5 عنصر را با مقدارهای درون فهرست، مقداردهی می‌کند.

mark[0]	mark[1]	mark[2]	mark[3]	mark[4]
19	10	8	17	9

نکته:

-هنگام استفاده از فهرست مقداردهی برای اعلان آرایه، می‌توانیم تعداد عناصر آرایه را هم به طور صریح ذکر کنیم.

در این صورت اگر تعداد عناصر ذکر شده از تعداد عناصر موجود در فهرست مقداردهی بیشتر باشد، خانه‌های بعدی با مقدار صفر پر می‌شوند:

	a
0	55.5
1	66.6
2	77.7
3	0.0
4	0.0
5	0.0
6	0.0

```
float a[7] = { 55.5, 66.6, 77.7 };
```

-دقت کنید که تعداد مقادیر موجود در فهرست مقدار دهی نباید از تعداد عناصر آرایه بیشتر باشد:

```
float a[3] = { 22.2, 44.4, 66.6, 88.8 };
// ERROR: too many values!
```

مقداردهی اولیه (پیش فرض)

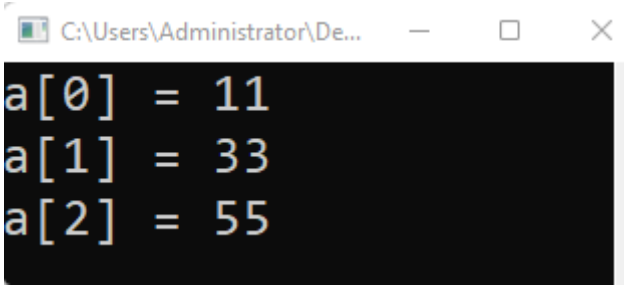
مثال: (هریک از مقداردهی های زیر را تحلیل کنید)

1. `int a[] = {22,44,66};`
2. `int a[2] = {22,44};`
3. `int a[2] = {22,44,66};`
4. `int a[4] = {22};`

مثال: دستیابی مستقیم به عناصر آرایه

برنامه ساده زیر یک آرایه سه عنصری را تعریف می‌کند و سپس مقادیری را در آن قرار داده و سرانجام این مقادیر را چاپ می‌کند:

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int a[3];
6     a[2] = 55;
7     a[0] = 11;
8     a[1] = 33;
9     cout << "a[0] = " << a[0] << endl;
10    cout << "a[1] = " << a[1] << endl;
11    cout << "a[2] = " << a[2] << endl;
12 }
```



مقداردهی با عملگر تخصیص مقدار

یک آرایه را می‌توانیم **به طور کامل با صفر مقداردهی اولیه** کنیم.
برای مثال سه اعلان زیر با هم برابرند:

```
float a[ ] = { 0, 0, 0, 0, 0, 0, 0, 0, 0 };
float a[9] = { 0, 0 };
float a[9] = { 0, 0, 0, 0, 0, 0, 0, 0, 0 };
```

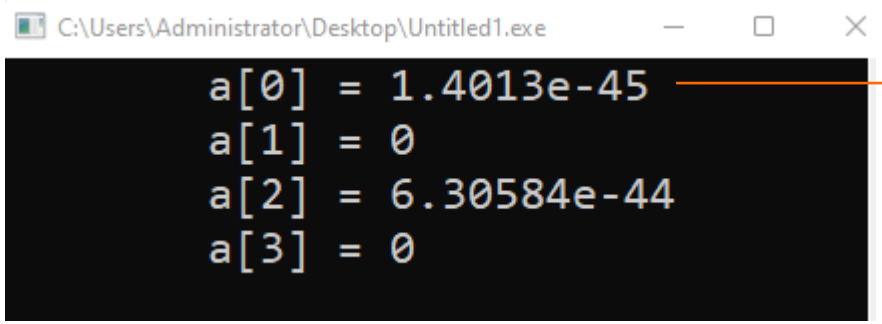
اما مطلب فوق اصلا به این معنی نیست که از فهرست مقداردهی استفاده نشود.

درست مثل یک متغیر معمولی، اگر یک آرایه مقداردهی اولیه نشود، عناصر آن **حاوی مقادیر زباله** خواهد بود.

مثال: آرایه مقدار دهی نشده

برنامهٔ زیر، آرایهٔ a را اعلان می‌کند ولی مقداردهی نمی‌کند.
با وجود این، مقادیر موجود در آن را چاپ می‌کند:

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      const int SIZE=4; // defines the size N for 4 elements
6      float a[SIZE];    // declares the array's elements as float
7      for (int i=0; i<SIZE; i++)
8        cout << "\ta[" << i << "] = " << a[i] << endl;
9  }
```



