برنامه‌سازی کامپیوتر

جلسه دهم - بخش اول

آشنایی با اشاره‌گرها در زبان C

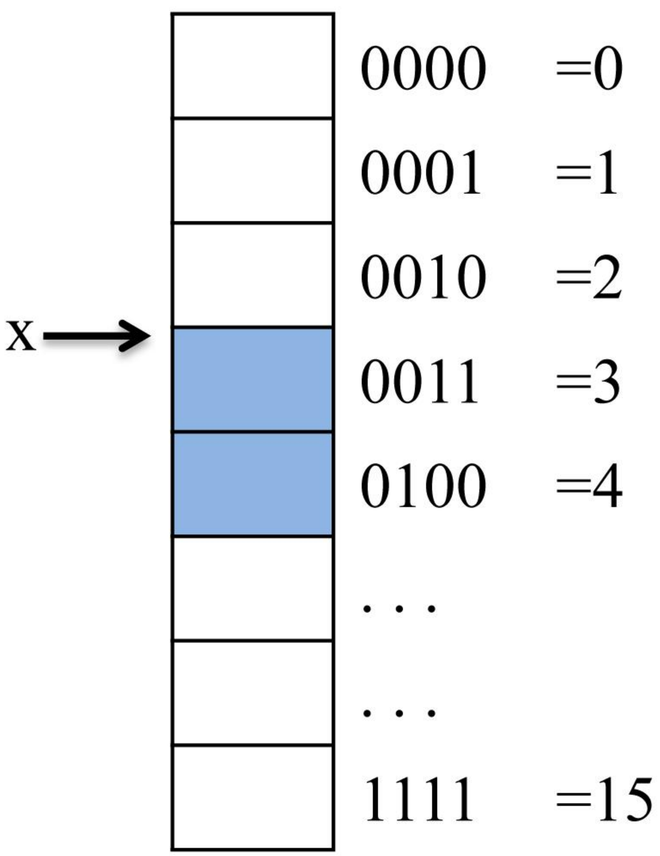
طرح کلی

- اشاره‌گرها
  - تعریف
  - کاربرد
  - عملگرها
- عملیات‌های اشاره‌گرها
  - انتساب
  - اعمال محاسباتی
  - مقایسه
- اشاره‌گرها و آرایه‌ها

مقدمه

- آدرس هر متغییر را در حافظه، اشاره‌گر گویند.

مثال : حافظه ۱۶ بایتی



int x;

- بایت یکی از تقسیمات حافظه است. به عبارت دیگر، حافظه کامپیوتر، مجموعه‌ای از چندین بایت است. هر بایت دارای یک شماره ردیف است. شماره ردیف هر بایت از حافظه را آدرس آن محل از حافظه گویند.

- متغییرها نامی برای محل‌های حافظه‌اند و لذا بایت‌هایی از حافظه را اشغال می‌کنند. آدرس اولین بایتی از حافظه که به متغییر اختصاص می‌یابد، آدرس آن متغییر است.

کاربرد اشاره‌گرها

- اغلب قابلیت‌های C به نقش اشاره‌گرها در این زبان برمی‌گردد:
  - **تخصیص حافظه پویا:** در این نوع تخصیص حافظه، برنامه در زمان اجرا از سیستم حافظه می‌گیرد و در صورت عدم نیاز، آن حافظه را به سیستم برمی‌گرداند.
  - موجب **بهبود کارآیی توابع** می‌شود (توابع را با استفاده از آدرس آن‌ها می‌توان فرآخوانی کرد).
  - سهولت کار با **رشته‌ها و آرایه‌ها**.
  - ارسال آرگومان‌ها از طریق **فراخوانی با ارجاع** را امکان‌پذیر می‌سازد.

متغییرهای اشاره‌گر

- اشاره‌گر می‌تواند در متغییری ذخیره شود. اما، با اینکه اشاره‌گر یک آدرس حافظه است و آدرس حافظه نیز یک عدد است، ولی نمی‌توان آن را در متغییرهایی از نوع int، double و یا غیره ذخیره کرد.
- متغییری که می‌خواهد اشاره‌گر را ذخیره کند باید هم‌نوع با اشاره‌گر باشد. این متغییرها را متغییرهای اشاره‌گر گویند.
- برای تعریف متغییرهای اشاره‌گر در زبان C بصورت زیر عمل می‌شود:

نوع متغییر* ;

- در تعریف متغییر اشاره‌گری که بخواهد آدرس متغییرهایی را نگه‌داری کند، باید نوع متغییر اشاره‌گر را هم‌نوع با آن متغییرها درنظر گرفت و در کنار متغییر اشاره‌گر، علامت * را قرار داد.

