中南大學
CENTRAL SOUTH UNIVERSITY

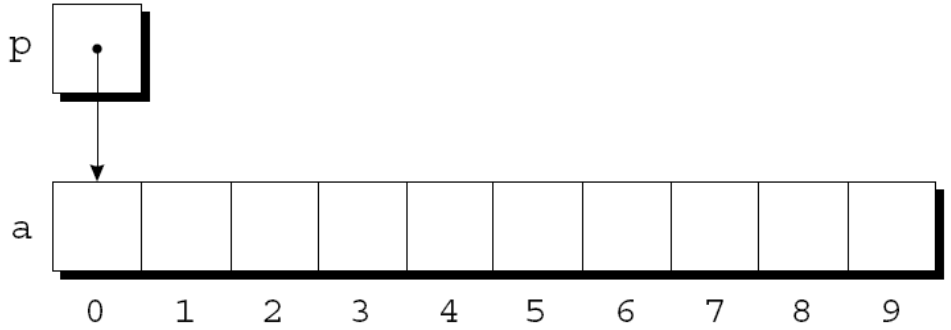
第十二讲

指　针　和　数　组

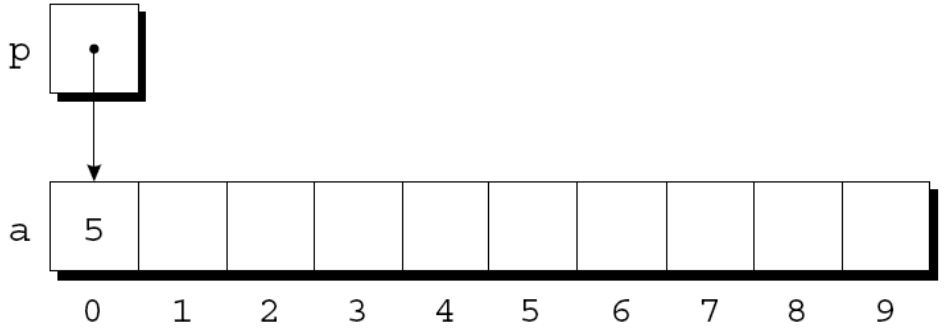
12.1 指针的算术运算

- 指针可以指向数组元素

```
int a[10], *p;
p = &a[0];
```




```
*p = 5;
```



12.1 指针的算术运算

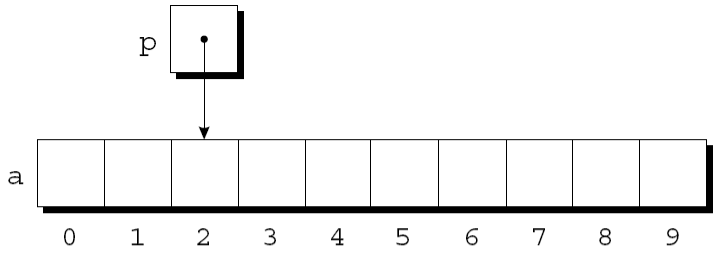
- 通过在p上执行指针算术运算可以访问数组a的其他元素
- C语言所支持的3种格式指针算术运算
  - 指针加上整数
  - 指针减去整数
  - 两个指针相减
- 假设有以下声明

```
int a[10], *p, *q, i;
```

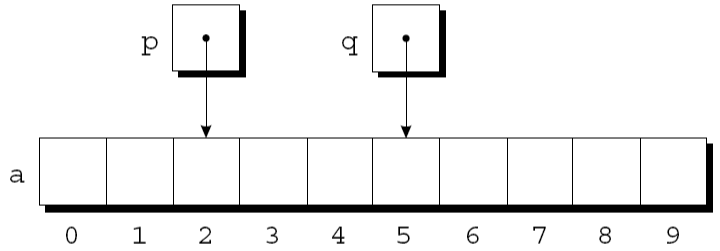
12.1 指针的算术运算

- 指针加上整数
  - 如果p指向a[i]，那么p+j指向a[i+j]（a[i+j]需存在）

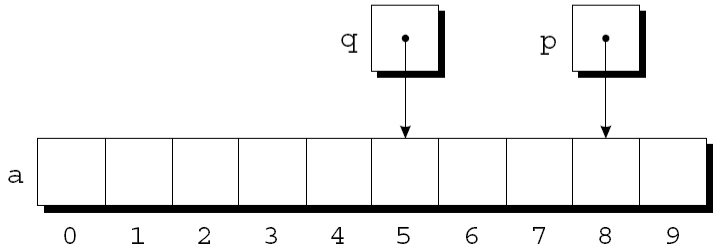
```
p = &a[2];
```




```
q = p + 3;
```



