آرایه

یکی از پرکاربردترین ساختمان های داده آرایه است که اغلب برای پیاده سازی داده های انتزاعی خطی بکار می رود .

تعریف آرایه

آرایه (array) لیست متناهی از عناصر داده‌ای هم نوع است.

محل یک عنصر درون آرایه توسط اندیس (Index) معین می شود. عنصر ai در مکان i ام آرایه قرار می گیرد به این ترتیب می توان به صورت تصادفی مقدار آرایه را بازیابی کرد. البته با روش ترتیبی هم می توان به مقادیر عناصر لیست دسترسی پیدا کرد .

به عناصر آرایه ممکن است به کمك مجموعه ای از اندیس ها مراجعه شود .

فرم کلی تعریف آرایه در زبان برنامه نویسی Pascal به صورت زیر است:

ArrayName : ARRAY [IndexType1, IndexType2,…, IndexTypen] OF Type;

IndexType مجموعه اندیس را تعیین می کند و می تواند از هر نوع اسکالری باشد. اگر دستور کامپایلر {$R+} فعال نباشد کنترلی روی اندیس های غیر مجاز نیست. اگر فعال باشد در صورت استفاده از اندیس غیرمجاز پیغام خطای Range Check Error صادر می شود .

در زبان برنامه نویسی C آرایه به صورت کلی زیر تعریف می شود:

Type ArrayName[Size1] [Size2] …[Sizen];

Size تعداد عناصر یک بعد آرایه را تعیین می کند. در زبان C کلیه اندیس ها عددی هستند و از صفر شروع می شوند (0-based indexing) و کنترلی روی محدوده اندیس ها وجود ندارد .

دراکثر زبان ها به آرایه به صورت ایستا حافظه اختصاص می دهند و اندازه آرایه در طول اجرای برنامه ثابت است. مگر اینکه حافظه پویا صریحا توسط برنامه نویس استفاده شود .

آرایه های یك بعدی

آرایه یک بعدی مجموعه متناهی از زوج ها به صورت ‹ اندیس،مقدار › است. بدین معنی که، به ازای یك اندیس یک مقدار مربوط به آن وجود دارد .

برای تعریف آرایه یك بعدی یک مجموعه اندیس تعریف می شود.

مثل (C) .آرایه num با ۲۰ عنصر صحیح تعریف شده است. عناصر آرایه در خانه های num[0], num[1], num[2],…, num[19] ذخیره می شوند .

int num [20] ;

مثل (C) .آرایه کاراکتری num با ٤ عنصر تعریف و مقداردهی شده است.

```
Char num[4]={'d','a','t','a'} ;
یا
Char num[4]={"data"};
```

آرایه های دوبعدی

یك آرایه دو بعدی مجموعه ای با m×n عنصر داده ای است که هر عنصر آن با یك جفت اندیس مشخص می شود .

آرایه دو بعدی را می توان به جدولی تشبیه کرد که دارای m سطر و n ستون است. هر سطر شامل عناصری است که اندیس های اول آنها برابر است و هر ستون شامل عناصری هستند که اندیس های دوم آنها برابر هستند .

آرایه های دوبعدی به عنوان ماتریس به کار می روند .

در تعریف آرایه دو بعدی دو مجموعه اندیس معین می شود. اندیس اول تعداد سطرها و اندیس آرایه تعداد ستون ها را مشخص می کند .

مثل (Pascal) .آرایه Table از نوع اعداد حقیقی با ٥ سطر و ٤ ستون تعریف شده است.

```
Table : Array[1..5,3..6] of Real;
یا
Table : Array[1..5] of Array [3..6] of Real;
```

مثل (C) .آرایه A از نوع اعداد صحیح با ۳ سطر و ٤ ستون

```
int A[3][4];
```

مثل (C) .آرایه دوبعدی num را می توان به دو صورت مقداردهی اولیه کرد.

```
int num [4][3]={{15,6,13},{9,17,2},{4,5,4},{10,11,12}} ;
یا
int num [4][3]={15,9,13,9,17,2,4,5,4,10,11,12};
```

آرایه های چندبعدی

آرایهn بعدی مجموعه ای از m1×m2×…×mn عنصر داده ای است که هر عنصر توسط n اندیس نظیر i1,i2,…,in مشخص می شوند. آرایه های چند بعدی در حافظه به صورت دنباله ای از خانه های پشت سر هم ذخیره می شوند .