مثال

مثال ۱:

```
int  *p ;
```

- p متغییر اشاره‌گری از نوع int است.
- p آدرس محل‌هایی از حافظه را که محتویات آن‌ها مقادیری از نوع صحیح‌اند نگه‌داری می‌کند.
- p می‌تواند به محل‌هایی اشاره کند که محتویات آن‌ها مقادیری از نوع صحیح می‌باشند.

مثال ۲:

```
int  *p1 , *p2 , v1 , v2 ;
double   *f1 , f2 ;
char   *ch ;
```

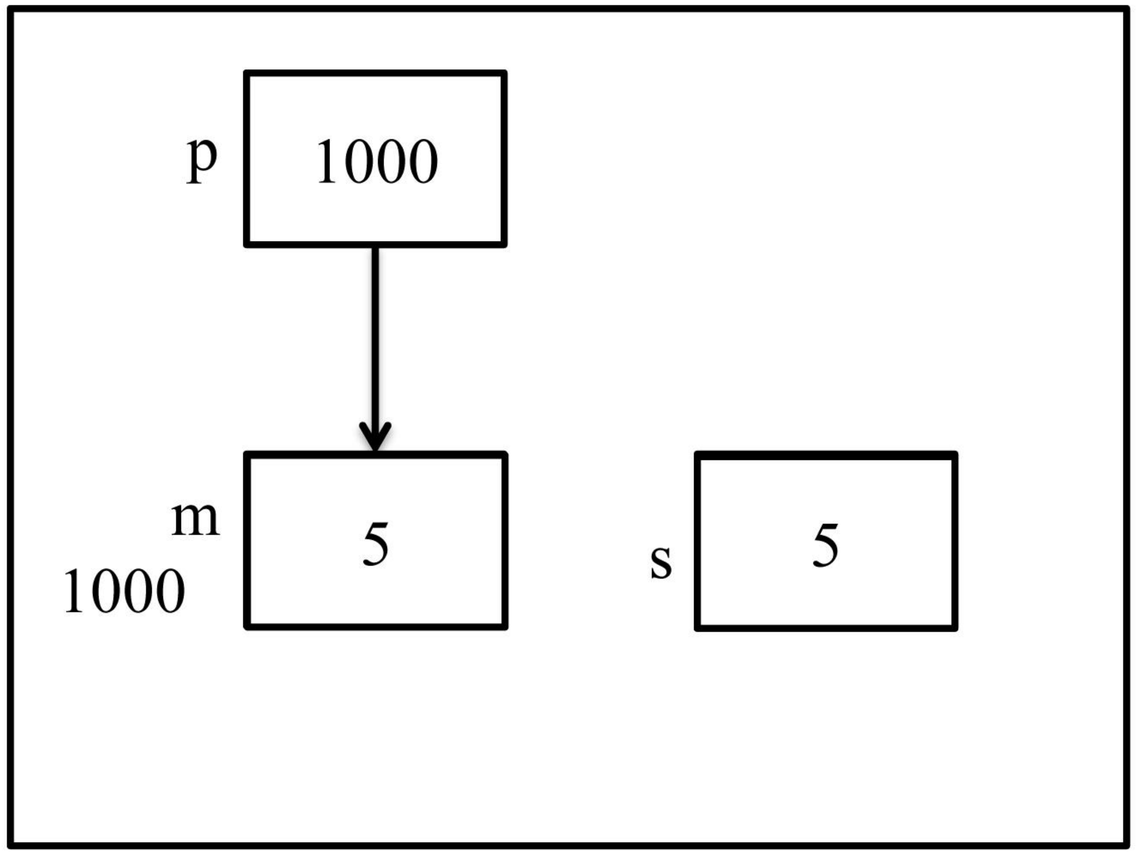
عملگرهای اشاره‌گر

عملگر & (آدرس) و * (محتویات)

- هریک از این دو عملگر، یک عملوند دارند. عملگر &، آدرس عملوند خودش را مشخص می‌کند.
- عملگر * محتویات جایی را مشخص می‌نماید که عملوندش به آن اشاره می‌کند.