1.4013*10^-45

پردازش آرایه ها : دستور sizeof[] (مقدار بیت حافظه)

مثال : مقداردهی آرایه با استفاده از فهرست مقداردهی برنامه زیر، آرایه a را مقداردهی کرده و سپس مقدار هر عنصر را چاپ می‌کند:

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {  float a[] = { 22.2, 44.4, 66.6 };
5     int size = sizeof(a)/sizeof(float);
6     for (int i=0; i<size; i++)
7        cout << "\ta[" << i << "] = " << a[i] << endl;
8  }
```

هر مقدار آرایه ۴ بیت را اشغال میکند

12/4=3

```
C:\Users\Administrator\Desktop\Untitle...
        a[0] = 22.2
        a[1] = 44.4
        a[2] = 66.6
```

آرایه‌ها را می‌توان با استفاده از عملگر جایگزینی مقداردهی کرد اما نمی‌توان مقدار آن‌ها را به یکدیگر تخصیص داد:

```
float a[7] = { 22.2, 44.4, 66.6 };
float b[7] = { 33.3, 55.5, 77.7 };
b = a;      // ERROR: arrays cannot be assigned!
            // invalid array assignment
```

همچنین نمی‌توانیم یک آرایه را به طور مستقیم برای مقداردهی به آرایهٔ دیگر استفاده کنیم:

```
float a[7] = { 22.2, 44.4, 66.6 };
float b[7] = a;     // ERROR: arrays cannot be used as nitializers!
```

دستیابی مستقیم به عناصر آرایه

برنامهٔ سادهٔ زیر یک آرایهٔ سه عنصری را تعریف می‌کند و سپس مقادیری را از کاربردریافت و چاپ می کند

```
1  int main()
2  {
3      int a[3];
4      cin>>a[0];
5      cin>>a[2];
6      cin>>a[1];
7      for(int i=0;i<3;i++)
8       cout << "a["<<i<<"] = " << a[i] << endl;
9  }
```

مقداردهی توسط کاربر

ایندکس بیرون از حدود آرایه

در بعضی از زبان‌های برنامه‌نویسی، ایندکس آرایه نمی‌تواند از محدودهٔ تعریف شده برای آن بیشتر باشد.

برای مثال در **پاسکال** اگر آرایهٔ a با تعداد پنج عنصر تعریف شده باشد و آنگاه a[7] دستیابی شود، **برنامه از کار می‌افتد.**

این سیستم حفاظتی در C++ وجود ندارد.

مثال بعدی نشان می‌دهد که ایندکس یک آرایه هنگام دستیابی می‌تواند بیشتر از عناصر تعریف شده برای آن باشد و باز هم بدون این که خطایی گرفته شود، برنامه ادامه یابد.

محدوده آرایه ها

تجاوز ایندکس آرایه از محدودهٔ تعریف شده برای آن برنامهٔ زیر یک خطای زمان اجرا دارد؛

به بخشی از حافظه دستیابی می‌کند که از محدودهٔ آرایه بیرون است:

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {   const int SIZE=4;
5      float a[SIZE] = { 33.3, 44.4, 55.5, 66.6 };
6      for (int i=0; i<7; i++) //ERROR: index is out of bounds!
7          cout << "\ta[" << i << "] = " << a[i] << endl;
8  }
```

```
C:\Users\Administrator\Desktop\Untitled1.exe
        a[0] = 33.3
        a[1] = 44.4
        a[2] = 55.5
        a[3] = 66.6
        a[4] = 2.07011e-38
        a[5] = 0
        a[6] = 5.60519e-45
```

آرایه‌ای که در این برنامه تعریف شده، چهار عنصر دارد ولی تلاش می‌شود به هفت عنصر دستیابی شود.