محاسبه فضا و آدرس

هر متغیری که تعریف می شود مقداری فضای حافظه را به خود اختصاص می دهد به نوع داده متغیر بستگی دارد. برای بدست آوردن میزان فضای اشغال شده بوسیله یك آرایه کافی است که تعداد عناصر آرایه در تعداد بایت های نوع آرایه ضرب شود. تعداد عناصر آرایه را طول یا اندازه آرایه می گویند .

در زبان Pascal اندازه آرایه به صورت زیر محاسبه می شود :

A[L1..U1, L2..U2, ….Ln..Un]

$$\prod_{i=1}^{n} U_i - L_i + 1$$

در زبان C اندازه آرایه به صورت زیر محاسبه می شود:

A[M1][M2]…. [Mn]

$$\prod_{i=1}^{n} M_i$$

مثل (Pascal) .آرایه زیر ۲٤ عنصر کاراکتری دارد.

A: Array [1..2,1..3,1..2,1..2] of Char;
(2-1+1)(3-1+1)(2-1+1)(2-1+1) = 24.

مثل (C) .میزان فضای اشغال شده توسط آرایه num به صورت زیر محاسبه می شود:

int num [10][5] ;
10×5×2 = 100

نمایش آرایه ها

حافظه کامپیوتر را می توان به صورت یک آرایه یک بعدی در نظر گرفت که آدرس ها اندیس های آن هستند. بنابراین در واقع برای نمایش آرایه در حافظه، یک آرایه n بعدی در یک آرایه یک بعدی جای داده می شود. برای این کار باید اندیس های آرایه n بعدی تبدیل به آدرس حافظه شود. دو روش برای این تبدیل وجود دارد :

- روش سطری(row-major order)
- روش ستونی (column-major order)

روش سطری

در روش سطری عناصر یک سطر پشت سر هم در حافظه قرار می گیرند در نتیجه اندیس های سمت راست سریع تر حرکت می کنند .

برای بدست آوردن آدرس هر عنصر درون حافظه کامپیوتر فقط آدرس اولین عنصر آرایه لازم است. این آدرس را آدرس شروع یا α می نامیم. با استفاده از آن آدرس محل بقیه عناصر آرایه در حافظه بدست می آید .

آرایه a[2][3] را درنظر بگیرید که از آدرس α حافظه شروع شده است.

آدرس	مقدار
α + 0 × sizeof(T)	a[0] [0]
α + 1 × sizeof(T)	a[0] [1]
α + 2 × sizeof(T)	a[0] [2]
α + 3 × sizeof(T)	a[1] [0]
α + 4 × sizeof(T)	a[1] [1]
α + 5 × sizeof(T)	a[1] [2]

آدرس اولین عنصر سطر اول برابر با α + 0×sizeof(t) = α. چون در هر سطر ۳ عنصر وجود دارد بنابراین آدرس عنصر اول سطر دوم برابربا α + 3×sizeof(t) می شود .

می توان یک فرمول کلی برای آرایه n بعدی بدست آورد. اگر آرایه A[δ1] [δ2] […] [δn] از آدرس α حافظه شروع شده باشد. آدرس خانه A[i1][i2]….[in] این آرایه به صورت زیر محاسبه می شود(Wتعداد بایت‌های نوع آرایه است):

α + W [(i1)δ2δ 3… δn + (i2)δ3…δn + … + (in)]

اگر آرایه توسط زبان Pascal به صورت A[L1..U1, L2..U2, …,Ln..Un] تعریف شده باشد فرمول به شکل زیر در می آید :

$\alpha + W [(i1-L1)\delta2\delta 3\ldots \delta n + (i2-L2)\delta3\ldots\delta n + \ldots + (in-Ln)]$
$\delta i=Ui-Li$

مثال ۱. آدرس خانه های آرایه یک بعدی به طریق زیر محاسبه می شود.