مثال:

```
int   *p , m , s ;
m = 5 ;
p = &m ;
s = *p ;
```

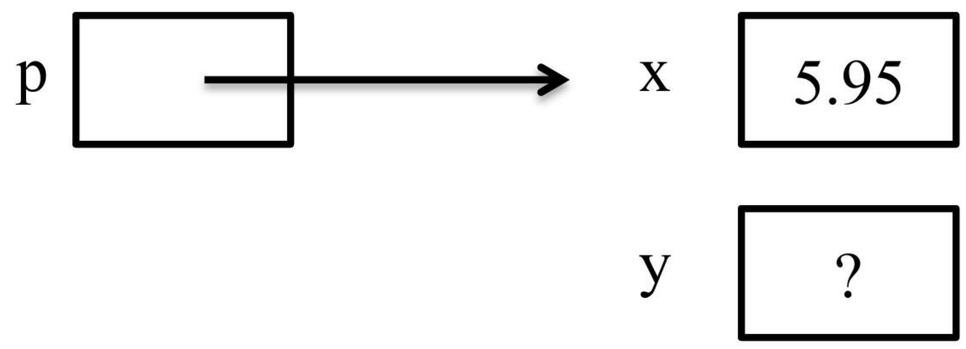


نکته

- اشاره‌گرها در زبان C دارای نوع‌اند. یعنی وقتی اشاره‌گری از نوع int تعریف می‌شود، باید به متغییرهایی از نوع int اشاره نماید و اشاره‌گر از نوع double باید به متغییرهایی از همین نوع اشاره نماید.
- اگر نوع متغییری که آدرس آن در یک اشاره‌گر قرار می‌گیرد، با نوع اشاره‌گر یکسان نباشد، کامپایلر زبان C خطایی را اعلام نمی‌کند.

مثال: خروجی قطعه کد مقابل چیست؟

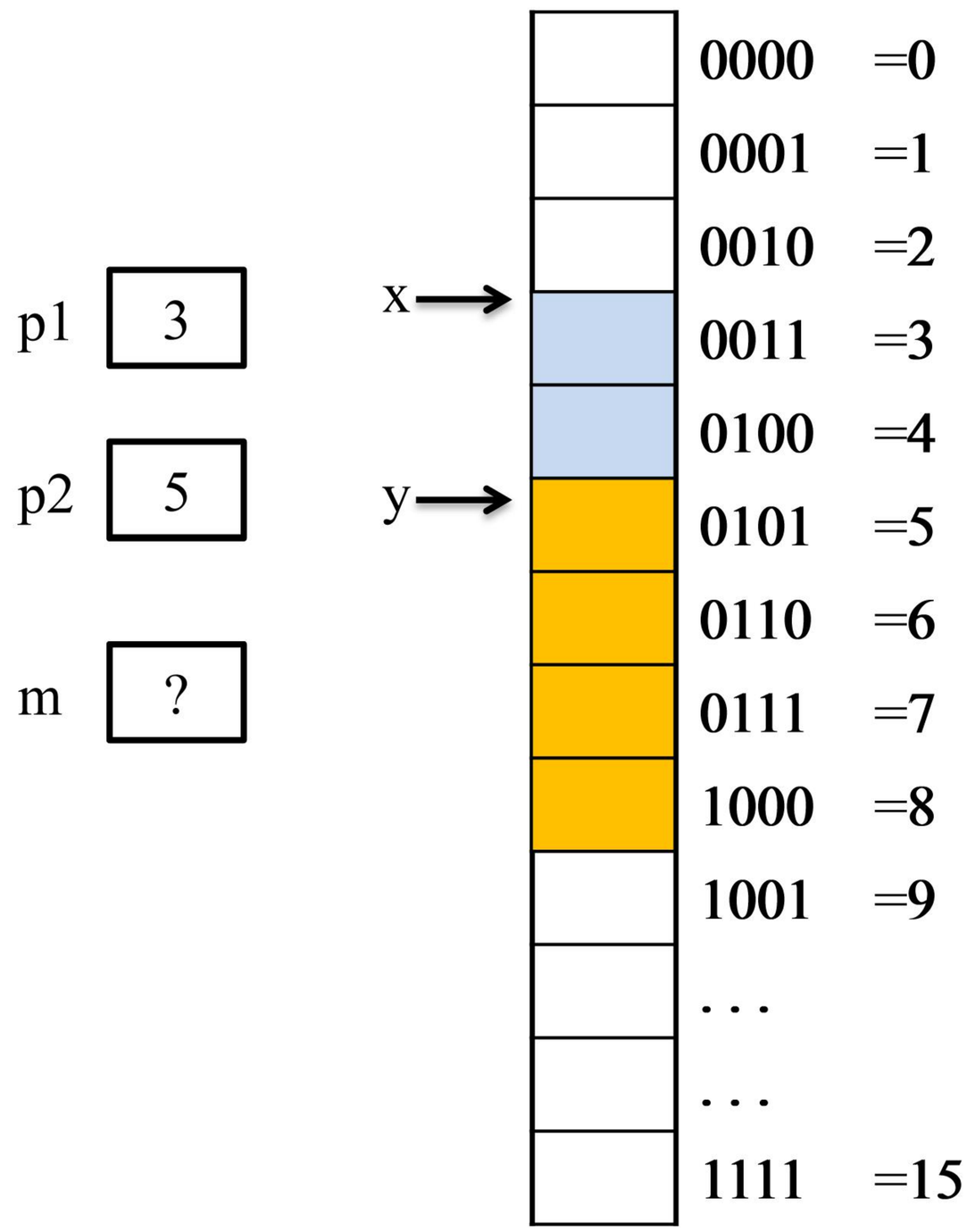
```
#include <stdio.h>
int main()
{
float   x, y ;
int   *p;
x = 5.95 ;
p = &x ;
y = *p ;
printf (“\n y = %f” , y) ;
return 0;
}
```