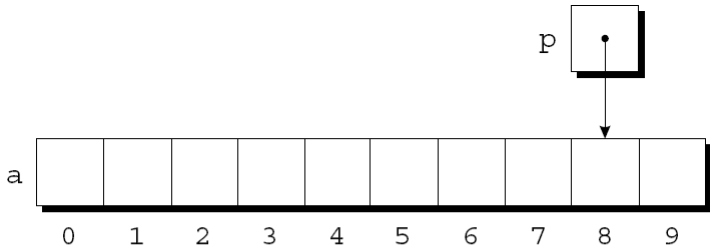

```
p += 6;
```



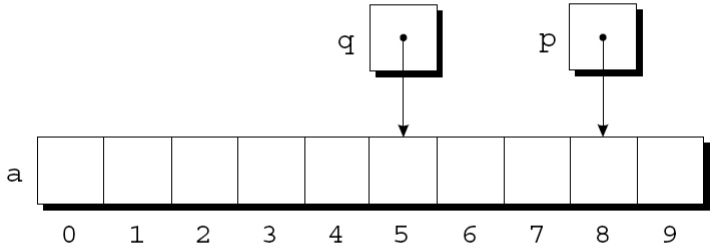
12.1 指针的算术运算

- 指针减去整数
  - 如果p指向a[i]，那么p-j指向a[i-j]（a[i-j]需存在）

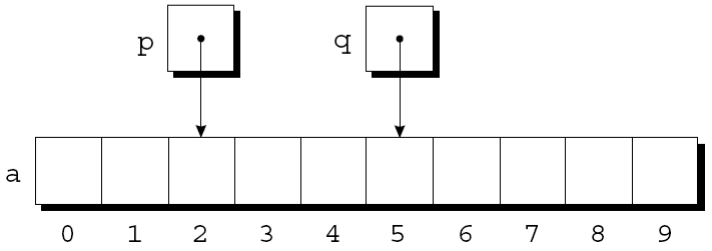
```
p = &a[8];
```




```
q = p - 3;
```



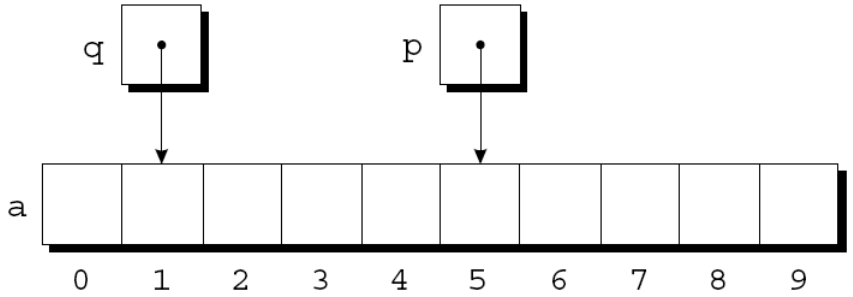

```
p -= 6;
```



12.1 指针的算术运算

- 两个指针相减
  - 当两个指针相减，结果为指针之间的距离（元素个数）

```
p = &a[5];
q = &a[1];
```




```
i = p - q;    /* i is 4 */
i = q - p;    /* i is -4 */
```

12.1 指针的算术运算

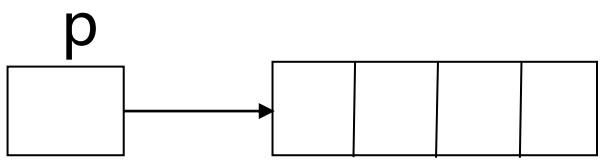
- 指针比较
  - 可以用关系运算符(<, <=, >, >=)和判等运算符(== and !=)进行指针比较
  - 只有在两个指针指向同一数组时才有意义

```
p = &a[5];
q = &a[1];
```