آدرس $A[i] = \alpha + W(i-L)$

مثال ۲. آرایه دو بعدی A را در نظر بگیرید. اگر آرایه از آدرس ۱۰۰ ذخیره شده باشد، آدرس خانه A[12][4] به طریق زیر محاسبه می شود.

A : Array [10..15][3..5] of integer;
$\delta1=15-10+1=6$, $\delta2=5-3+1=3$

آدرس $A[i1,i2] = \alpha + W [(i1-L1)\delta2+(i2-L2)]$
آدرس $A[12,4] = 100 + 2 [(12-10)3+(4-3)]$
$=114$

مثال ۳. اگر آرایه سه بعدی num از آدرس ۲۰۰۰ به بعد ذخیره شده باشد، آدرس خانه A[3][2][1] به طریق زیر محاسبه می شود:

float A[4][5][3] ;
$\delta1=4$,$\delta2=5$, $\delta3=3$

آدرس $A[i1][i2][i3] = \alpha + W [(i1)\delta2\delta3+(i2)\delta3+(i3)]$
آدرس $A[3][2][1] = 2000 + 4 [(3)5\times3+(2)3+(1)]$
$=2208$

روش ستونی

در روش ستونی عناصر یک ستون پشت سر هم در حافظه قرار می گیرند در نتیجه اندیس های سمت چپ سریع تر حرکت می کنند .

آدرس خانه A[i1][i2]… [in] در این روش توسط فرمول زیر محاسبه می شود:

$\alpha + W [(i1) + \delta1(i2)+ \delta1\delta2(i3) + \ldots + \delta1\delta2\delta 3\ldots\delta n-1(in)]$

اگر آرایه توسط زبان Pascal تعریف شده باشد فرمول به شکل زیر در می آید :

$\alpha + W [(i1-L1) + \delta1(i2-L2)+ \delta1\delta2(i3-L3) + \ldots + \delta1\delta2\delta 3\ldots\delta n-1(in-Ln)]$
$\delta i=Ui-Li$

مثال. درمثال ۲ اگر آرایه به صورت ستونی ذخیره شده باشد:

$A[i1,i2] = \alpha + W[(i1-L1) + \delta1(i2-L2)]$
آدرس $A[12,4] = 100 + 2 [(12-10) +6(4-3)]$
$=116$

مثال. درمثال ۳ اگر آرایه به صورت ستونی ذخیره شده باشد:

آدرس A[i1][i2][i3] = α + W[(i1)+ δ1 (i2) + δ1δ2 (i3)]
آدرس A[3][2][1] = 2000 + 4 [(3) +4(2)+4×5(1)]
=2124

آرایه های پویا

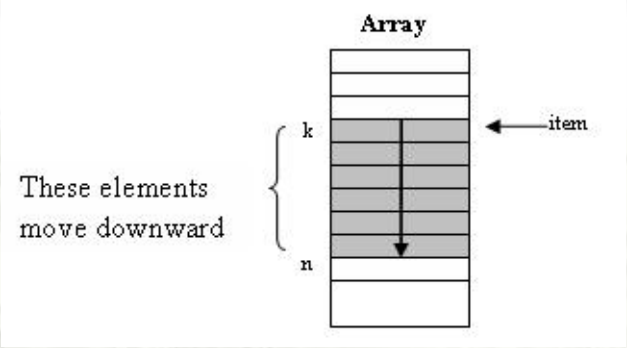
آرایه پویا (dynamic array) آرایه است اندازه اش در زمان اجرا با عمل درج یا حذف عناصردر آن تغییر می کند .