مثال

```
int   *p1, *p2, x , m;
float    y ;
x = 10 ;
y = 3.14
p1 = &x ;
p2 = &y ;
```

```
m = *p1;
m = *p2;
```



اعمال روی اشاره‌گرها

- تعداد اعمالی که می‌توان روی اشاره‌گرها انجام داد بسیار کمتر از اعمالی است که روی متغییرهای دیگر می‌توان انجام داد.
  - عمل انتساب اشاره‌گرها به یکدیگر.
  - اعمال محاسباتی جمع و تفریق.
  - عمل مقایسه اشاره‌گرها.

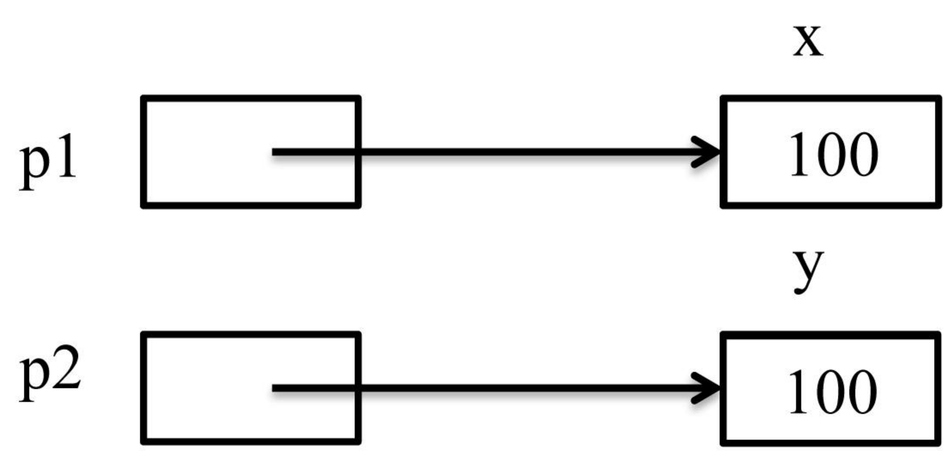
انتساب اشاره‌گرها به یکدیگر

اگر x و y دو متغییر باشند، دستور x=y آنچه را که در y قرار دارد در x قرار می‌دهد. این عمل را انتساب y به x گویند. دو اشاره‌گر را نیز می‌توان به یکدیگر نسبت داد. در این صورت هر دو اشاره‌گر به یک محل از حافظه اشاره خواهند کرد.

