编程笔记

1小技巧

```
①
int sum(int max)
{
if(max>1)
{return max + sum(max-1);}
else
return 1;
}
②这里指针变量指向一个格式字符串，用在printf函数中。
PF可以代替"%d,%d,%d,%d,%d\n"了
char *PF;
PF="%d,%d,%d,%d,%d\n";
printf("%d,%d,%d,%d,%d\n",a,*a,a[0],*a[0],&a[0][0]);
printf(PF,a,*a,a[0],*a[0],&a[0][0]);
这里两个输出结果是一样的
```

2.

数组赋值，可以不从零开始赋值，但没有被赋值到的，就是一个如-858993460之类的数，被赋值到的可以正常使用

而后面的也是如此，比如，如果前面的被赋值了5个，后面的没有赋值，要输出20个，那没有被赋值到的输出的数就是-858993460了。

```
for(i=1;i<=N;i++)//为方便计，从1起
    r[i]=1;
```

经常有人这样做，为了方便起，从1开始，而相关使

用和输出，也从1开始，这样比较符合常人的思维习惯.

3.结构体数组的定义以及初始化

没有赋值的部分自动成为0

```
#include <stdio.h>
typedef struct {
    int a ;
    int b;
} str_t;
int main()
{
    int i;
    str_t st[12] = {{1},{1}};
     for(i = 0; i < 12; i++){
         printf("%d\n", st[i].a);
    }
    return 0;
}
```

```
1
1
0
0
0
0
0
0
0
0
0
0
Press any key to continue
```

4.关于二维数组的行地址与行指针

当一个量到了具体的一个地址时，再给整体加上一个*，才能取出改地址上的值

如　　*(*(a+1)+1)　　*(a[1]+1)　　*(*(a+i)+j)

否则都还只是地址。

如　　*(a+1)+1　　a[1]+1　　a+1　*a　a　　a[1]　　&a[0]　&a[0][0]

5.关于指针的*和++，--

*p++, 由于++和*同优先级，结合方向自右而左，等价于*(p++)。

(p++)和(++p)作用不同。若p的初值为a，这*(p++)等价于a[0],*(++p)等价于a[1]

(*p)++表示p所指向的元素值加一

若果p当前指向a数组中的第i个元素，则

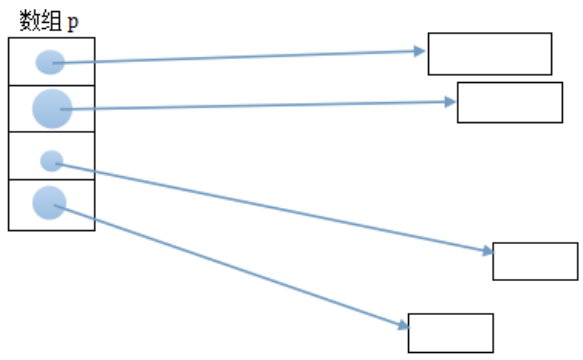
- *(p--)=a[i--],
- *(++p)=a[++i]
- *(--p)=a[--i]

6.指针数组与指向多维数组的指针变量

int a[4];

int (*p)[4]; //定义一个指针变量，指向一个包含4个int型数据的一维数组

int *p[4]; //定义一个一维数组。数组的每一个元素都是一个int *x型的指针变量（指针数组）



（b）指针数组