در زبان برنامه نویسی Visual Basic توسط دستور REDIM و در زبان C با تعریف حافظه پویا می توان آرایه پویا ایجاد کرد. در بعضی زبان ها مانند Perl کلیه آرایه ها به صورت پویا هستند .

الگوریتم های درج و حذف

درج عنصری در آرایه

برای درج عنصری در آرایه تعدادی از عناصر باید به سمت پائین منتقل شوند تا عنصر جدید در محل مورد نظر قرار گیرد. اگر بخواهیم عنصر جدید در مکان k ام آرایه درج شود کلیه عناصر از k به بعد باید شیفت داده شوند، سپس عنصر در مکانK ام ذخیره شود .



در کل n-k+1 عنصر باید جابجا شوند. اگر عنصر جدید در محل آخرین عنصر درج شود تنها عنصر آخر آرایه جابجا می شود. بدترین حالت زمانی اتفاق می افتد که بخواهیم عنصر جدید را درمکان اول آرایه درج کنیم در این حالت تعداد جابجائی‌ها برابر است با n می شود .

به طور متوسط نیاز به (n+1)/2جابجائی است.

با هربار عمل درج یک واحد به n تعداد عناصر آرایه اضافه می شود .تعداد عناصری که در آرایه درج شده اند را نشان می دهد و ربطی به طول آرایه ندارد .

الگوریتم زیر عنصر item را در مکان k ام آرایه A با n عنصر درج می کند .

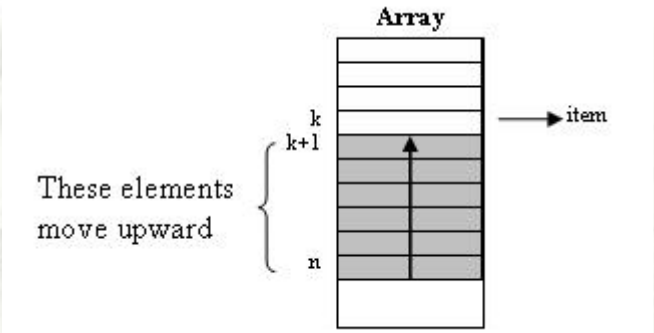
```
for (i:= n down to k )
  A[j+1] := A[j]
end for
```

```
A[k] := item
n := n +1
end
```

پیچیدگی الگوریتم فوق O(n) است.

حذف عنصری از آرایه

وقتی عنصری از آرایه حذف می شود عناصر بعدی باید به سمت بالا منتقل شوند تا محل عنصر حذف شده پر شود. برای حذف عنصری که در مکان k ام قرار دارد، کلیه عناصر از k+1 به بعد باید به سمت بالا شیفت داده شوند .



در کل n-k عنصر باید جابجا شوند. کمترین جابجائی وقتی است که عنصر آخر حذف شود که هیچ عنصری جابجا نمی شود. در بدترین حالت تعداد جابجائی‌ها برابر با n-1 است وقتی که اولین عنصر آرایه حذف می شود.
به طور متوسط نیاز به (n-1)/2 جابجائی است.
با هربار عمل درج یک واحد به n تعداد عناصر آرایه اضافه می شود n .تعداد عناصری که در آرایه درج شده اند را نشان می دهد و ربطی به طول آرایه ندارد .

الگوریتم زیر عنصر kام آرایه A را حذف و در item ذخیره می کند .

```
item := A[k]
for (j := k to n – 1)
  A[j] := A[j + 1]
end for
 n := n –1
end
```

پیچیدگی الگوریتم فوق O(n) است.
همانطور که مشاهده می شود عملیات درج و حذف در آرایه به طور متوسط منجر به انتقال نصف عناصر آرایه می شود. بنابراین در مواردی که مجموعه عناصر داده ای به طور مکرر درحال اضافه و حذف هستند آرایه خطی روش کارآمدی برای ذخیره داده ها نیست .