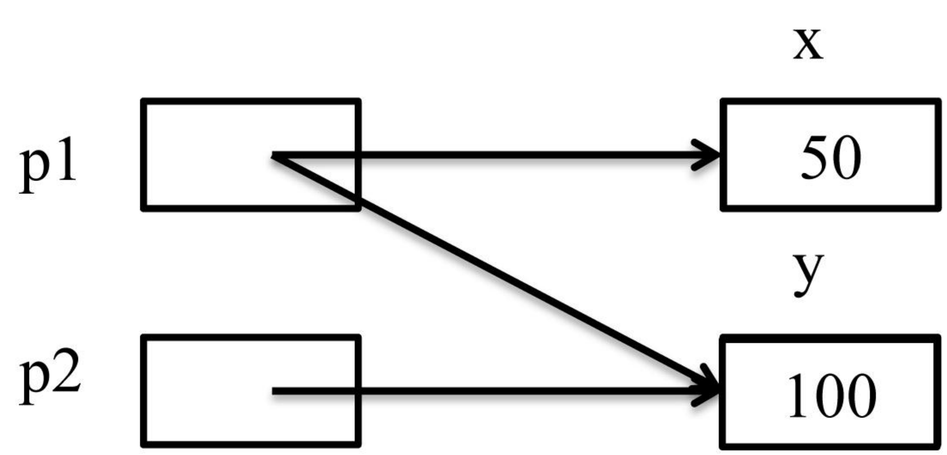
```
int   *p1 , *p2 , x , y ;
x = 50;
y = 100 ;
p1 = &x ;
p2 = &y ;
*p1 = *p2 ;
```

مثال

```
int   *p1 , *p2 , x , y ;
x = 50;
y = 100 ;
p1 = &x ;
p2 = &y ;
*p1 = *p2 ;
```




```
int   *p1 , *p2 , x , y ;
x = 50;
y = 100 ;
p1 = &x ;
p2 = &y ;
p1 = p2 ;
```