سه مقدار آخر واقعا جزو آرایه نیستند و فقط سلول‌هایی از حافظه‌اند که دقیقا بعد از عنصر چهارم آرایه قرار گرفته‌اند.

این سلول‌ها دارای مقدار زباله هستند.

اثر همسایگی

برنامهٔ زیر از ایندکس خارج از محدوده استفاده می‌کند و این باعث می‌شود که مقدار یک متغیر به طور ناخواسته تغییر کند:

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {float a[] = { 22.2, 44.4, 66.6 };
5     float x=11.1;
6     cout << "x = " << x << endl;
7     a[3] = 88.8;    // ERROR: index is out of bounds!
8     cout << "x = " << x << endl;
9  }
```

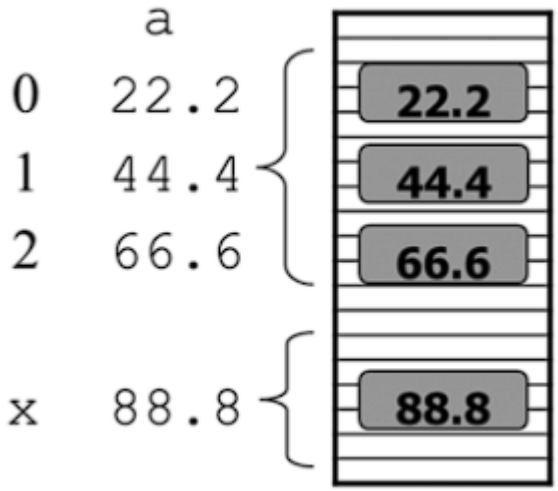
```
C:\Users\Adm...
x = 11.1
x = 88.8
```

اثر همسایگی

متغیر x بعد از آرایهٔ a اعلان شده، پس یک سلول چهاربایتی بلافاصله بعد از دوازده بایت آرایه به آن تخصیص می‌یابد.

بنابراین وقتی برنامه تلاش می‌کند مقدار ۸۸.۸ را در a[3] قرار دهد (که جزو آرایه نیست) این مقدار به شکل ناخواسته در x قرار می‌گیرد.

شکل مقابل نشان می‌دهد چطور این اتفاق در حافظه رخ می‌دهد.



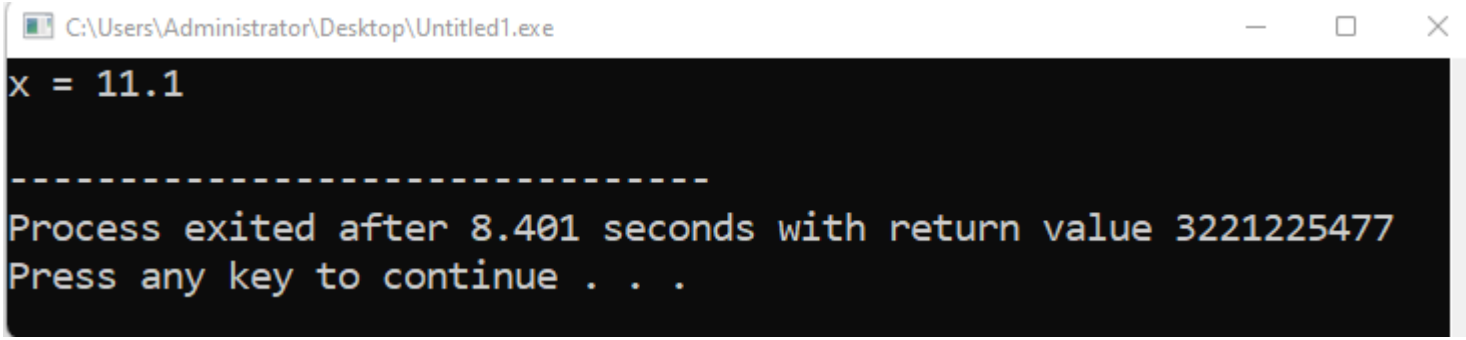
این خطا یکی از وحشت‌ناک‌ترین خطاهای زمان اجراست زیرا ممکن است اصلا نتوانیم منبع خطا را کشف کنیم. حتی ممکن است به این روش داده‌های برنامه‌های دیگری که در حال کارند را خراب کنیم و این باعث ایجاد اختلال در کل سیستم شود. به این خطا «اثر همسایگی» می‌گویند.

این وظیفه برنامه‌نویس است که تضمین کند ایندکس آرایه هیچ‌گاه از محدوده آن خارج نشود.

ایجاد استثنای مدیریت نشده

برنامه زیر از کار میافتد زیرا ایندکس آرایه خیلی بزرگ است:

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {   const int SIZE=4;
5      float a[] = { 22.2, 44.4, 66.6 };
6      float x=11.1;
7      cout << "x = " << x << endl;
8      a[3333] =88.8;//ERROR: index is out of bounds!
9      cout << "x = " << x << endl;
10 }
```