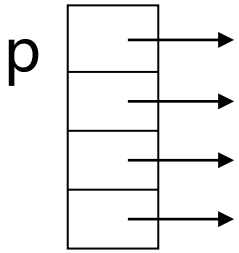
p <= q 的值是 0， p >= q 的值是 1

12.1 指针的算术运算

- 指向数组的指针
  - int (*p)[4]
  - p是指针，指向一个数组,数组有四个元素，每个元素是**整型**



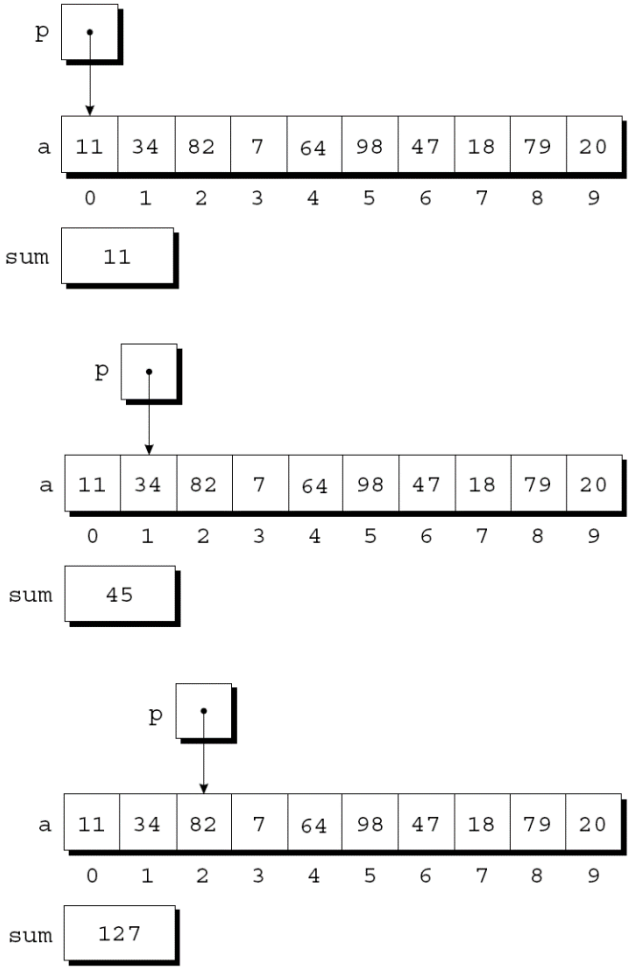
  - p+1: 指向下一行(加四个元素)
- 指针数组
  - int *p[4]
  - p是一个数组，数组的每一个元素是**指针**，指针指向整型变量



12.2 指针用于数组处理

- 通过指针变量进行重复自增来访问数组的元素

```
#define N 10
…
int a[N], sum, *p;
…
sum = 0;
for (p = &a[0]; p < &a[N]; p++)
   sum += *p;
```



12.2 指针用于数组处理

*运算与++运算符组合

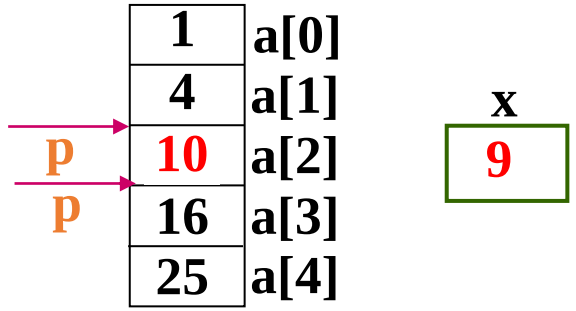
```
int a[5]={1,4,9,16,25},*p=a+2,x;
```

x=*p++; 等价于 x=*p; p++;

x=*++p; 等价于 p++; x=*p;

x=(*p)++; 等价于 x=*p; a[2]=a[2]+1;

x=++*p; 等价于 a[2]=a[2]+1; x=*p;



应用举例

```
for (p = &a[0]; p < &a[N]; p++)
        sum += *p;
```

```
p = &a[0];
while (p < &a[N])
sum += *p++;
```

12.3 用数组名作为指针

- 可以用数组名字作为指向数组第一个元素的指针

```
int a[10];
*a = 7;     /* stores 7 in a[0] */
*(a+1) = 12;    /* stores 12 in a[1] */
```

- *(a+i) 等价于 a[i]

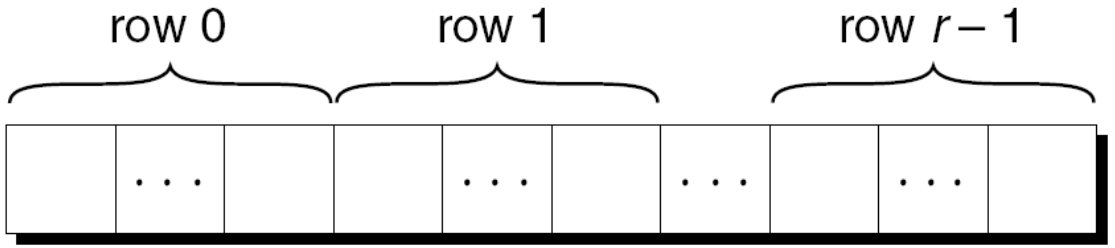
```
for (p = &a[0]; p < &a[N]; p++)
  sum += *p;
```

```
for (p = a; p < a + N; p++)
    sum += *p;
```