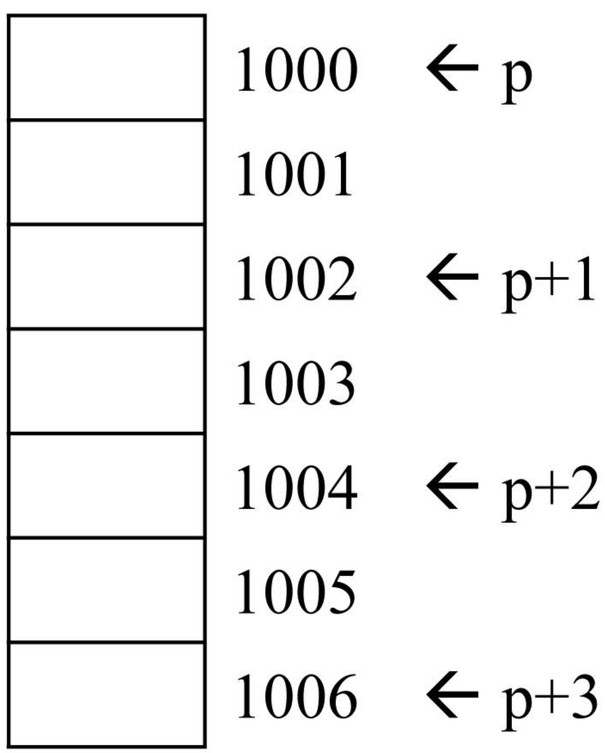


اعمال محاسباتی

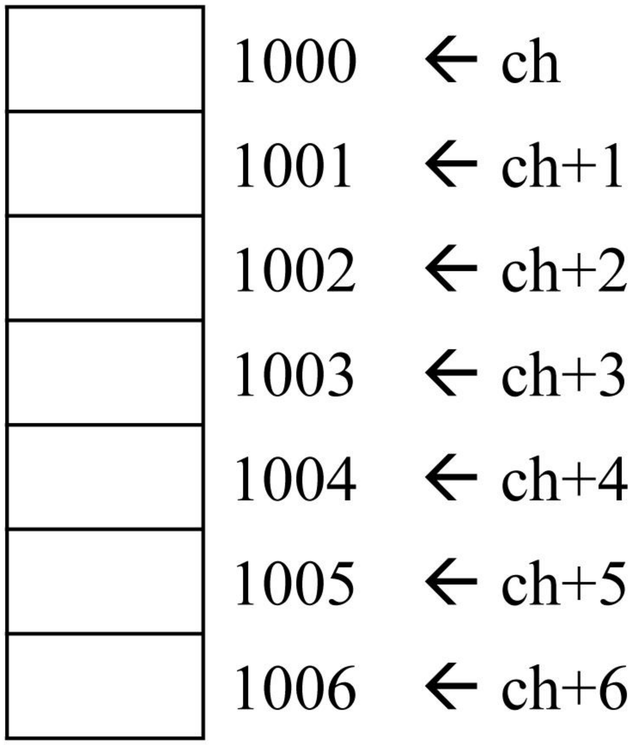
- اعمال جمع و تفریق را می‌توان برروی اشاره‌گرها انجام داد.
- با افزایش یک واحد به اشاره‌گر، به اندازه طول نوع اشاره‌گر، به آن اضافه می‌شود.
- با کاهش یک واحد به اشاره‌گر، به اندازه طول نوع اشاره‌گر، از آن کم می‌شود.

مثال:

int *p;



char *ch;

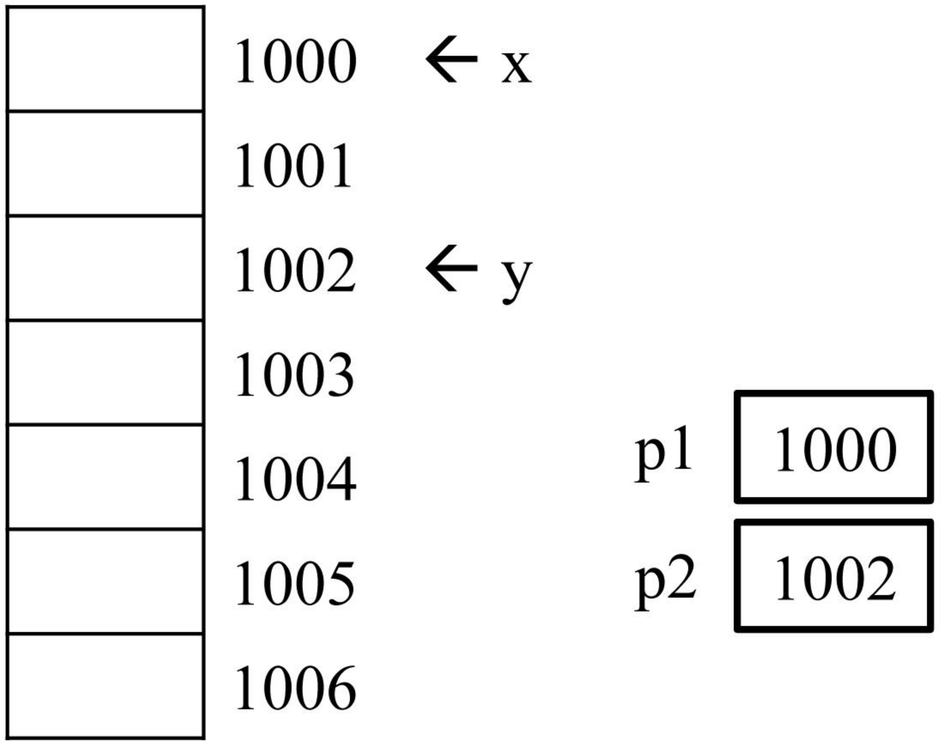


مقایسه اشاره‌گرها

اگر p1 و p2 دو اشاره‌گر باشند، با استفاده از عملگرهای رابطه‌ای همانند متغییرهای معمولی با هم مقایسه می‌شوند. یعنی مقدار آدرس هر یک از اشاره‌گرها، عدد بزرگتری باشد، آن اشاره‌گر بزرگتر خواهد بود.