ایجاد استثنای مدیریت نشده

وقتی این برنامه روی رایانه‌ای با سیستم عامل ویندوز اجرا شود، در سیستم های قدیمی یک صفحهٔ هشدار که در شکل نشان داده شده روی صفحه ظاهر می‌شد.

این پنجره بیان می‌کند که برنامه تلاش دارد به نشانی e۰۰۴۰۱۰۸ از حافظه دستیابی کند. این مکان خارج از حافظهٔ تخصیصی است که برای این برنامه منظور شده، بنابراین سیستم عامل برنامه را متوقف می‌کند.

خطایی که در مثال قبلی بیان شده یک «**استثنای مدیریت نشده**» نامیده می‌شود زیرا کدی وجود ندارد که به این استثنا پاسخ دهد.

در C++ می‌توانیم کدهایی به برنامه اضافه کنیم که هنگام رخ دادن حالت‌های استثنا، از توقف برنامه جلوگیری کند. به این کدها «**پردازش‌گر استثنا**» می‌گویند.

مثال

مثال آرایه

۱- خواندن نمرات درس مبانی کامپیوتر ۳۰ دانشجو، نمایش نمرات به ترتیب دریافت شده همراه با میانگین نمرات.

همچنین در آخر یک نمره از کاربر دریافت شود و تعداد نمرات بیشتر از آن نمره نمایش داده شود.

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      float a[30],sum=0,avr=0,n,c=0;
6      for (int i=0;i<30;i++){
7          cout << "nomre " << i+1 <<" om: ";
8          cin >> a[i];
9          sum=sum+a[i];
10     }
11     avr=sum/30;
12     cout << "1- Avrage= " << avr << endl;
13     cout << "2- enter Your Number: ";
14     cin >> n;
15     for (int j=0;j<30;j++){
16         if(n<a[j]) ++c;
17     }
18     cout << "2- Score More Than "<<n<<" = "<<c;
19     return 0;
20 }
```

مثال آرایه

۲- خواندن درجه حرارت ۴۰ شهر، سپس سردترین شهر و شماره اندیس آن، همراه با درجه حرارت شهرها به ترتیب عکس خواندن نمایش داده شود.

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int a[40],min=1000,cindex;
6      for (int i=0;i<40;i++){
7          cout << "Degree City " << i+1 <<": ";
8          cin >> a[i];
9          if(a[i]<min) {
10             min=a[i];
11             cindex=i;
12         }
13     }
14     cout << "1- Index of Coldest= " << cindex << endl;
15     cout << "2- Degree of Coldest= "<<min<<endl;
16     for (int j=39;j>=0;j--){
17         cout << "City["<<j+1<<"]="<<a[j]<<"\t";
18     }
19
20     return 0;
21 }
```