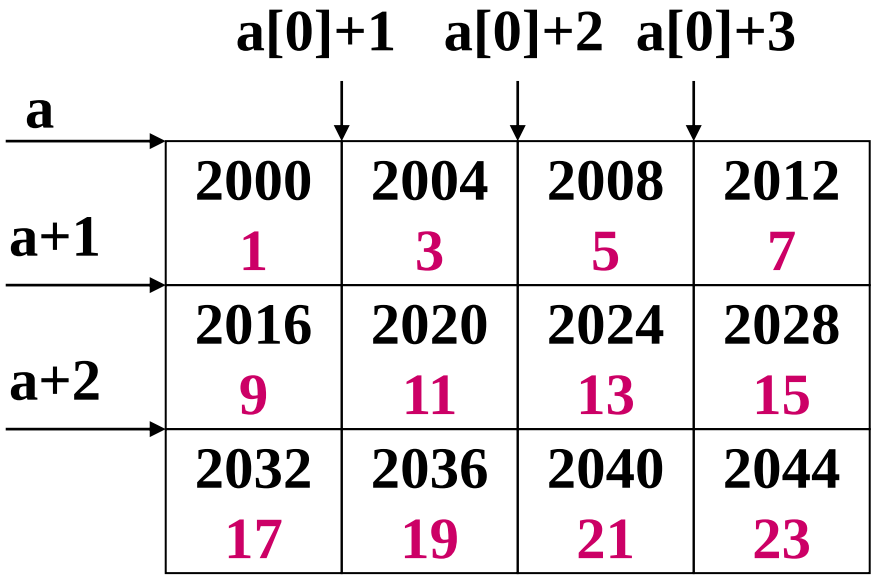
12.3指针和多维数组

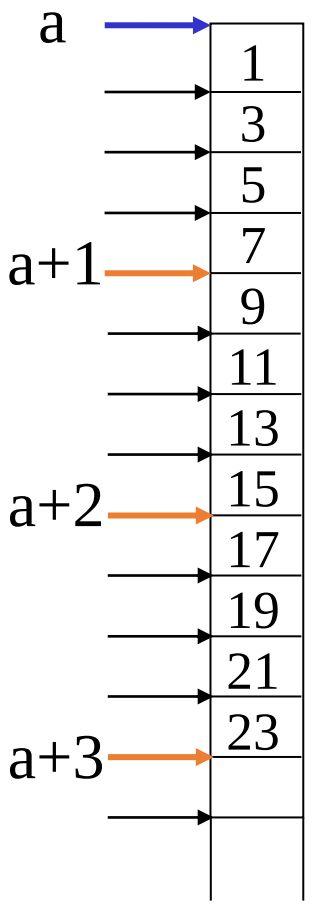
处理多维数组的元素

按照行优先存储二维数组



[例]int a[3][4]={1,3,5,7,9,11,13,15,17,19,21,23};





a: 数组名代表数组的首地址,二级指针。

a[0],a[1],a[2]: 代表对应行的首地址,一级指针。

12.3指针和多维数组

处理多维数组的元素

表示形式	含义	地址
a	二维数组第0行首地址	2000
a[0],*(a+0),*a	第0行第0列元素地址	2000
a+1	第1行首地址	2016
a[1], *(a+1)	第1行第0列元素地址	2016
a[1]+2,*(a+1)+2,&a[1][2]	第1行第2列元素地址	2024
(a[1]+2),(*(a+1)+2),a[1][2]	第1行第2列元素的值	元素值为13

比较:

a, &a[0], a[0],*a, &a[0][0], a[0][0]

a+i,&a[i], a[i],*(a+i), **(a+i)

表示第i行第j列元素:

a[i][j] *(*(a+i)+j)

*(a[i]+j) *(a+i+j) ×

12.3指针和多维数组

- 处理多维数组的元素
  - 指针p指向二维数组的第一个元素，可以通过重复自增访问二维数组每一个元素

```
int a[NUM_ROWS][NUM_COLS];
```

```
int a[NUM_ROWS][NUM_COLS];

int row, col;
…
for (row = 0; row < NUM_ROWS; row++)
  for (col = 0; col < NUM_COLS; col++)
    a[row][col] = 0;
```

```
int a[NUM_ROWS][NUM_COLS];

int *p;
…
for (p = &a[0][0];
     p <= &a[NUM_ROWS-1][NUM_COLS-1];p++)
  *p = 0;
```

12.3指针和多维数组

- 处理多维数组的行
  - 为了访问第i行元素看需要初始化p指向数组a中第i行的第0列元素

```
p = &a[i][0];
```

简化：

```
p = a[i]
```

```
int a[NUM_ROWS][NUM_COLS], *p, i;
…
for (p = a[i]; p < a[i] + NUM_COLS; p++)
  *p = 0;
```

12.3指针和多维数组

- 处理多维数组的列

```
int a[NUM_ROWS][NUM_COLS], (*p)[NUM_COLS], i;
…
for (p = &a[0]; p < &a[NUM_ROWS]; p++)
  (*p)[i] = 0;
```