مثال : با استفاده از دستورات زیر، دو اشاره‌گر p1 و p2 باهم مقایسه می‌شوند. فرض کنید p1 به محل 1000 حافظه و p2 به محل 1002 حافظه اشاره می‌کند. در این صورت شرط (p1 = = p2) ارزش نادرستی دارد.

```
int   x , y , *p1 , *p2 ;
p1 = &x ;
p2 = &y ;
if (p1 = = p2)
   . . .
else
   . . .
```



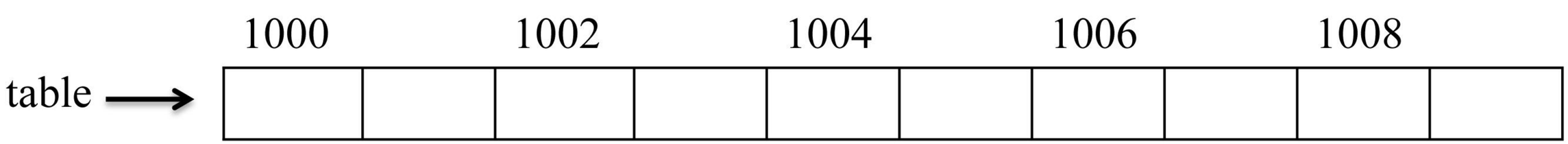
اشاره‌گرها و آرایه‌ها

- اشاره‌گرها حاوی آدرس‌اند و اسم آرایه نیز یک آدرس است.
- اسم آرایه اولین عنصر آرایه را مشخص می‌کند.
- اسم آرایه، آدرس اولین محلی را که عناصر آرایه از آنجا به بعد در حافظه ذخیره می‌شوند، نگه‌داری می‌کند.

اسم آرایه، یک اشاره‌گر است.

مثال:

```
int table [5] ;
```

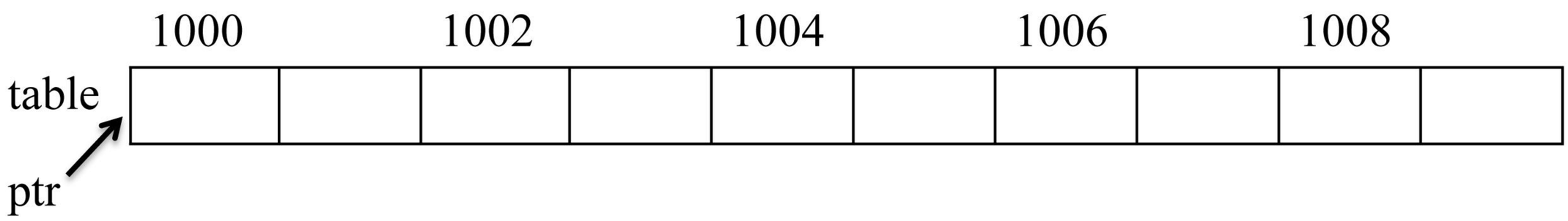


مثال

int table [5] ;

int *ptr ;

اگر اولین عنصر آرایه در محل 1000 حافظه وجود داشته باشد، table به محل 1000 حافظه اشاره خواهد کرد.



ptr = table ;			ptr و table به ابتدای آرایه اشاره می‌کنند.
*(ptr + 1)	معادل است با	table [1]	عنصر دوم آرایه
ptr [2]	معادل است با	*(table + 2)	عنصر سوم آرایه
*ptr	معادل است با	*table	عنصر اول آرایه

مثال

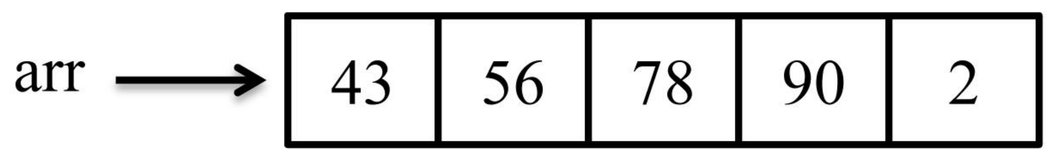
برنامه‌ای بنویسید که ۵ عدد را به‌ترتیب گرفته و آن‌ها را به‌ترتیب از آخر به اول چاپ کند.

```
#include <stdio.h>
int main()
{  int arr[5], i;
   printf("Enter five values:");
   for (i=0;i<5;i++)
     scanf("%d",&arr[i]);

   printf("\nReverse of values:");
   for (i=4;i>=0;i--)
     printf("%5d", *(arr+i));
   getch();
   return 0;
}
```

```
Enter five values:    43  56  78  90  2

Reverse of values:    2   90   78   56   43
```



جمع‌بندی

- اشاره‌گرها
  - تعریف
  - کاربرد
  - عملگرها
- عملیات‌های اشاره‌گرها
  - انتساب
  - اعمال محاسباتی
  - مقایسه
- اشاره‌گرها و آرایه‌ها