آرایه های چند بعدی

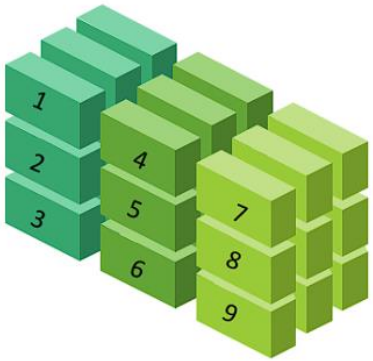
همهٔ آرایه‌هایی که تاکنون تعریف کردیم، **یک بعدی هستند، خطی هستند، رشته‌ای هستند.**

می‌توانیم آرایه‌ای تعریف کنیم که از نوع آرایه باشد، یعنی هر خانه از آن آرایه، خود یک آرایه باشد.

به این قبیل آرایه‌ها، **آرایه‌های چندبعدی** می‌گوییم.

یک آرایهٔ دو بعدی آرایه‌ای است که هر خانه از آن، خود یک آرایهٔ یک بعدی باشد.

یک آرایهٔ سه بعدی آرایه‌ای است که هر خانه از آن یک آرایهٔ دو بعدی باشد.



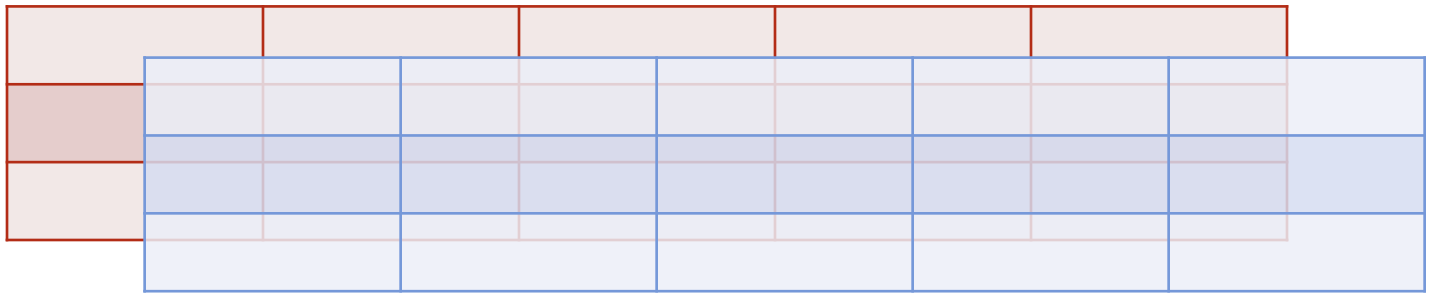
دستور int a[5]; آرایه‌ای با **پنج عنصر** از نوع int تعریف می‌کند. این یک آرایهٔ یک بعدی است.

دستور int a[3][5]; آرایه‌ای با سه عنصر تعریف می‌کند که هر عنصر، خود یک آرایهٔ پنج عنصری از نوع int است.

این یک آرایهٔ دو بعدی است که در مجموع **پانزده عضو** دارد.

چند بعدی

دستور int a[2][3][5]; آرایه‌ای با دو عنصر تعریف می‌کند که هر عنصر، سه آرایه است که هر آرایه پنج عضو از نوع int دارد. این یک آرایهٔ سه بعدی است که در مجموع سی عضو دارد.



شکل دستیابی به عناصر در آرایه‌های چند بعدی مانند آرایه‌های یک بعدی است. مثلا دستور

```
a[1][2][3] = 99;
```

مقدار 99 را در عنصری قرار می‌دهد که ایندکس آن عنصر (1,2,3) است.

تعریف آرایه و مقداردهی

```
int a[2][3]={1,12,3,14,5,16};
```

1	12	3
14	5	16

```
int b[2][3]={{1,12,3},{14,5,16}};
```

1	12	3
14	5	16

```
int c[2][3]={{1,12,3}};
```

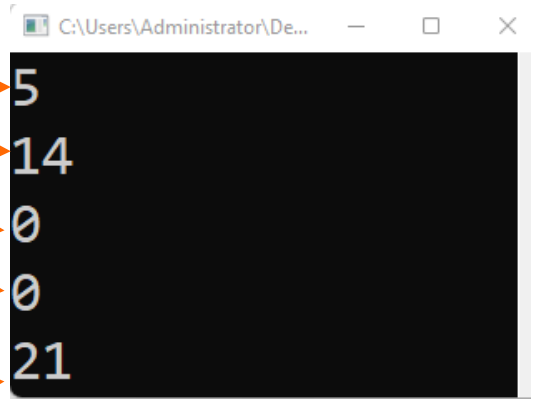
1	12	3
0	0	0

```
int d[2][3]={1,12,3};
```

1	12	3
0	0	0

آرایه دوبعدی

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int a[2][3]={1,12,3,14,5,16};
6      int b[2][3]={{1,12,3},{14,5,16}};
7      int c[2][3]={{1,12,3}};
8      int d[2][3]={1,12,3};
9      cout<<a[1][1]<<endl;
10     cout<<b[1][0]<<endl;
11     cout<<c[1][0]<<endl;
12     cout<<d[1][0]<<endl;
13     b[1][1] = 34;
14     c[1][1] = a[1][1]+b[1][2];
15     cout<<c[1][1];
16     return 0;
17 }
```



نوشتن و خواندن یک آرایهٔ دو بعدی

برنامهٔ زیر نشان می‌دهد که یک آرایهٔ دوبعدی چگونه پردازش می‌شود:

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {  int a[3][5];
5    // read array
6     cout << "Enter 15 integers, 5 per row:\n";
7     for (int i=0; i<3; i++)
8     {  cout << "ROW " << i << ": ";
9        for (int j=0; j<5; j++)
10       cin >> a[i][j];     }
11   //write array
12   for (int i=0; i<3; i++)
13    {  for (int j=0; j<5; j++)
14          cout << " " << a[i][j];
15       cout << endl;    }
16 }
```

```
C:\Users\Administrator\Desktop\dobodi.exe
Enter 15 integers, 5 per row:
ROW 0: 1 3 4 5 4
ROW 1: 5 6 3 6 8
ROW 2: 8 5 0 4 3
 1 3 4 5 4
 5 6 3 6 8
 8 5 0 4 3
```

مثال

۳- خواندن شماره دانشجویی و نمرات درس مبانی کامپیوتر ۱۰ دانشجو، نمایش نمرات و شماره دانشجویی با فرمت خوانا.

همچنین یک شماره دانشجویی از کاربر دریافت شود و نمره آن دانشجو نمایش داده شود.

```
#include <iostream>
using namespace std;
int main()
{
    int a[2][10],n,s;
    for (int i=0; i<10; i++)
    {
        cout << "StdNo " << i+1 << ": ";
        cin >> a[0][i];
        cout << "Nomre " << i+1 << ": ";
        cin >> a[1][i];
    }
    cout << "Your STDNO= ";
    cin >> n;

    cout << "StdNo\tNomre"<<endl;
    for (int j=0; j<10; j++)
    {
        cout << a[0][j] <<"\t"<< a[1][j]<<endl;
        if (a[0][j]==n) s=a[1][j];
    }
    cout <<endl<<"\t Youre Score = "<< s;
}
```

با تشکر از همراهی شما

محمد سعید صفایی صادق

www.SaeidSafaei.ir